Computing has grown to be pervasive throughout engineering, science, business, society, and entertainment. The availability of relatively inexpensive high performance computing capabilities, ubiquitous high speed and wireless networking, and mobile computing have powered a technology-driven restructuring of the way society and most professions now operate. Information, and its associated processing and transport, is the commodity upon which corporations are built and lives are improved. At the center of this revolution, making it happen, are those who study computer science.

There are two dimensions to the field of computer science that establish it as a unique area. CS can be viewed as a stand-alone discipline worthy of study unto itself, and/or as an empowering discipline to be studied in conjunction with other areas. Core CS careers include (but are not limited to) software design and development, computer systems engineering or administration, and information security. Application areas span a wide range of fields and disciplines such as robotics, medical and health informatics, scientific research, data analytics, entrepreneurship, gaming/entertainment, and business computing to name a few.

Because computer science is a highly diverse and broadly applied field, studies can proceed in many different directions. Accordingly, the undergraduate and graduate programs in the Department of Computer Science at Johns Hopkins are flexible curricula designed to accommodate a wide range of goals. A student at Johns Hopkins can pursue appropriately customized versions of the following computer science programs: minor, bachelor of science, bachelor of arts, masters of science in engineering, and doctor of philosophy. Most of this catalog section is devoted to details regarding these programs.

Computer science research laboratories are currently active in the following areas at Hopkins: algorithm design and analysis, human-computer interaction, machine learning, data intensive computing, health informatics, computational medicine, computer vision and image processing, computer graphics, geometric modeling, programming languages, natural language and speech processing, information retrieval, cryptography and information security, secure and robust systems, storage systems, high-performance computing clusters, robots and computer programming, computational genomics, networks and distributed systems, stream processing, robotics, and computer-integrated surgical systems.

Additionally, interdisciplinary research centers in the university have heavy involvement by Computer Science faculty: the Information Security Institute (ISI) (https://isi.jhu.edu), the Laboratory for Computational Sensing and Robotics (LCSR) (https://lcsr.jhu.edu), the Center for Language and Speech Processing (CLSP) (https://www.clsp.jhu.edu), the Institute for Data Intensive Engineering and Science (IDIES) (http://idies.jhu.edu), the Institute for Computational Medicine (ICM) (https://icm.jhu.edu), the Center for Computational Biology (CCB) (http://ccb.jhu.edu), and the Malone Center for Engineering in Healthcare (MCEH) (https://malonecenter.jhu.edu). An important component of the educational process in the department is the opportunity for undergraduate and graduate student participation in the research programs of the faculty. In particular, original research in close association with individual faculty members is emphasized at the graduate level.

There are several closely related programs at the undergraduate and graduate levels which involve significant coursework and faculty involvement from the Department of Computer Science. The Laboratory for Computational Sensing and Robotics (LCSR) offers a minor in robotics and also a minor in computer integrated surgery through the Engineering Research Center for Computer Integrated Surgical Systems and Technology. Details of these programs may be found elsewhere in this catalog in the section pertaining to the Laboratory for Computational Sensing and Robotics (http://e-catalog.jhu.edu/departments-program-requirements-and-courses/engineering/robotics-computational-sensing).

Undergraduates with a strong interest in system design and performance may elect to pursue a bachelor degree in computer engineering (http://e-catalog.jhu.edu/departments-program-requirements-and-courses/engineering/electrical-computer-engineering). This field of study includes course work in computer science, as well as electrical and computer engineering. Although jointly administered by both departments, specific goals and requirements of the computer engineering degree may be found in the catalog section pertaining to the Department of Electrical and Computer Engineering only.

At the graduate level, the LCSR (http://e-catalog.jhu.edu/departments-program-requirements-and-courses/engineering/robotics-computational-sensing) offers a Master of Science in Engineering (M.S.E.) in Robotics, designed for students from a wide variety of engineering, scientific, and mathematical backgrounds to advance their interdisciplinary knowledge in robotics. Details of this program may be found in the LCSR section of this catalog, or on the web at www.lcsr.jhu.edu/MSE. Lastly, the Master of Science in Security Informatics (M.S.S.I.) is a specialized graduate program offered through the Information Security Institute (http://e-catalog.jhu.edu/departments-program-requirements-and-courses/engineering/information-security-institute) (ISI) in the WSE. The field of security informatics is fundamentally based on information security and assurance technologies (hardware, software, and networks) as related to issues such as policy, management, privacy/trust, health care, and law, from both national and international perspectives. Interested students can obtain detailed information regarding the M.S.S.I. online at https://isi.jhu.edu or in the ISI section of this catalog.

For additional information regarding the academic programs available in Computer Science, and the facilities provided, please consult the sections which follow, or the departmental website cs.jhu.edu.

Facilities

The CS department is primarily housed in Malone Hall, a state of the art, open concept research facility. Additional department research space is located in the adjacent Hackerman Hall.

The general department computing facilities include numerous workstations and servers. Two undergraduate laboratories combine to provide approximately 24 Linux workstations. One of these is a collaboration room allowing students to work in a team-based environment, with several private breakout rooms as well. At the graduate level there is a Master’s Lab consisting of a collaboration area and workstation area, both consisting of several Linux workstations. All Ph.D. students are assigned dedicated desks in their research labs.

Focused research laboratories have significant resources that provide greater specialization, including isolated networks of PCs for security studies, high-performance computing clusters, robots and computer vision systems, a mock operating room equipped with medical robots and imaging equipment, and more.
The general department computing facilities are tied together by our own LAN, and access to specialized hardware in other departments, labs, and institutions is available via the university intranet and the Internet. In addition, the university provides wireless access to the JHU intranet and the Internet, as well as server systems that provide e-mail accounts for all students.

**Undergraduate Programs**

(See also General Requirements for Departmental Majors [http://catalog.jhu.edu/undergrad-students/academic-policies/requirements-for-a-bachelors-degree](http://catalog.jhu.edu/undergrad-students/academic-policies/requirements-for-a-bachelors-degree))

The objectives of our bachelor degree programs are to train computer scientists who will be able to:

- Successfully engage in professional practice in the computing sciences or apply computer science tools and techniques to another field of interest.
- Pursue advanced study in the computing sciences.
- Work successfully in both independent and team environments.
- Lead teams and provide vision for innovation.
- Behave in a professional and ethical manner.

A successful major program of study leads to either the Bachelor of Science in Computer Science (B.S.) or the Bachelor of Arts in Computer Science (B.A.). Students should decide which degree program to complete by about their junior year. Both degree programs require specific courses and/or credits in several key areas: computer science, math, basic science, humanities and social sciences. However, there is much flexibility in how these requirements are fulfilled. Undergraduate majors may choose to pursue a broad selection of computer science and distributional courses, or to pursue a focus area within the field. Current foci primarily reflect departmental and school research strengths: big data, computational biology, fundamentals of computing, information security, natural language processing, robotics, systems and networking; while a few are directed towards career paths: software engineering, entrepreneurship/business computing, and video game design. Further information on these focus areas may be found in the computer science undergraduate advising manual (http://www.cs.jhu.edu/undergraduate-studies/academics/ugrad-advising-manual).

All undergraduate students majoring or minoring in computer science must have a faculty advisor in the department. They will be assigned an advisor as entering freshmen or upon deciding on the major/minor. Every major must follow a program approved by their faculty advisor.

The department also offers a minor in computer science, and tangentially, a minor in computer integrated surgery and a minor in robotics. Some students majoring in computer science may be eligible for a combined bachelor's/master's degree program. Requirements for these programs are included here as well. Additional details regarding undergraduate programs can be found in the department's undergraduate advising manual (http://www.cs.jhu.edu/undergraduate-studies/academics/ugrad-advising-manual) or on the website at [www.cs.jhu.edu](http://www.cs.jhu.edu).

**Requirements for the B.S. Degree**

The Bachelor of Science in Computer Science degree program is accredited by the Computing Accreditation Commission of ABET, www.abet.org (http://www.abet.org). It provides for the acquisition of the following knowledge base and skill set:

- Analyze a complex computing problem and apply principles of computing and other relevant disciplines to identify solutions.
- Design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program’s discipline.
- Communicate effectively in a variety of professional contexts.
- Recognize professional responsibilities and make informed judgments in computing practice based on legal and ethical principles.
- Function effectively as a member or leader of a team engaged in activities appropriate to the program’s discipline.
- Apply computer science theory and software development fundamentals to produce computing-based solutions.

To meet the course credit requirements for the B.S. in computer science, the student must complete a minimum of 126 credits. The basic requirements for the B.S. degree are as follows:

<table>
<thead>
<tr>
<th>Course</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Computer Science</strong></td>
<td>42</td>
</tr>
<tr>
<td><strong>Mathematics</strong></td>
<td>24</td>
</tr>
<tr>
<td><strong>Basic Sciences</strong></td>
<td>16</td>
</tr>
<tr>
<td><strong>Humanities/Social Sciences</strong></td>
<td>18</td>
</tr>
<tr>
<td><strong>Two Writing Intensive Courses</strong></td>
<td>26</td>
</tr>
<tr>
<td><strong>Electives</strong></td>
<td>26</td>
</tr>
<tr>
<td><strong>Total Credits</strong></td>
<td>126</td>
</tr>
</tbody>
</table>

Details and course recommendations for these distributional requirements below. These requirements add up to 100 credits and fulfill general university and WSE requirements, leaving 26 pure elective credits. Electives should not be taken on an S/U basis. By university policy, no more than 18 D or D+ credits can be counted toward the total credit requirements for a degree. The Courses and Curriculum Planning section (https://www.cs.jhu.edu/course-info) of the departmental Course Information webpage has lists of course area designations (Applications, Reasoning, Software, Systems, Theory), approved science courses, and courses approved as “CS other.”

**Computer Science (42 credits)**

CORE: The following foundational courses in computer science must be included in a student’s program:

<table>
<thead>
<tr>
<th>Course</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>EN.601.104</td>
<td>Computer Ethics (or Practical Ethics for Future Leaders which does not count towards CS credits)</td>
</tr>
<tr>
<td>EN.500.112</td>
<td>Gateway Computing: JAVA (or equivalent)</td>
</tr>
<tr>
<td>EN.601.220</td>
<td>Intermediate Programming</td>
</tr>
<tr>
<td>EN.601.226</td>
<td>Data Structures</td>
</tr>
<tr>
<td>EN.601.229</td>
<td>Computer System Fundamentals</td>
</tr>
<tr>
<td>EN.601.231</td>
<td>Automata &amp; Computation Theory</td>
</tr>
<tr>
<td>EN.601.433</td>
<td>Intro Algorithms</td>
</tr>
<tr>
<td><strong>UPPER: At least 16 credit hours must be at the 300-level or above, including EN.601.433. At least one course in three different classification areas (Applications, Reasoning, Software, Systems, Theory) must be chosen. An exhaustive list of the area classifications for each of our courses may be found on the department’s website linked above. They will also be encoded as POS (program of study) tags in SIS. [Note that course descriptions include old area designators which may have changed in 2019.]</strong></td>
<td>13</td>
</tr>
</tbody>
</table>
TEAM: Students must take at least one of the following courses to satisfy the team requirement. The course satisfying this requirement may overlap other requirements.

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>EN.601.290</td>
<td>User Interfaces and Mobile Applications</td>
<td>3</td>
</tr>
<tr>
<td>EN.601.295</td>
<td>Developing Health IT Applications</td>
<td>3</td>
</tr>
<tr>
<td>EN.601.310</td>
<td>Software for Resilient Communities</td>
<td>3</td>
</tr>
<tr>
<td>EN.601.411</td>
<td>Computer Science Innovation &amp; Entrepreneurship II</td>
<td>3</td>
</tr>
<tr>
<td>EN.601.421</td>
<td>Object Oriented Software Engineering</td>
<td>3</td>
</tr>
<tr>
<td>EN.601.447</td>
<td>Computational Genomics: Sequences</td>
<td>3</td>
</tr>
<tr>
<td>EN.601.490</td>
<td>Introduction to Human-Computer Interaction</td>
<td>3</td>
</tr>
<tr>
<td>EN.580.437</td>
<td>Neuro Data Design I (counts as &quot;CS other&quot;)</td>
<td>4</td>
</tr>
<tr>
<td>EN.580.438</td>
<td>Neuro Data Design II (counts as &quot;CS other&quot;)</td>
<td>4</td>
</tr>
</tbody>
</table>

CS ELECTIVES: Eight additional credits of Computer Science are required.

Mathematics (24 credits)
The following courses or equivalent substitutes such as AP credit must be included:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>AS.110.108</td>
<td>Calculus I</td>
<td>4</td>
</tr>
<tr>
<td>AS.110.109</td>
<td>Calculus II (For Physical Sciences and Engineering)</td>
<td>4</td>
</tr>
<tr>
<td>EN.553.171</td>
<td>Discrete Mathematics</td>
<td>4</td>
</tr>
</tbody>
</table>

The remaining courses must be 200-level or above, chosen from Mathematics (AS.110.xxx) or Applied Math and Statistics (EN.553.xxx), and must include coverage of both probability and statistics. Note that students will need at least six courses to fulfill the credit requirement. Some highly recommended math electives are:***

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>AS.110.201</td>
<td>Linear Algebra</td>
<td>4</td>
</tr>
<tr>
<td>EN.553.420</td>
<td>Introduction to Probability</td>
<td>4</td>
</tr>
<tr>
<td>EN.553.430</td>
<td>Introduction to Statistics</td>
<td>4</td>
</tr>
</tbody>
</table>

Basic Sciences (16 credits)
At least two semesters of physics or two semesters of chemistry, with the associated laboratories, must be included. The remaining courses must be chosen in accordance with the list posted on the department's website, which includes most 'N' (natural science) designated courses in the Sciences and Engineering, but not all. At most 2 credits from S/U intersession courses may be used to fulfill this requirement.

Humanities/Social Sciences (18 credits)
As per WSE requirements, six courses in the Humanities and Social and Behavioral Sciences must be taken, with each course at least 3 credits.**** These courses must have either Humanities ('H') or Social and Behavioral Sciences ('S') area (or both) designators on them. Foreign language courses (without an 'H' or 'S') may also be used to satisfy this requirement.

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>EN.500.112</td>
<td>Gateway Computing: JAVA (or equivalent)</td>
<td>3</td>
</tr>
<tr>
<td>EN.601.220</td>
<td>Intermediate Programming</td>
<td>4</td>
</tr>
<tr>
<td>EN.601.226</td>
<td>Data Structures</td>
<td>4</td>
</tr>
<tr>
<td>EN.601.229</td>
<td>Computer System Fundamentals</td>
<td>3</td>
</tr>
<tr>
<td>EN.601.231</td>
<td>Automata &amp; Computation Theory</td>
<td>3</td>
</tr>
<tr>
<td>EN.601.433</td>
<td>Intro Algorithms</td>
<td>3</td>
</tr>
</tbody>
</table>

UPPER: At least 15 credit hours must be at the 300-level or above, including EN.601.433.

Mathematics (20 credits)
The following courses must be included:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>AS.110.108</td>
<td>Calculus I</td>
<td>4</td>
</tr>
</tbody>
</table>

** No more than 6 independent type credits (courses numbered 601.5xx) and no more than 3 credits of short courses (1-credit special topics courses) can be counted toward this requirement. However, B.S. students doing the Senior Honors Thesis (EN.601.519-EN.601.520) may use an additional three credits of independent work toward their CS requirements, for a total of 9 credits.

No courses with grades below C- or with S/U grades can be used to fulfill this requirement unless they are not offered for a grade. At most 4 S/U credits may be applied towards this requirement.

** Up to 6 of the 42 required credits may be from an approved list of relevant courses in other departments, which includes some courses cross-listed in CS. These courses may only count as "CS other" credits, not upper level CS credits (regardless of course level in the other department). See department website for the list.

**** See the Distribution tab in the Requirements for a Bachelor's Degree (http://e-catalog.jhu.edu/undergrad-students/academic-policies/requirements-for-a-bachelors-degree) section for two exceptions to the rule that each H/S distribution course be at least 3 credits.

Requirements for the B.A. Degree
To meet the course credit requirements for the B.A. in computer science, the student must complete a minimum of 120 credits. The basic requirements for the B.A. degree are:

<table>
<thead>
<tr>
<th>Department</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computer Science</td>
<td>32</td>
</tr>
<tr>
<td>Mathematics</td>
<td>20</td>
</tr>
<tr>
<td>Basic Sciences</td>
<td>12</td>
</tr>
<tr>
<td>Humanities/Social Sciences</td>
<td>18</td>
</tr>
<tr>
<td>Foreign Language</td>
<td>6</td>
</tr>
<tr>
<td>4 Writing Intensive Courses</td>
<td></td>
</tr>
<tr>
<td>Electives</td>
<td>32</td>
</tr>
<tr>
<td>Total Credits</td>
<td>120</td>
</tr>
</tbody>
</table>

Details and course recommendations of these distributional requirements are below. These requirements add up to 88 credits and fulfill general university distribution requirements. Except for electives, courses should not be taken on a S/U basis. By university policy, no more than 18 D or D+ credits can be counted toward the total credit requirements for a degree.

Computer Science (32 credits) *
CORE: The following foundational courses in computer science must be included in a student's program:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>EN.500.112</td>
<td>Gateway Computing: JAVA (or equivalent)</td>
<td>3</td>
</tr>
<tr>
<td>EN.601.220</td>
<td>Intermediate Programming</td>
<td>4</td>
</tr>
<tr>
<td>EN.601.226</td>
<td>Data Structures</td>
<td>4</td>
</tr>
<tr>
<td>EN.601.229</td>
<td>Computer System Fundamentals</td>
<td>3</td>
</tr>
<tr>
<td>EN.601.231</td>
<td>Automata &amp; Computation Theory</td>
<td>3</td>
</tr>
<tr>
<td>EN.601.433</td>
<td>Intro Algorithms</td>
<td>3</td>
</tr>
</tbody>
</table>

** No more than 6 independent type credits (courses numbered 601.5xx) and no more than 3 credits of short courses (1-credit special topics courses) can be counted toward this requirement. However, B.S. students doing the Senior Honors Thesis (EN.601.519-EN.601.520) may use an additional three credits of independent work toward their CS requirements, for a total of 9 credits.

No courses with grades below C- or with S/U grades can be used to fulfill this requirement unless they are not offered for a grade. At most 4 S/U credits may be applied towards this requirement.

** Up to 6 of the 42 required credits may be from an approved list of relevant courses in other departments, which includes some courses cross-listed in CS. These courses may only count as "CS other" credits, not upper level CS credits (regardless of course level in the other department). See department website for the list.

**** See the Distribution tab in the Requirements for a Bachelor's Degree (http://e-catalog.jhu.edu/undergrad-students/academic-policies/requirements-for-a-bachelors-degree) section for two exceptions to the rule that each H/S distribution course be at least 3 credits.

Mathematics (20 credits)
The following courses must be included:

<table>
<thead>
<tr>
<th>Course Code</th>
<th>Course Title</th>
<th>Credits</th>
</tr>
</thead>
<tbody>
<tr>
<td>AS.110.108</td>
<td>Calculus I</td>
<td>4</td>
</tr>
</tbody>
</table>
AS.110.109  Calculus II (For Physical Sciences and Engineering)  4

EN.553.171  Discrete Mathematics  4

The remaining courses may be chosen from Mathematics (110.xxx) or Applied Math and Statistics (553.xxx). At least one course must be 200-level or above. Strongly recommended: Linear Algebra, Probability & Statistics. Note that at least five courses will be needed to fulfill this requirement.

**Basic Sciences (12 credits)**

At least two semesters of physics or chemistry or a combination of both, with the associated laboratories, must be included. The remaining courses must be chosen in accordance with the list posted on the department website, which includes most ‘N’ designated courses in the Sciences and Engineering, but not all. At most 2 credits from (S/U) intersession courses may be used to fulfill this requirement.

**Humanities/Social Sciences (18 credits)**

Six courses in the Humanities/Social Sciences must be taken, with each course at least 3 credits.** At least two 3-credit courses at the 300-level or above are required. As befits a B.A. degree, students have ample flexibility to choose courses that broaden the scope of their study, in consultation with their advisors. A subset of the courses selected to satisfy this requirement should demonstrate coherence within an area. Any course with Humanities or Social Sciences area designators may fulfill these distributional requirements.

**Foreign Language (6 credits)**

At least 6 credits in one foreign language or demonstrated proficiency at the intermediate level are required. These foreign language credits are in addition to the 18 required Humanities/Social Sciences credits.

**Writing Requirement**

All primary computer science majors pursuing a B.A. degree are required to fulfill the university's requirement of four writing intensive courses, each at least 3 credits. At least one course must be explicitly focused on writing skills in English (eg, Expository Writing, Professional Communication, Fiction & Poetry). Students must receive at least a C- grade in these courses. These courses may overlap other requirement areas.

**Electives**

Electives may be any credit bearing courses, to be chosen by the student with the guidance of their advisor as needed.

Total Credits  120

* No more than 3 independent type credits (courses numbered 601.xxx) and no more than 3 credits of short courses (1-credit special topics courses) can be counted toward this requirement. However, B.A. students doing the Senior Honors Thesis (EN.601.519-EN.601.520) may use an additional 3 credits of independent work toward their CS requirements, for a total of 6 credits.

No courses with grades below C- or with S/U grades may be used to fulfill this requirement unless they are not offered for a grade.

No more than 3 credits may be used to fulfill this requirement.

**See the Distribution tab in the Requirements for a Bachelor’s Degree (http://e-catalog.jhu.edu/undergrad-students/academic-policies/requirements-for-a-bachelors-degree) section for two exceptions to the rule that each H/S distribution course be at least 3 credits.

**Minor in Computer Science**

To satisfy the course credit requirements for a minor in computer science, a student must take a minimum of seven courses, with a total of at least 23 credits, earning at least a C- in each course. These must include four core courses, to provide the student with a foundation, and three upper-level courses (300-level and above), to allow the student to pursue an advanced area in depth. Short courses cannot be used toward the minor requirements. All courses must be taken for a grade, not S/U.

While not explicitly required, we also strongly recommend taking EN.553.171 Discrete Mathematics as preparation for several computer science courses, including EN.601.231 Automata & Computation Theory and EN.601.433 Intro Algorithms. However, Discrete Mathematics does not count towards the minor course requirements.

**Core Courses**

EN.500.112  Gateway Computing: JAVA (or equivalent)  3
EN.601.220  Intermediate Programming  4
EN.601.226  Data Structures  4
Choose one of the following:

EN.601.229  Computer System Fundamentals  3
EN.601.231  Automata & Computation Theory

**Upper-Level Courses**

Minors must take three CS courses (EN.601.xxx) at the 300-level or above. These courses should be chosen to form a cohesive minor and must be approved by the computer science minor advisor. It is strongly recommended that students choose all three courses from within one of the five course classification areas of applications, reasoning, software, systems, or theory.

Total Credits  23

* A current listing of course area designators is provided on the departmental website. These are also encoded as POS (program of study) tags in SIS. [Note that course descriptions include old area designators which may have changed in 2019.]

Some upper level courses have math pre-requisites as well (Probability/Statistics or Linear Algebra most commonly), so plan ahead.

Students whose primary major is in the Whiting School may use the same courses to satisfy the requirements of the primary major and also those of a computer science minor. Students who plan to fulfill requirements for a minor must go to the Department of Computer Science Academic Program Coordinator or the Director of Undergraduate Studies to declare the minor and be assigned a minor advisor, and inform the Office of Academic Advising by the end of their junior year.

**Short Courses**

The Department of Computer Science offers 1-credit short courses covering a variety of topics in computer science and engineering. The purpose of the short courses is to expose students to topics of current interest in the field of computer science and engineering. Short courses are taught not only by faculty and graduate students in the Department of Computer Science and visiting faculty from other universities, but by individuals from local government or industry who have demonstrable expertise in a given area and are practicing the application of computer science theory and concepts.

Students should be aware that short course offerings are likely to change from year to year, depending on instructor commitments: there is no guarantee that the same course will be available at a later time. Students
interested in getting details about a particular short course can contact
the instructor through the departmental office. No more than 3 credits of
short courses may be applied toward the computer science course credit
requirement for the B.S. or the B.A. degrees.

Double Majors with Computer Science
It is possible for students to pursue a double major program in which one
of the majors is computer science. The computer science requirements
are flexible enough to allow for combination with most majors in the
Whiting School of Engineering and the Krieger School of Arts and
Sciences. Whether computer science is your primary or secondary major,
you will be assigned a faculty advisor in the department. In order to
declare a first or second major in computer science, students should
see the Academic Program Coordinator or the Director of Undergraduate
Studies before the start of senior year. Those students must also inform
the Office of Academic Affairs of the Whiting School of Engineering and
the Registrar of their double major status. Subject to restrictions set by
the department offering a second major, students whose primary major is
in the Whiting School may use courses to satisfy both the requirements
of the student’s primary major and those of a double major.

Combined Bachelor’s/Master’s Program
As early as the beginning of their junior year, qualified students may
apply for admission to a combined bachelor’s/master’s program which
combines a B.S. or B.A. degree (in any department) with a master of
science in engineering degree in Computer Science. This program
allows students to simultaneously pursue both an undergraduate and a
graduate degree program of study. Generally, the combined B.S./M.S.E.
or B.A./M.S.E. program is accomplished in five years, although some
students take more or less time. Applicants are judged on the basis
of their performance in courses and their letters of recommendation.
Double counting of at most two courses is subject to current WSE and
departmental policies. Students may not take a 601.3xx or 601.4xx
course as an undergraduate and the corresponding 601.6xx course
for the M.S.E. degree. Combined students will have a graduate faculty
advisor in the Computer Science Department who must approve the
courses to be applied toward the master’s degree. For information on the
requirements of the M.S.E. degree, see the Graduate Programs tab on this
page, or ask in the departmental office for the document that lists those
requirements.

Graduate Programs
Every graduate student in the Department of Computer Science must
follow a program approved by a faculty advisor in the department. The
advisor assigned to a student may change, subject to the acceptance of
the new advisor.

Requirements for the M.S.E. Degree
The Master of Science in Engineering (M.S.E.) is a daytime in-person
program offered by the Department of Computer Science. Most students
complete the program in three full-time semesters. Two semesters of
residence as a full-time graduate student are required. Those interested
in part-time evening/remote study should refer to the Engineering for
Professionals Program at ep.jhu.edu.

Entering students are expected to have completed a program of study
equivalent to that required by the B.S. in computer science. Applicants
from other disciplines are expected to have coursework (or equivalent
experience) in intermediate programming (C++ and Java), data structures,
computer system fundamentals, and automata theory. Upon admission
to the Master of Science in Engineering program, a student is assigned
a graduate advisor from the Department of Computer Science who must
approve the courses to be applied to the M.S.E. degree.

The Department of Computer Science classifies its courses into five sub-
areas: Applications, Reasoning, Software, Systems and Theory. All M.S.E.
candidates must complete at least one course (3 class hours/credits each)
from each of these five areas. A current listing of courses with area
designators (http://cs.jhu.edu/courses/courseareas.pdf) is provided on
the departmental website. The areas are also encoded as POS (program
of study) tags in SIS. [Note that course descriptions include old area
designators which may have changed in 2019.] M.S.E. students must
also complete an additional three elective courses (chosen from any CS
area or from closely related departments such as Electrical and Computer
Engineering, Cognitive Science, Mathematics, or Applied Mathematics
and Statistics) approved by the advisor, for a total of eight graduate-level
courses.

In addition to the eight courses, a student must elect one of the following
options in order to fulfill the degree requirements:

- Two additional (graduate-level) courses in Computer Science,
  approved by the advisor as above.
- A research project including an approved project report that will be
  made publicly available supervised by a faculty member that has
  a primary and/or joint appointment in Computer Science or has a
  Computer Science affiliate as a co-advisor.
- An original, faculty-approved master’s thesis, submitted to the Milton
  S. Eisenhower Library through the ETD process.

By satisfying the Ph.D. qualifying course requirements and the
first qualifying project, a student will also satisfy the M.S.E. degree
requirements (unless more than two course requirements have been
satisfied using courses transferred from other institutions). Please refer
to the Ph.D. program information for details.

All M.S.E. degree candidates are encouraged to regularly attend the
department seminars.

Course Requirement Details

- All courses counted toward the M.S.E. degree requirement must be
taken at a graduate-appropriate level. In the Department of Computer
Science, this includes courses that are 600-level and above, as well as
400-level courses only for students in the combined BS/MSE program
who have not yet switched to graduate status.
- At most, two courses with grades less than B- may be counted toward
  the coursework requirements. No courses with grades less than C-
  may be counted.
- The overall grade point average of the courses counted toward the
  coursework requirements must be a 3.0 or higher (B average).
- At most, two independent study courses can be counted toward the
course requirements.
- Other than independent study courses, no courses with grades of P
  or S can be counted toward the coursework requirement. Courses
  with grades of P or S will not be included in the grade point average
calculation.
- One of the courses required for the M.S.E. degree, but only one, can
  be replaced by 3 credits from comparable short courses.
- A majority of the courses counted toward the degree must be taught
  in the Department of Computer Science.
- At most, two courses can be transferred from graduate programs
  of other institutions to be counted toward the degree requirements.
  Such transfer courses must be approved by the student’s faculty
advisors and the department. It is the obligation of the student to provide all necessary data to the Department of Computer Science regarding the course(s) for which transfer credit is being requested.

- Students in the combined BS/MSE program may transfer up to two graduate-qualified courses which also are counted toward the undergraduate degree, as well as any other graduate-qualified courses taken while an undergraduate which are not counted toward the undergraduate degree.

- At most, two courses completed in the Engineering for Professionals program can be counted towards the degree requirements with advisor approval.

- A grade of D or F can result in probation; a second D or F is cause for being dropped from the program.

- Every student must successfully pass Academic Ethics (EN.500.603).

- Any master's student engaged in research for payment or to help meet degree requirements is required to complete Responsible Conduct of Research training. Students receiving payment from NIH training grants or fellowships must take the in person course—AS.360.625 Responsible Conduct of Research. All other students can take the course online—AS.360.624 Responsible Conduct of Research (Online). Instructions for accessing and signing up for the course can be found here: http://engineering.jhu.edu/wse-research/resources-policies-forms/responsible-conduct-of-research/online-training-course-for-the-responsible-conduct-of-research/. Additional information regarding this training can be found here: http://eng.jhu.edu/wse/page/conduct-of-research-training. Students who are required to complete this training will not receive a diploma until the course has been completed.

Tuition Support
M.S.E. students are not normally eligible for tuition waivers, but will be able to work on campus up to 19.99 hours per week for hourly rates if they find campus employment (see the Office of Student Employment for open positions). There are also course assistant positions available for qualified students who are seeking financial support. Those interested must apply at the start of each semester for specific courses in need.

Requirements for the Ph.D. Degree
The goal of the Doctor of Philosophy (Ph.D.) program in the Department of Computer Science is to prepare first-rate scholars in computer science. Successful graduates may assume significant positions in academia, research institutes, industry, or government laboratories.

Applications for admission to the Ph.D. program in Computer Science are reviewed by a faculty committee. Although the specific criteria are not rigid, all students admitted will exhibit exceptional intellectual achievements and promise. Applicants must submit letters of recommendation, GRE scores, and (for international applicants) TOEFL/IELTS scores. Visit grad.jhu.edu for more information on the application process.

In keeping with Hopkins' traditions, program requirements are flexible, as described below. For more detailed policies regarding the PhD program, please visit the Advising Manual (http://www.cs.jhu.edu/graduate-studies/phd-requirements) on our departmental website.

University Residency
Two consecutive semesters of residence as a full-time graduate student are required.

Seminar Attendance
All Ph.D. degree candidates are required to enroll and maintain satisfactory attendance in Computer Science Seminar EN.601.801 each semester for the duration of their enrollment in the program. Although seminar attendance is required, the seminar may not be counted toward the qualifying course requirement.

Responsible Conduct of Research and Academic Ethics
All doctoral students are required to take AS.360.625 Responsible Conduct of Research. Students are expected to complete the course by the end of the first year but have until the start of the fourth semester to meet this requirement. Failure to do so may result in a loss of funding. Additional information regarding this requirement can be found here: http://eng.jhu.edu/wse/page/conduct-of-research-training. In addition, all doctoral students must complete the course Academic Ethics (EN.500.603).

Qualifying Course Requirements
The Department of Computer Science classifies its courses into five sub-areas: Applications, Reasoning, Software, Systems and Theory. All Ph.D. candidates must complete at least one course (3 class hours/credits each) from each of these five areas. A current listing of courses with area designators (http://cs.jhu.edu/courses/courseareas.pdf) is provided on the departmental website. The areas are also encoded as POS (program of study) tags in SIS. [Note that course descriptions include old area designators which may have changed in 2019.] Ph.D. students must also complete an additional three elective graduate courses (chosen from any CS area or from closely related departments such as Electrical and Computer Engineering, Cognitive Science, Mathematics, or Applied Mathematics and Statistics) for a total of eight courses. Computer Science graduate students may count 600-level and above graduate courses. The coursework program must be approved by the student's faculty advisor. The overall grade point average for these eight courses must be at least equivalent to a B+. No course with a grade of less than C- may be counted toward this Ph.D. qualifying course requirement. Other than independent study courses, no courses with grades of P or S can be counted toward the coursework requirement. Courses with grades of P or S will not be included in the grade point average calculation. One of the courses required for the degree, but only one, may be replaced by 3 credits from comparable short courses. With approval of the student's faculty advisor, up to two courses can be transferred from graduate programs of other institutions; more than two such courses can be transferred with approval of the department. It is the obligation of the student to provide all necessary data to the Department of Computer Science regarding the course(s) for which transfer credit is being requested. Students are expected to complete the course requirements by the end of their second year as a Ph.D. candidate.

Qualifying Project Requirements
A Ph.D. student must complete two projects, each under the supervision and with the written agreement of a different faculty member in the Department of Computer Science. Upon conclusion of each project, the student must write a "Project Report" describing the project in detail. This report will be a public document and will be kept on file in the department office. The supervising faculty member must approve the project report. Students are expected to complete the qualifying projects by the end of their third year as a Ph.D. candidate.

Upon completion of the Ph.D. qualifying course requirements and the first qualifying project, students are ordinarily eligible to receive a master of science in engineering degree. The degree will be awarded upon student request.
Graduate Board Oral Examination (GBO)
This examination is a university requirement, ideally taken in the student's third year. The oral exam is administered by a panel consisting of the research sponsor, two faculty members from the Department of Computer Science, and two from outside the department. The exam seeks to establish the student's readiness to conduct original research in the area of his or her "Preliminary Research Proposal," which should be distributed to the examiners in advance and presented by the student at the start of the exam.

Part-Time Ph.D.
Two consecutive semesters of residence as a full-time graduate student are required by the university. Attempting to obtain a Ph.D. is a major commitment and involves close coordination with a faculty advisor in the department. Part-time students must be able to establish and maintain these close links, therefore part-time study is by advanced permission only.

Departmental Seminar
Ph.D. students must give an official departmental seminar on their research area. This is to be done after the GBO and prior to the dissertation defense, or as part of the dissertation defense.

Dissertation and Defense
Ph.D. students must write a dissertation consisting of original research in their chosen area. They must deliver a public presentation of the dissertation before a dissertation committee consisting of the faculty advisor, a second faculty member in the Department of Computer Science (who must have a primary tenure-track appointment in the Department if the advisor does not), and one or more other members with Ph.D. degrees. In conformity with University requirements, the members of the dissertation committee must submit a referee's letter to the Graduate Board recommending that the dissertation be accepted. Completed dissertations will be formatted and submitted to the Milton S. Eisenhower Library for electronic publication (http://guides.library.jhu.edu/etd).

Teaching Requirement
All Ph.D. students are required to serve as a Teaching Assistant at least one semester during their program of study. As part of the requirement, the supervising course instructor must give the TA an opportunity to be in front of a group of students at least once during the course. Students are required to sign-up for the course EN.601.807 Teaching Practicum during the semester in which the requirement is being fulfilled, and at the end of the semester his/her performance will be evaluated by the course instructor.

Student Progress Review
Ph.D. students are reviewed annually by their advisor(s) and the department, and notified in writing as to their standing in the program. Students deemed to not be making satisfactory progress may be placed on probation.

Financial Aid
All full-time CS PhD students in good academic standing are fully-funded. Additional financial aid may be available for candidates of high promise. Fellowships provide a student with a stipend plus tuition. Research assistantships are available on sponsored research projects directed by members of the faculty. Teaching assistantships normally consist of tuition plus a stipend commensurate with the teaching or grading duties assigned. Students determined to have significant deficiency in spoken English may be required to take one or more semesters of English as a Second Language in order to qualify for employment as a teaching or research assistant.

For current faculty and contact information go to http://cs.jhu.edu/faculty/

Faculty Chair
Randal Burns
Professor: storage systems, high performance computing, and data-intensive scientific computing.

Professors
Yair Amir
Distributed systems, resilient clouds and critical infrastructure, overlay networks, distributed algorithms.

Jason M. Eisner
Computational linguistics (syntax and phonology), natural language processing, statistical machine learning, programming language design.

Gregory D. Hager
Mandell Bellmore Professor; Director, Malone Center for Engineering in Healthcare: vision, robotics, human-machine systems, computer-integrated medicine.

Philipp Koehn
Statistical machine translation.

S. Rao Kosaraju
Edward J. Schaefer Professor in Engineering: design of algorithms, pattern matching, and derandomization.

Aviel Rubin
Technical Director, Information Security Institute: system and networking security, computer privacy, applied cryptography.

Steven Salzberg
Bloomberg Distinguished Professor: genomics, bioinformatics, genome assembly, gene finding, sequence analysis algorithms.

Scott F. Smith
Programming languages, type systems, security in language design, component programming languages.

Alexander Szalay
Bloomberg Distinguished Professor: data intensive computing, theoretical astrophysics, galaxy formation.

Russell H. Taylor
John C. Malone Professor; Director, LCSR & CISST ERC: medical robotics, computer-integrated interventional medicine, medical image analysis, human-machine robotic systems.

David Yarowsky
Natural language and speech processing, information retrieval, machine translation, and machine learning.

Alan Yuille
Bloomberg Distinguished Professor: computational models of vision, mathematical models of cognition, artificial intelligence, neural networks.

Associate Professors
Vladimir Braverman
Algorithms, massive data sets, data streams, and database systems.
Mark Dredze
John C. Malone Assistant Professor: machine learning, natural language processing, health informatics.

Matthew Green
Applied cryptography, cryptographic protocol design, analysis of practical security systems, privacy-preserving storage and identification technologies.

Michael Kazhdan
Computer graphics, surface reconstruction, image and geometry processing.

Michael Schatz
Bloomberg Distinguished Associate Professor: novel algorithms for comparative genomics, human genetics, and personalized medicine.

Assistant Professors
Raman Arora
Machine learning, statistical signal processing, stochastic approximation algorithms, applications to speech and language processing.

Yinzhi Cao
Web security, network security, mobile security.

Michael Dinitz
Theoretical computer science, approximation algorithms, applications to networks and distributed computing.

Soudeh Ghorbani
Networked systems, high-performance infrastructure, cloud applications, system design.

Chien-Ming Huang
John C. Malone Assistant Professor: Human robot interaction, human computer interaction, artificial intelligence, robotics.

Ryan P. Huang
Operating systems, distributed systems, software engineering, programming languages.

Abhishek Jain
Cryptography, security, theoretical computer science.

Xin Jin
Computer networking, distributed systems, software-defined networking, cloud computing, big data.

Benjamin Langmead
Computational genomics, sequence alignment, text indexing, parallel and high performance computing.

Xin Li
Theory of computation, randomness, complexity theory, distributed computing, cryptography.

Suchi Saria
John C. Malone Assistant Professor: machine learning, computational medicine, health informatics, and applications of machine learning in natural language processing, activity recognition and human-machine systems.

Ilya Shpitser
John C. Malone Assistant Professor: causal inference and missing data, graphical models, longitudinal data analysis in medicine and epidemiology.

Benjamin Van Durme
Artificial intelligence, natural language processing (computational semantics), and streaming algorithms.

Research Professors
Susan Hohenberger-Waters
Cryptography, computer security, algorithms, and complexity theory.

Peter Kazanzides
Medical robots, computer-assisted surgery, real-time systems.

Nassir Navab
Augmented reality, vision, medical image computing and computer assisted interventions.

Associate Research Professors
Philippe Burlina
Computer vision, visual analysis and communications, multi-modality image exploitation, enterprise software systems for content and e-process management.

James Mayfield
Information retrieval, cross-language retrieval, information extraction, natural language processing.

Assistant Research Professors
Kevin Duh
Natural language processing, machine learning.

Will Gray Roncal
Computer vision, image analysis, big data analytics.

Simon Leonard
Visual servoing, hand-eye coordination.

Christine Piatko
Computational geometry, information visualization, information retrieval.

Austin Reiter
Application of computer vision to robotics, machine vision, 3D reconstruction, image registration, visual recognition.

Joao Sedoc
Natural language understanding, conversational agents, hierarchical models, deep learning spectral clustering.

Wei Shen
Computer vision, deep learning, biomedical image analysis.

Mathias Unberath
Imaging systems, machine learning, augmented reality.

I-Jeng Wang
Wireless networking, Bayesian networks, probabilistic models.

Associate Teaching Professors
David Hovemeyer
Computer science education, static analysis, systems.

Sara Miner More
Foundations of computing, computer science education, cryptography, information flow.
Joanne Selinski
Director of Undergraduate Studies: computer science education, user interface design, software engineering.

**Lecturers**

M. Ali Darvish Darab
Software engineering, software testing, computer science education.

Ali Madooei
Teaching, computing education research, active learning, blended learning.

**Associate Research Scientist**

Anton Dahbura
Executive Director, Information Security Institute: Information security, fault-tolerant computing, distributed systems, testing.

**Assistant Research Scientist**

Matt Post
Machine translation, syntax, parsing and language modeling.

**Part-time Faculty**

Ayo Akinyele
Adjunct Associate Research Engineer: applied cryptography, crypto automation, computer security.

Harold Lehmann
Lecturer; Professor (Health Sciences Informatics): medical informatics.

Timothy Leschke
Lecturer: computer ethics.

Seth Nielson
Adjunct Associate Research Scientist: network security.

Lanier Watkins
Adjunct Associate Research Scientist: critical infrastructure security, network security.

**Visiting Faculty**

Mitra Basu
Visiting Professor: computational biology, pattern recognition, neural networks, artificial intelligence.

**Adjunct Faculty**

Gabor Fichtinger
Adjunct Professor: applied surgical robotics, surgical CAD/CAM systems, percutaneous therapies, stereotactic radiosurgery.

**Joint Appointments**

Amitabh Basu
Assistant Professor (Applied Mathematics and Statistics): optimization, discrete and combinatorial geometry, operations research, convex analysis.

Alexis Battle
Assistant Professor (Biomedical Engineering): genetics of complex traits, graphical models, transfer learning, structured regularization methods.

Muyinatu A. Lediju Bell
Assistant Professor (Electrical and Computer Engineering): ultrasound imaging, photoacoustic imaging, image quality improvements, advanced beamforming methods, light delivery systems, medical robotics, image-guided surgery, technology development, medical device design, clinical translation.

Emad Docter
Assistant Professor (Radiology-Medical Imaging Physics): image-guided intervention, ultrasound imaging, elasticity, and thermal imaging.

Tamas Budavari
Assistant Professor (Applied Mathematics and Statistics): computational statistics, bayesian inference, low-dimensional embeddings, streaming and randomized algorithms.

Gregory Chirikjian
Professor (Mechanical Engineering): robotics, kinematics, dynamics, control, motion planning.

Noah Cowan
Professor (Mechanical Engineering): sensor-based control of locomotion and manipulation, and biologically inspired robotics.

Ralph Etienne-Cummings
Professor (Electrical and Computer Engineering): mixed-signal VLSI, computational sensors, robotics, neuromorphic engineering.

James Fill
Professor (Applied Mathematics and Statistics): probability, stochastic processes, random structures, and algorithms.

Liliana Florea
Assistant Professor (McKusick-Nathans Institute for Genetic Medicine): application of computation techniques towards modeling and solving problems in biology and genetic medicine.

Rachel Karchin
Associate Professor (Biomedical Engineering): computational molecular biology, bioinformatics, genetic variation.

Sanjeev Khudanpur
Associate Professor (Electrical and Computer Engineering): information theory, statistical language modeling for speech recognition and machine translation.

Tal Linzen
Assistant Professor (Cognitive Science): computational psycholinguistics.

Enrique Mallada
Assistant Professor (Electrical and Computer Engineering): networked dynamics, distributed systems.

Michael I Miller
Professor (Biomedical Engineering): image understanding, computer vision, medical imaging, computational anatomy.

Vishal Patel
Assistant Professor (Electrical & Computer Engineering): signal and image processing, computer vision, machine learning.

Mihaela Pertea
Assistant Professor (Medicine): computational tools for RNA-seq analysis, gene finding, splice site prediction, sequence motif finding.

Carey Priebe
Professor (Applied Mathematics and Statistics): computational statistics, kernel and mixture estimates, statistical pattern recognition, and statistical image analysis.

Jerry L. Prince
Courses

EN.601.104. Computer Ethics. 1.0 Credit.
Students will examine a variety of topics regarding policy, legal, and moral issues related to the computer science profession itself and to the proliferation of computers in all aspects of society, especially in the era of the Internet. The course will cover various general issues related to ethical frameworks and apply those frameworks more specifically to the use of computers and the Internet. The topics will include privacy issues, computer crime, intellectual property law -- specifically copyright and patent issues, globalization, and ethical responsibilities for computer science professionals. Work in the course will consist of weekly assignments on one or more of the readings and a final paper on a topic chosen by the student and approved by the instructor.
Prerequisites: NA
Corequisites: NA
Instructor(s): T. Leschke
Area: Humanities
NA.

EN.601.105. M & Ms: Freshman Experience. 1.0 Credit.
This course provides freshmen computer science majors with an introduction to the field and department. A variety of faculty members will provide a mix of historical context and current topics. Classes will be interactive, enabling students to think about and explore topics in a fun way, as well as get to know their classmates. CS non-freshmen and minors may enroll by permission only. Satisfactory/Unsatisfactory only.
Prerequisites: NA
Corequisites: NA
Instructor(s): J. Selinski
Area: NA
NA.

EN.601.220. Intermediate Programming. 4.0 Credits.
This course teaches intermediate to advanced programming, using C and C++. (Prior knowledge of these languages is not expected.) We will cover low-level programming techniques, as well as object-oriented class design, and the use of class libraries. Specific topics include pointers, dynamic memory allocation, polymorphism, overloading, inheritance, templates, collections, exceptions, and others as time permits. Students are expected to learn syntax and some language specific features independently. Course work involves significant programming projects in both languages.
Prerequisites: AP Computer Science, or C+ or better in EN.600.107/EN.601.107 OR EN.580.200 OR EN.500.112 OR EN.500.113 OR EN.500.114 or instructor permission.
Corequisites: NA
Instructor(s): A. Madooei; J. Selinski
Area: Engineering
NA.

EN.601.226. Data Structures. 4.0 Credits.
This course covers the design and implementation of data structures including arrays, stacks, queues, linked lists, binary trees, heaps, balanced trees (e.g. 2-3 trees, AVL-trees) and graphs. Other topics include sorting, hashing, memory allocation, and garbage collection. Course work involves both written homework and Java programming assignments.
Prerequisites: C+ or better in EN.600.120 OR EN.601.220 OR EN.600.107 OR EN.601.107 OR EN.500.112 OR (EN.500.113 AND EN.500.132) OR (EN.500.114 AND EN.500.132) or by permission of instructor.
Corequisites: NA
Instructor(s): A. Madooei; J. Selinski
Area: Engineering, Quantitative and Mathematical Sciences
NA.

For current course information and registration go to https://sis.jhu.edu/classes/
EN.601.229. Computer System Fundamentals. 3.0 Credits.
We study the design and performance of a variety of computer systems from simple 8-bit micro-controllers through 32/64-bit RISC architectures all the way to ubiquitous x86 CISC architecture. We'll start from logic gates and digital circuits before delving into arithmetic and logic units, registers, caches, memory, stacks and procedure calls, pipelined execution, superscalar architectures, memory management units, etc. Along the way we'll study several typical instruction set architectures and review concepts such as interrupts, hardware and software exceptions, serial and other peripheral communications protocols, etc. A number of programming projects, frequently done in assembly language and using various processor simulators, round out the course.
Prerequisites: EN.600.120/EN.601.220
Corequisites: NA
Area: Engineering
NA.

EN.601.231. Automata & Computation Theory. 3.0 Credits.
This course is an introduction to the theory of computing. topics include design of finite state automata, pushdown automata, linear bounded automata, Turing machines and phrase structure grammars; correspondence between automata and grammars; computable functions, decidable and undecidable problems, P and NP problems, NP-completeness, and randomization. Students may not receive credit for EN.601.231/EN.600.271 and EN.601.631/EN.600.471 for the same degree.
Prerequisites: EN.550.171/EN.553.171 OR EN.553.172
Corequisites: NA
Area: Engineering, Quantitative and Mathematical Sciences
NA.

EN.601.255. Introduction to Video Game Design. 3.0 Credits.
A broad survey course in video game design (as opposed to mathematical game theory), covering artistic, technical, as well as sociological aspects of video games. Students will learn about the history of video games, archetypal game styles, computer graphics and programming, user interface and interaction design, graphical design, spatial and object design, character animation, basic game physics, plot and character development, as well as psychological and sociological impact of games. Students will design and implement an experimental video game in interdisciplinary teams of 3-4 students as part of a semester-long project. Section 1 requires technical skills, including at least one programming course (preferably 2 or more). Section 2 requires artistic skills, including at least one multimedia course (preferably 2 or more). Open to sophomores and above.
Prerequisites: NA
Corequisites: EN.601.256
Instructor(s): P. Froehlich
Area: Engineering
NA.

EN.601.256. Introduction to Video Game Design Lab. 1.0 Credit.
A lab course in support of 601.255: Introduction to Video Game Design covering a variety of multi-media techniques and applications from image processing, through sound design, to 3D modeling and animation. See 601.255: Introduction to Video Game Design for details about enrolling.
Prerequisites: NA
Corequisites: EN.600.255
Instructor(s): P. Froehlich
Area: NA

EN.601.290. User Interfaces and Mobile Applications. 3.0 Credits.
This course will provide students with a rich development experience, focused on the design and implementation of user interfaces and mobile applications. A brief overview of human computer interaction will provide context for designing, prototyping and evaluating user interfaces. Students will invent their own mobile applications and implement them using the Android SDK, which is JAVA based. An overview of the Android platform and available technologies will be provided, as well as XML for layouts, and general concepts for effective mobile development. Students will be expected to explore and experiment with outside resources in order to learn technical details independently. There will also be an emphasis on building teamwork skills, and on using modern development techniques and tools.
Prerequisites: EN.600.120 AND EN.600.226
Corequisites: NA
Instructor(s): Staff
Area: Engineering
NA.

EN.601.295. Developing Health IT Applications. 3.0 Credits.
This course is a project-based introduction to working on successful projects in health care. In the first half of the term, students perform reading and homework assignments designed to introduce: (1) the context of health care delivery and health IT, (2) techniques to overcome challenges to conducting health care data analyses, and (3) techniques to design meaningful applications around health care data. In the second half of the term, students work in small groups to solve a real-world problem of their choosing. Includes exercises in written and oral communication and team building. [Oral starting 2019]
Prerequisites: EN.600.120/.601.220 AND EN.600/601.226
Corequisites: NA
Instructor(s): C. Overby Taylor; I. Shpitser
Area: Engineering
NA.

EN.601.310. Software for Resilient Communities. 3.0 Credits.
This is a project-based course focusing on the design and implementation of practical software systems. Students will work in small teams to design and develop useful open-source software products that support our communities. Students will be paired with community partners and will aim to develop software that can be used after the course ends to solve real problems facing those partners today. Instructors will connect with the community partners and determine viable project areas prior to the course start. Students will meet with their community partners to analyze the challenges in their project area, agree on a concrete target project outcome, and gather requirements for their project. Based on these requirements, students will design and implement open-source software systems. [Oral]
Prerequisites: EN.600.120/EN.601.220 AND EN.600.226
Corequisites: NA
Instructor(s): A. Babay; Y. Amir
Area: Engineering
NA.
EN.600.315. Databases. 3.0 Credits.
Introduction to database management systems and database design, focusing on the relational and object-oriented data models, query languages and query optimization, transaction processing, parallel and distributed databases, recovery and security issues, commercial systems and case studies, heterogeneous and multimedia databases, and data mining. [Systems] (www.cs.jhu.edu/~yarowsky/cs415.html)
Prerequisites: EN.600.226/EN.601.226; Students may receive credit for only one of EN.600.315, EN.600.415, EN.601.318, EN.601.415, EN.601.615.
Corequisites: NA
Instructor(s): D. Yarowsky
Area: Engineering
NA.

EN.601.317. Distributed Systems. 3.0 Credits.
This course teaches how to design and implement protocols that enable processes to exchange information, cooperate, and coordinate efficiently in a consistent manner over a computer network. Topics include communication protocols, group communication, distributed databases, distributed operating systems, and security. [Systems]
Prerequisites: Students may receive credit for EN.601.317 or EN.601.417 but not both.
Corequisites: Students may receive credit for 600.337 or 600.437, but not both.
Corequisites: NA
Instructor(s): Y. Amir
Area: Engineering
NA.

EN.601.318. Operating Systems. 3.0 Credits.
This course covers fundamental topics related to operating systems theory and practice. Topics include processor management, storage management, concurrency control, multi-programming and processing, device drivers, operating system components (e.g., file system, kernel), modeling and performance measurement, protection and security, and recent innovations in operating system structure. Course work includes the implementation of operating systems techniques and routines, and critical parts of a small but functional operating system.
Prerequisites: EN.600.120/EN.601.220 AND EN.600.226/EN.601.226 AND EN.600.233/EN.601.229; Students may receive credit for only one of EN.600.318, EN.600.418, EN.601.318, EN.601.418, EN.601.618.
Corequisites: NA
Instructor(s): P. Huang
Area: Engineering
NA.

EN.601.320. Parallel Programming. 3.0 Credits.
This course prepares the programmer to tackle the massive data sets and huge problem size of modern scientific and enterprise computing. Google and IBM have commented that undergraduate CS majors are unable to "break the single server mindset" (http://www.google.com/intl/en/press/pressrel/20071008_ibm_univ.html). Students taking this course will abandon the comfort of serial algorithmic thinking and learn to harness the power of cutting-edge software and hardware technologies. The issue of parallelism spans many architectural levels. Even "single server" systems must parallelize computation in order to exploit the inherent parallelism of recent multi-core processors. The course will examine different forms of parallelism in four sections. These are: (1) massive data-parallel computations with Hadoop; (2) programming compute clusters with MPI; (3) thread-level parallelism in Java; and, (4) GPGPU parallel programming with NVIDIA's Cuda. Each section will be approximately 3 weeks and each section will involve a programming project. The course is also suitable for undergraduate and graduate students from other science and engineering disciplines that have prior programming experience. [Systems] Prerequisite: EN.600.233/EN.601.229 and EN.601.226
Prerequisites: EN.600/601.226 AND EN.600.233/601.229; Students may receive credit for only one of EN.600.320, EN.600.420, EN.601.320, EN.601.420, EN.601.620.
Corequisites: NA
Instructor(s): R. Burns
Area: Engineering
NA.

EN.601.325. Declarative Methods. 3.0 Credits.
Suppose you could simply write down a description of your problem, and let the computer figure out how to solve it. What notation could you use? What strategy should the computer then use? In this survey class, you'll learn to recognize when your problem is an instance of satisfiability, constraint programming, logic programming, dynamic programming, or mathematical programming (e.g., integer linear programming). For each of these related paradigms, you'll learn to reformulate hard problems in the required notation and apply off-the-shelf software that can solve any problem in that notation -- including NP-complete problems and many of the problems you'll see in other courses and in the real world. You'll also gain some understanding of the general-purpose algorithms that power the software. [Analysis] Students can only receive credit for EN.601.325 or EN.601.425, not both.
Prerequisites: NA
Corequisites: NA
Instructor(s): J. Eisner
Area: Engineering
NA.

EN.601.328. Compilers and Interpreters. 3.0 Credits.
Introduction to compiler design, including lexical analysis, parsing, syntax-directed translation, symbol tables, run-time environments, and code generation and optimization. Students are required to write a compiler as a course project. [Systems] Co-listed with EN.601.428
Prerequisites: EN.600.120 AND EN.600.226; Students may receive credit for only one of EN.600.328, EN.600.428, EN.601.328, EN.601.428, EN.601.628.
Corequisites: NA
Instructor(s): P. Froehlich
Area: Engineering
NA.
EN.601.340. Web Security. 3.0 Credits.
This course begins with reviewing basic knowledge of the World Wide Web, and then exploring the central defense concepts behind Web security, such as same-origin policy, cross-origin resource sharing, and browser sandboxing. It will cover the most popular Web vulnerabilities, such as cross-site scripting (XSS) and SQL injection, as well as how to attack and penetrate software with such vulnerabilities. Students will learn how to detect, respond, and recover from security incidents. Newly proposed research techniques will also be discussed. [Systems]
Prerequisites: (EN.600.226 OR EN.601.226) AND (EN.600.233 OR EN.601.229)
Corequisites: NA
Area: Engineering

EN.601.350. Introduction to Genomic Research. 3.0 Credits.
This course will use a project-based approach to introduce undergraduates to research in computational biology and genomics. During the semester, students will take a series of large data sets, all derived from recent research, and learn all the computational steps required to convert raw data into a polished analysis. Data challenges might include the DNA sequences from a bacterial genome project, the RNA sequences from an experiment to measure gene expression, the DNA from a human microbiome sequencing experiment, and others. Topics may vary from year to year. In addition to computational data analysis, students will learn to do critical reading of the scientific literature by reading high-profile research papers that generated groundbreaking or controversial results. [Applications] Recommended Course Background: Knowledge of the Unix operating system and programming expertise in a language such as Perl or Python.
Prerequisites: NA
Corequisites: NA
Instructor(s): S. Salzberg
Area: Engineering
NA.

EN.601.355. Video Game Design Project. 3.0 Credits.
An intensive capstone design project experience in video game development. Students will work in groups of 4-8 on developing a complete video game of publishable quality. Teams will (hopefully) include programmers, visual artists, composers, and writers. Students will be mentored by experts from industry and academia. Aside from the project itself, project management and communication skills will be emphasized. Enrollment is limited to ensure parity between the various disciplines. [General] May involve travel to MICA. Junior or senior standing recommended.
Prerequisites: EN.600.255
Corequisites: NA
Instructor(s): P. Froehlich
Area: Engineering
NA.

EN.601.356. Seminar: Computer Integrated Surgery II. 1.0 Credit.
Students may receive credit for EN.601.456 or EN.601.356, but not both. Lecture only version of EN.601.456 (no project). Recommended Course Background: EN.601.455 or instructor permission required.
Prerequisites: NA
Corequisites: NA
Instructor(s): R. Taylor
Area: Engineering
NA.

EN.601.365. Knowledge Discovery from Text. 3.0 Credits.
The world is full of text: webpages, emails, newspaper articles, tweets, medical records, and so on. The purpose of text is for people to convey knowledge to other people. This course focuses on how computers analyze large, potentially streaming, text collections to automatically discover knowledge on their own (and to help people better find it themselves). Lectures and assignments will cover relevant topics in automatic classification (applied machine learning), linguistics, high-performance computing, and systems engineering, working with software systems for automatic question answering, populating knowledge bases, and aggregate analysis of social media such as Twitter. [Applications]
Prerequisites: EN.600.120 AND EN.600.226
Corequisites: NA
Instructor(s): B. Van Durme; T. Lippincott
Area: Engineering
NA.

EN.601.382. Deep Learning Lab. 1.0 Credit.
This course is an optional hands-on lab supplement for a few courses in the curriculum. It will provide tutorial support and practical experience for developing deep ML systems using PyTorch and TensorFlow, and may provide exposure to some other frameworks. It will also go into detail on practical methods for scalable learning on large data sets, and other more practical issues in setting up deep learning systems.
Prerequisites: NA
Corequisites: Students must also register for EN.601.482 OR EN.601.682 OR EN.601.765.
Instructor(s): J. Wu
Area: Engineering
NA.

EN.601.402. Digital Health and Biomedical Informatics. 1.0 Credit.
Advances in technology are driving a change in medicine, from personalized medicine to population health. Computers and information technology will be critical to this transition. We shall discuss some of the coming changes in terms of computer technology, including computer-based patient records, clinical practice guidelines, and region-wide health information exchanges. We will discuss the underlying technologies driving these developments - databases and warehouses, controlled vocabularies, and decision support.
Prerequisites: NA
Corequisites: NA
Instructor(s): H. Lehmann
Area: Engineering
NA.

EN.601.411. Computer Science Innovation & Entrepreneurship II. 3.0 Credits.
This course is the second half of a two-course sequence and is a continuation of course EN.660.410.01, CS Innovation and Entrepreneurship, offered by the Center for Leadership Education (CLE). In this sequel course the student groups, directed by CS faculty, will implement the business idea which was developed in the first course and will present the implementations and business plans to an outside panel made up of practitioners, industry representatives, and venture capitalists. [General]
Prerequisites: EN.660.410
Corequisites: NA
Instructor(s): A. Dahbura; L. Aronhime
Area: Engineering
NA.
EN.601.414. Computer Networks. 4.0 Credits.
Topics covered will include application layer protocols (e.g. HTTP, FTP, SMTP), transport layer protocols (UDP, TCP), network layer protocols (e.g. IP, ICMP), link layer protocols (e.g. Ethernet) and wireless protocols (e.g. IEEE 802.11). The course will also cover routing protocols such as link state and distance vector, multicast routing, and path vector protocols (e.g. BGP). The class will examine security issues such as firewalls and denial of service attacks. We will also study DNS, NAT, Web caching and CDNs, peer to peer, and protocol tunneling. Finally, we will explore security protocols (e.g. TLS, SSH, IPsec), as well as some basic cryptography necessary to understand these. Grading will be based on hands-on programming assignments, homeworks and two exams. [Systems] Prerequisites: EN.600.226 AND EN.600.233; Students may receive credit for only one of EN.600.344, EN.600.444, EN.601.414, EN.601.614.
Corequisites: NA
Instructor(s): M. Green; X. Jin
Area: Engineering
NA.

EN.601.415. Databases. 3.0 Credits.
Similar material as EN.601.315 covered in more depth for advanced undergraduates. Introduction to database management systems and database design, focusing on the relational and object-oriented data models, query languages and query optimization, transaction processing, parallel and distributed databases, recovery and security issues, commercial systems and case studies, heterogeneous and multimedia databases, and data mining. [Systems] (www.cs.jhu.edu/~yarowsky/cs415.html) Prerequisites: EN.600.226/EN.601.226; Students may receive credit for only one of EN.600.315, EN.600.415, EN.601.315, EN.601.415, EN.601.615.
Corequisites: NA
Instructor(s): D. Yarowsky
Area: Engineering
NA.

EN.601.417. Distributed Systems. 3.0 Credits.
Graduate version of 601.317 Systems. Students may receive credit for 601.317 or 601.417 but not both. Recommended Course Background: EN.601.220, EN.601.226
Prerequisites: Students may receive credit for only one of 417/617; (EN.600.120 OR EN.601.220) AND (EN.600.226 OR EN.601.226)
Corequisites: NA
Instructor(s): Y. Amir
Area: Engineering
NA.

EN.601.418. Operating Systems. 3.0 Credits.
Similar material as EN.601.318, covered in more depth. Intended for advanced undergraduate students. This course covers fundamental topics related to operating systems theory and practice. Topics include processor management, storage management, concurrency control, multi-programming and processing, device drivers, operating system components (e.g., file system, kernel), modeling and performance measurement, protection and security, and recent innovations in operating system structure. Course work includes the implementation of operating systems techniques and routines, and critical parts of a small but functional operating system. Prerequisites: EN.600.120/EN.601.220 AND EN.600.226/EN.601.226 AND EN.600.233/EN.601.229; Students may receive credit for only one of EN.600.318, EN.600.418, EN.601.318, EN.601.418, EN.601.618.
Corequisites: NA
Instructor(s): P. Huang
Area: Engineering
NA.

EN.601.419. Cloud Computing. 3.0 Credits.
Clouds host a wide range of the applications that we rely on today. In this course, we study common cloud applications, traffic patterns that they generate, critical networking infrastructures that support them, and core networking and distributed systems concepts, algorithms, and technologies used inside clouds. We will also study how today’s application demand is influencing the network’s design, explore current practice, and how we can build future’s networked infrastructure to better enable both efficient transfer of big data and low-latency requirements of real-time applications. The format of this course will be a mix of lectures, discussions, assignments, and a project designed to help students practice and apply the theories and techniques covered in the course. [Systems] Prerequisites: EN.601.226 or permission. Students can only receive credit for one of 601.419/619. Recommended: a course in operating systems, networks or systems programming. Prerequisites: Students may earn credit for EN.601.419 or EN.601.619, but not both.; EN.601.226 (or EN.600.226)
Corequisites: NA
Instructor(s): S. Ghorbani Khaledi
Area: Engineering
NA.

EN.601.420. Parallel Programming. 3.0 Credits.
More advanced version of EN.601.320. Students may receive credit for EN.601.320 or EN.601.420, but not both. Prerequisite: EN.601.226 and EN.601.229 Computer System Fundamentals. Prerequisites: EN.600/601.226 AND EN.600.233/601.229; Students may receive credit for only one of EN.600.320, EN.600.420, EN.601.320, EN.601.420, EN.601.620.
Corequisites: NA
Instructor(s): R. Burns
Area: Engineering
NA.
EN.601.421. Object Oriented Software Engineering. 3.0 Credits.
This course covers object-oriented software construction methodologies and their application. The main component of the course is a large team project on a topic of your choosing. Course topics covered include object-oriented analysis and design, UML, design patterns, refactoring, program testing, code repositories, team programming, and code reviews. [Systems or Applications] (http://pl.cs.jhu.edu/oose/index.shtml)
Prerequisites: EN.600.120/EN.601.220 AND EN.600.226/EN.601.226; Students may receive credit for only one of EN.600.321, EN.601.421, EN.601.421, EN.601.621.
Corequisites: NA
Instructor(s): L. Facchinetti
Area: Engineering
NA.

EN.601.422. Software Testing & Debugging. 3.0 Credits.
Studies show that testing can account for over 50% of software development costs. This course presents a comprehensive study of software testing, principles, methodologies, tools, and techniques. Topics include testing principles, coverage (graph coverage, logic coverage, input space partitioning, and syntax-based coverage), unit testing, higher-order testing (integration, system-level, acceptance), testing approaches (white-box, black-box, grey-box), regression testing, debugging, delta debugging, and several specific types of functional and non-functional testing as schedule/interest permits (GUI testing, usability testing, security testing, load/performance testing, A/B testing etc.). For practical topics, state-of-the-art tools/techniques will be studied and utilized. [Systems]
Prerequisites: EN.601.290 OR EN.601.421; Students can take EN.601.422 or EN.601.622, but not both.
Corequisites: NA
Instructor(s): M. Darvish Darab
Area: Engineering
NA.

EN.601.425. Declarative Methods. 3.0 Credits.
Students can only receive credit for EN.601.325 or EN.601.425, not both. Graduate level version of EN.601.325. Recommended Course Background: EN.601.226, EN.601.231, AS.110.107/AS.110.109
Prerequisites: NA
Corequisites: NA
Instructor(s): J. Eisner
Area: Engineering
NA.

EN.601.426. Principles of Programming Languages. 3.0 Credits.
Functional, object-oriented, and other language features are studied independent of a particular programming language. Students become familiar with these features by implementing them. Most of the implementations are in the form of small language interpreters. Some type checkers and a small compiler will also be written. The total amount of code written will not be overly large, as the emphasis is on concepts. The ML programming language is the implementation language used. [Analysis] Prerequisites include EN.601.226. No Freshmen or Sophomores.
Prerequisites: EN.600.226
Corequisites: NA
Instructor(s): S. Smith
Area: Engineering, Quantitative and Mathematical Sciences
NA.

EN.601.427. Principles of Programming Languages II. 3.0 Credits.
This course is designed as a follow-on to Principles of Programming Languages. It will cover a wide array of fundamental topics in programming languages, including advanced functional programming, the theory of inductive definitions, advanced operational semantics, advanced type systems, program analysis, program verification, theorem provers and SAT solvers. [Analysis]
Prerequisites: EN.601.426
Corequisites: NA
Instructor(s): S. Smith
Area: Engineering
NA.

EN.601.428. Compilers & Interpreters. 3.0 Credits.
Prerequisites: EN.600.120 AND EN.600.226 AND EN.600.233
Corequisites: NA
Instructor(s): P. Froehlich
Area: Engineering
NA.

EN.601.430. Combinatorics & Graph Theory in Computer Science. 3.0 Credits.
This is a graduate level course studying the applications of combinatorics and graph theory in computer science. We will start with some basic combinatorial techniques such as counting and pigeon hole principle, and then move to advanced techniques such as the probabilistic method, spectral graph theory and additive combinatorics. We shall see their applications in various areas in computer science, such as proving lower bounds in computational models, randomized algorithms, coding theory and pseudorandomness. [Analysis] Recommended Course Background: probability theory and linear algebra
Prerequisites: EN.553.171 OR EN.553.172 OR EN.550.171; probability theory and linear algebra recommended; Students may receive credit for only one of 430/630
Corequisites: NA
Instructor(s): X. Li
Area: Engineering, Quantitative and Mathematical Sciences
NA.

EN.601.433. Intro Algorithms. 3.0 Credits.
This course concentrates on the design of algorithms and the rigorous analysis of their efficiency. topics include the basic definitions of algorithmic complexity (worst case, average case); basic tools such as dynamic programming, sorting, searching, and selection; advanced data structures and their applications (such as union-find); graph algorithms and searching techniques such as minimum spanning trees, depth-first search, shortest paths, design of online algorithms and competitive analysis. [Analysis]
Prerequisites: EN.600.226/EN.601.226 AND (EN.553.171/EN.553.172 OR EN.550.171); Students may receive credit for only one of EN.600.363, EN.600.463, EN.601.433, EN.601.633.
Corequisites: NA
Instructor(s): M. Dinitz
Area: Engineering, Quantitative and Mathematical Sciences
NA.
EN.601.434. Randomized and Big Data Algorithms. 3.0 Credits.
The course emphasizes algorithmic design aspects, and how randomization can be a helpful tool. The topics covered include: tail inequalities, linear programming relaxation & randomized rounding, de-randomization, existence proofs, universal hashing, markov chains, metropolis and metropolis-hastings methods, mixing by coupling and by eigenvalues, counting problems, semi-definite programming and rounding, lower bound arguments, and applications of expanders.
[Analysis] (www.cs.jhu.edu/~cs464) Recommended Course Background: Probability
Prerequisites: (( N.600.363 OR EN.600.463) OR (EN.601.433 OR EN.601.633)) AND (EN.550.310 OR EN.553.310 OR EN.553.311 OR EN.550.420 OR EN.550.620) or equivalent.;Students may receive credit for only one of EN.600.464, EN.600.664, EN.601.434, EN.601.634.
Corequisites: NA
Instructor(s): V. Braverman
Area: Engineering, Quantitative and Mathematical Sciences
NA.

EN.601.435. Approximation Algorithms. 3.0 Credits.
This course provides an introduction to approximation algorithms. Topics include vertex cover, TSP, Steiner trees, cuts, greedy approach, linear and semi-definite programming, primal-dual method, and randomization. Additional topics will be covered as time permits. There will be a final project. Students may receive credit for EN.601.435 or EN.601.635, but not both. [Analysis]
Prerequisites: EN.600.363 OR EN.601.433 OR EN.601.633 OR permission.
Corequisites: NA
Instructor(s): M. Dinitz
Area: Engineering, Quantitative and Mathematical Sciences
NA.

EN.601.436. Algorithmic Game Theory. 3.0 Credits.
This course provides an introduction to algorithmic game theory: the study of games from the perspective of algorithms and theoretical computer science. There will be a particular focus on games that arise naturally from economic interactions involving computer systems (such as economic interactions between large-scale networks, online advertising markets, etc.), but there will also be broad coverage of games and mechanisms of all sorts. Topics covered will include a) complexity of computing equilibria and algorithms for doing so, b) (in)efficiency of equilibria, and c) algorithmic mechanism design. [Analysis]
Prerequisites: EN.600.363 OR EN.600.463 OR EN.601.433 OR EN.601.633
Corequisites: NA
Instructor(s): M. Dinitz
Area: Engineering, Quantitative and Mathematical Sciences
NA.

EN.601.440. Web Security. 3.0 Credits.
This course begins with reviewing basic knowledge of the World Wide Web, and then exploring the central defense concepts behind Web security, such as same-origin policy, cross-origin resource sharing, and browser sandboxing. It will cover the most popular Web vulnerabilities, such as cross-site scripting (XSS) and SQL injection, as well as how to attack and penetrate software with such vulnerabilities. Students will learn how to detect, respond, and recover from security incidents. Newly proposed research techniques will also be discussed. [Systems]
Prerequisites: (EN.601.226 OR EN.600.226) AND (EN.601.229 OR EN.600.233);Students may receive credit for only one of 340/440/640.
Corequisites: NA
Instructor(s): Y. Cao
Area: Engineering
NA.

EN.601.441. Blockchains and Cryptocurrencies. 3.0 Credits.
This course will introduce students to cryptocurrencies and the main underlying technology of Blockchains. The course will start with the relevant background in cryptography and then proceed to cover the recent advances in the design and applications of blockchains. This course should primarily appeal to students who want to conduct research in this area or wish to build new applications on top of blockchains. It should also appeal to those who have a casual interest in this topic or are generally interested in cryptography. Students are expected to have mathematical maturity. [Analysis]
Prerequisites: Students may receive credit for only one of EN.600.451 OR EN.601.441 OR EN.601.441;EN.601.226 AND (EN.553.211 OR EN.553.310 OR EN.553.311 OR EN.560.348 OR EN.553.420)
Corequisites: NA
Instructor(s): A. Jain; M. Green
Area: Engineering
NA.

EN.601.442. Modern Cryptography. 3.0 Credits.
Modern Cryptography includes seemingly paradoxical notions such as communicating privately without a shared secret, proving things without leaking knowledge, and computing on encrypted data. In this challenging but rewarding course we will start from the basics of private and public key cryptography and go all the way up to advanced notions such as zero-knowledge proofs, functional encryption and program obfuscation. The class will focus on rigorous proofs and require mathematical maturity. [Analysis]
Prerequisites: (EN.601.231 OR EN.600.271 OR EN.600.471) AND (EN.550.310 OR EN.553.310 OR EN.553.311 OR EN.550.331 OR EN.550.420 OR EN.553.420);Students may receive credit for only one of EN.600.442, EN.601.442, EN.601.642.
Corequisites: NA
Instructor(s): A. Jain
Area: Engineering, Quantitative and Mathematical Sciences
NA.
EN.601.443. Security & Privacy in Computing. 3.0 Credits.
Lecture topics will include computer security, network security, basic cryptography, system design methodology, and privacy. There will be a heavy work load, including written homework, programming assignments, exams and a comprehensive final. The class will also include a semester-long project that will be done in teams and will include a presentation by each group to the class. [Applications] Recommended Course Background: A basic course in operating systems and networking, or permission of instructor.
Prerequisites: (EN.600.318/EN.601.318 OR EN.600.418/EN.601.418) OR (EN.600.344 OR EN.600.444/EN.601.414) AND (EN.600.233 OR EN.601.229); Students may receive credit for only one of EN.600.443, EN.601.443, EN.601.643.
Corequisites: NA
Instructor(s): A. Rubin
Area: Engineering
NA.

EN.601.444. Network Security. 3.0 Credits.
This course focuses on communication security in computer systems and networks. The course is intended to provide students with an introduction to the field of network security. The course covers network security services such as authentication and access control, integrity and confidentiality of data, firewalls and related technologies, Web security and privacy. Course work involves implementing various security techniques. A course project is required. [Systems]
Prerequisites: EN.600.120 AND EN.600.226 AND (EN.600.344 OR EN.600.444) or permission; Students may receive credit for only one of EN.600.424, EN.650.424, EN.601.444, EN.601.644.
Corequisites: NA
Instructor(s): S. Nielsen
Area: Engineering
NA.

EN.601.445. Practical Cryptographic Systems. 3.0 Credits.
This semester-long course will teach systems and cryptographic design principles by example: by studying and identifying flaws in widely-deployed cryptographic products and protocols. Our focus will be on the techniques used in practical security systems, the mistakes that lead to failure, and the approaches that might have avoided the problem. We will place a particular emphasis on the techniques of provable security and the feasibility of reverse-engineering undocumented cryptographic systems. [Systems]
Prerequisites: EN.600.226 AND EN.600.233; Students may receive credit for only one of EN.600.454, EN.601.445, EN.601.645.
Corequisites: NA
Instructor(s): M. Green
Area: Engineering
NA.

EN.601.447. Computational Genomics: Sequences. 3.0 Credits.
Your genome is the blueprint for the molecules in your body. It’s also a string of letters (A, C, G and T) about 3 billion letters long. How does this string give rise to you? Your heart, your brain, your health? This, broadly speaking, is what genomics research is about. This course will familiarize you with a breadth of topics from the field of computational genomics. The emphasis is on current research problems, real-world genomics data, and efficient software implementations for analyzing data. Topics will include: string matching, sequence alignment and indexing, assembly, and sequence models. Course will involve significant programming projects. [Applications]
Prerequisites: EN.600.120/EN.601.220 AND EN.600.226/EN.601.226; Students may receive credit for only one of EN.600.439, EN.600.639, EN.601.447, EN.601.647.
Corequisites: NA
Instructor(s): B. Langmead, B. Solomon
Area: Engineering
NA.

EN.601.448. Computational Genomics: Data Analysis. 3.0 Credits.
Genomic data has the potential to reveal causes of disease, novel drug targets, and relationships among genes and pathways in our cells. However, identifying meaningful patterns from high-dimensional genomic data has required development of new computational tools. This course will cover current approaches in computational analysis of genomic data with a focus on statistical methods and machine learning. Topics will include disease association, prediction tasks, clustering and dimensionality reduction, data integration, and network reconstruction. There will be some programming and a project component. [Applications]
Prerequisites: EN.601.226 or other programming experience, probability and statistics, linear algebra or calculus.
Prerequisites: Students may receive credit for only one of EN.600.438, EN.600.638, EN.601.448, EN.601.648.
Corequisites: NA
Instructor(s): A. Battle
Area: Engineering
NA.

EN.601.452. Computational Biomedical Research. 3.0 Credits.
[Co-listed with AS.020.415] This course for advanced undergraduates includes classroom instruction in interdisciplinary research approaches and lab work on an independent research project in the lab of a Bloomberg Distinguished Professor and other distinguished faculty. Lectures will focus on cross-cutting techniques such as data visualization, statistical inference, and scientific computing. In addition to two 50-minute classes per week, students will commit to working approximately 3 hours per week in the lab of one of the professors. The student and professor will work together to schedule the research project. Students will present their work at a symposium at the end of the semester.
Prerequisites: NA
Corequisites: NA
Instructor(s): M. Schatz
Area: Engineering
NA.
EN.601.454. Augmented Reality. 3.0 Credits.
Same as EN.601.654, for undergraduate students. This course introduces students to the field of Augmented Reality. It reviews its basic definitions, principles and applications. It then focuses on Medical Augmented Reality and its particular requirements. The course also discusses the main issues of calibration, tracking, multi-modal registration, advance visualization and display technologies. Homework in this course will relate to the mathematical methods used for calibration, tracking and visualization in medical augmented reality. [Applications]
Prerequisites: EN.600.120 AND EN.600.226 AND (AS.110.201 OR EN.550.291); Students may receive credit for only one of EN.600.484, EN.600.684, EN.601.454, EN.601.654.
Corequisites: NA
Instructor(s): R. Taylor
Area: Engineering

EN.601.455. Computer Integrated Surgery I. 4.0 Credits.
This course focuses on computer-based techniques, systems, and applications exploiting quantitative information from medical images and sensors to assist clinicians in all phases of treatment from diagnosis to preoperative planning, execution, and follow-up. It emphasizes the relationship between problem definition, computer-based technology, and clinical application and includes a number of guest lectures given by surgeons and other experts on requirements and opportunities in particular clinical areas. Recommended Course Background: EN.601.220, EN.601.457, EN.601.461, image processing.
Prerequisites: EN.600.226/EN.601.226 AND (AS.110.201 OR AS.110.212) or permission of the instructor; Students may receive credit for only one of EN.600.445, EN.600.645, EN.601.455, EN.601.655.
Corequisites: NA
Instructor(s): R. Taylor
Area: Engineering

EN.601.456. Computer Integrated Surgery II. 3.0 Credits.
This weekly lecture/seminar course addresses similar material to EN.601.445, but covers selected topics in greater depth. In addition to material covered in lectures/seminars by the instructor and other faculty, students are expected to read and provide critical analysis/presentations of selected papers in recitation sessions. Students taking this course are required to undertake and report on a significant term project under the supervision of the instructor and clinical end users. Typically, this project is an extension of the term project from EN.601.445, although it does not have to be. Grades are based both on the project and on classroom recitations. Students wishing to attend the weekly lectures as a 1-credit seminar should sign up for EN.600.452. [Applications]
Prerequisites: EN.600.445 or EN.600.645 or permission; Students may receive credit for only one of EN.600.446, EN.600.646, EN.601.456, EN.601.656.
Corequisites: NA
Instructor(s): R. Taylor
Area: Engineering

EN.601.457. Computer Graphics. 3.0 Credits.
This course introduces computer graphics techniques and applications, including image processing, rendering, modeling and animation. [Applications]
Prerequisites: EN.600.120/601.220 AND EN.600.226/601.226, and linear algebra or permission of instructor; Students may receive credit for only one of EN.600.357, EN.600.457, EN.601.457, EN.601.657.
Corequisites: NA
Instructor(s): M. Kazhdan
Area: Engineering, Quantitative and Mathematical Sciences

EN.601.459. Computational Geometry. 3.0 Credits.
This course will provide an introduction to computational geometry. It will cover a number of topics in two- and three-dimensions, including polygon triangulations and partitions, convex hulls, Delaunay and Voronoi diagrams, arrangements, and spatial queries. Time-permitting, we will also look at k-d trees, general BSP-trees, and quadtrees. [Analysis]
Recommended Course Background: EN.601.220 AND EN.601.226 AND (EN.600.363 OR EN.601.433).
Prerequisites: Students may receive credit for EN.600.459 or EN.600.659, but not both.
Corequisites: NA
Instructor(s): M. Kazhdan
Area: Engineering, Quantitative and Mathematical Sciences

EN.601.461. Computer Vision. 3.0 Credits.
This course gives an overview of fundamental methods in computer vision from a computational perspective. Methods studied include: camera systems and their modeling, computation of 3-D geometry from binocular stereo, motion, and photometric stereo; and object recognition. Edge detection and color perception are covered as well. Elements of machine vision and biological vision are also included. [Applications]
Prerequisites: intro programming, linear algebra, and prob/stat.
Prerequisites: Probability Statistics, Linear Algebra, and Computing
- (EN.550.310 OR EN.550.311 OR (EN.550.420 AND EN.550.430) OR EN.550.348) AND (AS.110.201 OR AS.110.212 OR EN.553.291) AND (EN.500.112 OR EN.500.113 OR (EN.500.114 OR EN.600.120/ EN.601.120) OR EN.600.170/EN.600.170 OR EN.580.200[C] OR AS.250.205); Students may receive credit for only one of EN.600.361, EN.600.461, EN.600.661, EN.601.461, EN.601.661.
Corequisites: NA
Instructor(s): G. Hager
Area: Engineering, Quantitative and Mathematical Sciences

EN.601.463. Algorithms for Sensor-Based Robotics. 3.0 Credits.
This course surveys the development of robotic systems for navigating in an environment from an algorithmic perspective. It will cover basic kinematics, configuration space concepts, motion planning, and localization and mapping. It will describe these concepts in the context of the ROS software system, and will present examples relevant to mobile platforms, manipulation, robotics surgery, and human-machine systems. [Analysis]
Prerequisites: EN.600.226 OR EN.601.226 and Linear Algebra and Probability; Students may receive credit for only one of EN.600.336, EN.600.436, EN.600.636, EN.601.463, EN.601.663.
Corequisites: NA
Instructor(s): S. Leonard
Area: Engineering

Area: Engineering
Instructor(s): N. Navab
Corequisites: NA

Area: Engineering, Quantitative and Mathematical Sciences
Instructor(s): G. Hager
Corequisites: NA

Area: Engineering, Quantitative and Mathematical Sciences
Instructor(s): M. Kazhdan
Corequisites: NA

Area: Engineering, Quantitative and Mathematical Sciences
Instructor(s): S. Leonard
Corequisites: NA
EN.601.467. Introduction to Human Language Technology. 3.0 Credits.
This course gives an overview of basic foundations and applications of human language technology, such as: morphological, syntactic, semantic, and pragmatic processing; machine learning; signal processing; speech recognition; speech synthesis; information retrieval; text classification; topic modelling; information extraction; knowledge representation; machine translation; dialog systems; etc. [Applications] Pre-req: EN.601.226 Data Structures; knowledge of Python recommended.
Prerequisites: EN.601.226 OR EN.600.226
Corequisites: NA
Instructor(s): P. Koehn
Area: Engineering
NA.

EN.601.468. Machine Translation. 3.0 Credits.
Google translate can instantly translate between any pair of over fifty human languages (for instance, from French to English). How does it do that? Why does it make the errors that it does? And how can you build something better? Modern translation systems learn to translate by reading millions of words of already translated text, and this course will show you how they work. The course covers a diverse set of fundamental building blocks from linguistics, machine learning, algorithms, data structures, and formal language theory, along with their application to a real and difficult problem in artificial intelligence.
Prerequisites: EN.600.226 and prob/stat.; Students may receive credit for only one of EN.600.468, EN.601.468, EN.601.668.
Corequisites: NA
Instructor(s): P. Koehn
Area: Engineering
NA.

EN.601.475. Machine Learning. 3.0 Credits.
Machine learning is subfield of computer science and artificial intelligence, whose goal is to develop computational systems, methods, and algorithms that can learn from data to improve their performance. This course introduces the foundational concepts of modern Machine Learning, including core principles, popular algorithms and modeling platforms. This will include both supervised learning, which includes popular algorithms like SVMs, logistic regression, boosting and deep learning, as well as unsupervised learning frameworks, which include Expectation Maximization and graphical models. Homework assignments include a heavy programming components, requiring students to implement several machine learning algorithms in a common learning framework. Additionally, analytical homework questions will explore various machine learning concepts, building on the pre-requisites that include probability, linear algebra, multi-variate calculus and basic optimization. Students in the course will develop a learning system for a final project. [Applications or Analysis] Required course background: multivariable calculus, probability, linear algebra.
Prerequisites: Linear Algebra, Probability, Statistics, Calc III, and Intro Computing/Programming - AS.110.202 AND (EN.550.310 OR EN.553.310 OR EN.553.311 OR (EN.550.420 OR EN.553.420) AND (EN.550.430 OR EN.553.430)) OR EN.560.348 AND (AS.110.201 OR AS.110.212 OR EN.553.291) AND (EN.500.112 OR EN.500.113 OR EN.500.114 OR (EN.601.220 OR EN.600.120) OR AS.250.205 OR EN.580.200 OR (EN.600.107 OR EN.601.107)); Students may receive credit for only one of EN.600.475, EN.601.475, EN.601.675.
Corequisites: NA
Instructor(s): M. Dredge
Area: Engineering
NA.
EN.601.476. Machine Learning: Data to Models. 3.0 Credits.
How can robots localize themselves in an environment when navigating? Which factors predict whether patients are at greatest risk for complications in the hospital? Can we reconstruct the brain's "connectome" from fMRI data? Many such big data questions can be answered using the paradigm of probabilistic models in machine learning. This is the second course on machine learning which focuses on probabilistic graphical models. You will learn about directed and undirected graphical models, inference methods, sampling, structure learning algorithms, latent variables, and temporal models. There will be regular assignments, which include theory and some programming. Students will analyze real data for their final project, applying methods discussed in class and writing up a report of their results. [Analysis or Applications] Students may receive credit for EN.600.476 or EN.600.676, but not both.
Prerequisites: EN.600.475 OR EN.601.475 or equivalent.;Students may receive credit for only one of EN.600.476, EN.601.476, EN.601.676.
Corequisites: NA
Instructor(s): D. Malinsky
Area: Engineering, Quantitative and Mathematical Sciences
NA.

EN.601.477. Causal Inference. 3.0 Credits.
Big data" is not necessarily "high quality data." Systematically missing records, unobserved confounders, and selection effects present in many datasets make it harder than ever to answer scientifically meaningful questions. This course will teach mathematical tools to help you reason about causes, effects, and bias sources in data with confidence. We will use graphical causal models, and potential outcomes to formalize what causal effects mean, describe how to express these effects as functions of observed data, and use regression model techniques to estimate them. We will consider techniques for handling missing values, structure learning algorithms for inferring causal directionality from data, and connections between causal inference and reinforcement learning. [Analysis] Pre-requisites: familiarity with the R programming language, multivariate calculus, basics of linear algebra and probability.
Prerequisites: EN.600.475/EN.601.475 OR (EN.550.310 OR EN.550.420) AND AS.110.202 or permission of instructor.;Students may receive credit for only one of EN.600.477, EN.600.677, EN.601.477, EN.601.677.
Corequisites: NA
Instructor(s): I. Shpitser
Area: Engineering, Quantitative and Mathematical Sciences
NA.

EN.601.479. Representation Learning. 3.0 Credits.
Often the success of a machine learning project depends on the choice of features used. Machine learning has made great progress in training classification, regression and recognition systems when "good" representations, or features, of input data are available. However, much human effort is spent on designing good features which are usually knowledge-based and engineered by domain experts over years of trial and error. A natural question to ask then is "Can we automate the learning of useful features from raw data?" Representation learning algorithms such as principal component analysis aim at discovering better representations of inputs by learning transformations of data that disentangle factors of variation in data while retaining most of the information. The success of such data-driven approaches to feature learning depends not only on how much data we can process but also on how well the features that we learn correlate with the underlying unknown labels (semantic content in the data). This course will focus on scalable machine learning approaches for learning representations from large amounts of unlabeled, multi-modal, and heterogeneous data. We will cover topics including deep learning, multi-view learning, dimensionality reduction, similarity-based learning, and spectral learning. Students may receive credit for 601.479 or 601.679 but not both. [Analysis or Applications] Required course background: machine learning or basic probability and linear algebra.
Prerequisites: If you have completed EN.600.679 you may not enroll in EN.600.479.
Corequisites: NA
Instructor(s): R. Arora
Area: Engineering
NA.

EN.601.481. Machine Learning: Optimization. 3.0 Credits.
Optimization is at the heart of machine learning. Most machine learning problems can be posed as optimization problems. However, unlike mathematical optimization where the focus is on efficient algorithms for finding solutions with a high degree of accuracy as measured by optimality conditions, optimization for machine learning focuses on algorithms that are efficient and generalize well. In this course, we will focus on optimization for problems that arise in machine learning, design and analysis of algorithms for solving these problems, and the interplay of optimization and machine learning. The coursework will include homework assignments and a final project focusing on applying optimization algorithms to real world machine learning problems. [Analysis or Applications] Prerequisites: EN.601.475] OR ( EN.553.310 OR ( EN.553.420 AND EN.553.430 ) AND AS.110.201 AND AS.110.202 );Students may receive credit for only one of EN.601.481/681.
Corequisites: NA
Instructor(s): R. Arora
Area: Engineering, Quantitative and Mathematical Sciences
NA.
EN.601.482. Machine Learning: Deep Learning. 3.0 Credits.
Deep learning (DL) has emerged as a powerful tool for solving data-intensive learning problems such as supervised learning for classification or regression, dimensionality reduction, and control. As such, it has a broad range of applications including speech and text understanding, computer vision, medical imaging, and perception-based robotics. The goal of this course is to introduce the basic concepts of deep learning (DL). The course will include a brief introduction to the basic theoretical and methodological underpinnings of machine learning, commonly used architectures for DL, DL optimization methods, DL programming systems, and specialized applications to computer vision, speech understanding, and robotics. Students will be expected to solve several DL problems on standardized data sets, and will be given the opportunity to pursue team projects on topics of their choice. [Applications] Students should also consider taking EN.601.382 Deep Learning Lab as a supplement. Students may choose to skip the lab course if they already have a strong programming background and are comfortable learning on their own using online resources and tutorials. Pre-req: (AS.110.201 or AS.110.212 or EN.553.291) and (EN.553.310 or EN.553.311 or EN.553.420); Calc III and numerical optimization recommended.
Prerequisites: (AS.110.201 OR AS.110.212 OR EN.553.291) AND (EN.553.310 OR EN.553.311 OR EN.553.420 OR EN.560.348); AND EN.601.475. Calc III and numerical optimization recommended.
Corequisites: NA
Instructor(s): M. Unberath
Area: Engineering
NA.

EN.601.485. Probabilistic Models of the Visual Cortex. 3.0 Credits.
The course gives an introduction to computational models of the mammalian visual cortex. It covers topics in low-, mid-, and high-level vision. It briefly discusses the relevant evidence from anatomy, electrophysiology, imaging (e.g., fMRI), and psychophysics. It concentrates on mathematical modelling of these phenomena taking into account recent progress in probabilistic models of computer vision and developments in machine learning, such as deep networks. [Applications or Analysis]. Programming experience (Python preferred).
Prerequisites: AS.110.106 OR AS.110.108
Corequisites: NA
Instructor(s): A. Yuille
Area: Quantitative and Mathematical Sciences
NA.

EN.601.490. Introduction to Human-Computer Interaction. 3.0 Credits.
This course is designed to introduce undergraduate and graduate students to design techniques and practices in human-computer interaction (HCI), the study of interactions between humans and computing systems. Students will learn design techniques and evaluation methods, as well as current practices and exploratory approaches, in HCI through lectures, readings, and assignments. Students will practice various design techniques and evaluation methods through hands-on projects focusing on different computing technologies and application domains. This course is intended for undergraduate and graduate students in Computer Science/Cognitive Science/Psychology. Interested students from different disciplines should contact the instructor before enrolling in this course. [Applications] Recommended Background: Basic programming skills.
Prerequisites: Students can receive credit for either EN.601.490 or EN.601.690, but not both.
Corequisites: NA
Instructor(s): C. Huang
Area: Engineering
NA.

EN.601.491. Human-Robot Interaction. 3.0 Credits.
This course is designed to introduce advanced students to research methods and topics in human-robot interaction (HRI), an emerging research area focusing on the design and evaluation of interactions between humans and robotic technologies. Students will (1) learn design principles for building and research methods of evaluating interactive robot systems through lectures, readings, and assignments, (2) read and discuss relevant literature to gain sufficient knowledge of various research topics in HRI, and (3) work on a substantial project that integrates the principles, methods, and knowledge learned in this course. [Applications]
Prerequisites: EN.601.220/EN.600.120 AND EN.601.226/EN.600.226
Corequisites: NA
Instructor(s): C. Huang
Area: Engineering
NA.

EN.601.501. Computer Science Workshop. 1.0 - 3.0 Credits.
An applications-oriented, computer science project done under the supervision and with the sponsorship of a faculty member in the Department of Computer Science. Computer Science Workshop provides a student with an opportunity to apply theory and concepts of computer science to a significant project of mutual interest to the student and a Computer Science faculty member. Permission to enroll in CSW is granted by the faculty sponsor after his/her approval of a project proposal from the student. Interested students are advised to consult with Computer Science faculty members before preparing a Computer Science Workshop project proposal.
Prerequisites: You must request Independent Academic Work using the Independent Academic Work form found in Student Self-Service: Registration > Online Forms.
Corequisites: NA
Instructor(s): D. Yarowsky; J. Selinski; M. Darvish Darab; S. Smith
Area: NA
NA.

EN.601.503. Independent Study. 1.0 - 3.0 Credits.
Individual guided study for undergraduate students under the direction of a faculty member in the department. The program of study, including the credit to be assigned, must be worked out in advance between the student and the faculty member involved. Permission required.
Prerequisites: You must request Independent Academic Work using the Independent Academic Work form found in Student Self-Service: Registration > Online Forms.
Corequisites: NA
Instructor(s): Staff
Area: NA
NA.

EN.601.507. Undergraduate Research. 1.0 - 3.0 Credits.
Individual research for undergraduates under the direction of a faculty member in the department. The program of research, including the credit to be assigned, must be worked out in advance between the student and the faculty member involved. Permission required.
Prerequisites: You must request Independent Academic Work using the Independent Academic Work form found in Student Self-Service: Registration > Online Forms.
Corequisites: NA
Instructor(s): Staff
Area: NA
NA.
EN.601.509. Computer Science Internship. 1.0 Credit.
Individual work in the field with a learning component, supervised by a faculty member in the department. The program of study and credit assigned must be worked out in advance between the student and the faculty member involved. Students may not receive credit for work that they are paid to do. As a rule of thumb, 40 hours of work is equivalent to one credit. Permission required.
Prerequisites: You must request Independent Academic Work using the Independent Academic Work form found in Student Self-Service: Registration > Online Forms.
Corequisites: NA
Instructor(s): Staff
Area: NA
NA.

EN.601.517. Group Undergraduate Research. 1.0 - 3.0 Credits.
Independent research for undergraduates under the direction of a faculty member in the department. This course has a weekly research group meeting that students are expected to attend. The program of research, including the credit to be assigned, must be worked out in advance between the student and the faculty member involved.
Prerequisites: You must request Independent Academic Work using the Independent Academic Work form found in Student Self-Service: Registration > Online Forms.
Corequisites: NA
Instructor(s): Staff
Area: NA
NA.

EN.601.519. Senior Honors Thesis. 3.0 Credits.
The student will undertake a substantial independent research project under the supervision of a faculty member, potentially leading to the notation "Departmental Honors with Thesis" on the final transcript. Students are expected to enroll in both semesters of this course during their senior year. Project proposals must be submitted and accepted in the preceding spring semester (junior year) before registration. Students will present their work publicly before April 1st of senior year. They will also submit a first draft of their project report (thesis document) at that time. Faculty will meet to decide if the thesis will be accepted for honors. Computer science majors only. Students should have a 3.5 GPA in computer science courses at the end of their junior year and permission of faculty sponsor.
Prerequisites: You must request Independent Academic Work using the Independent Academic Work form found in Student Self-Service: Registration > Online Forms.
Corequisites: NA
Instructor(s): Staff
Area: NA
NA.

EN.601.520. Senior Honors Thesis. 0.0 - 3.0 Credits.
For computer science majors only, a continuation of EN.601.519.
Recommended Course Background: EN.601.519
Prerequisites: You must request Independent Academic Work using the Independent Academic Work form found in Student Self-Service: Registration > Online Forms.
Corequisites: NA
Instructor(s): Staff
Area: NA
NA.

EN.601.556. Senior Thesis In CIS. 3.0 Credits.
Prerequisites: You must request Independent Academic Work using the Independent Academic Work form found in Student Self-Service: Registration > Online Forms.;EN.600.445 or permission of instructor.
Corequisites: NA
Instructor(s): R. Taylor
Area: NA
NA.

EN.601.611. Computer Science Innovation & Entrepreneurship II. 3.0 Credits.
This course is the second half of a two-course sequence and is a continuation of course EN.660.410.01, CS Innovation and Entrepreneurship, offered by the Center for Leadership Education (CLE). In this sequel course the student groups, directed by CS faculty, will implement the business idea which was developed in the first course and will present the implementations and business plans to an outside panel made up of practitioners, industry representatives, and venture capitalists. [General]
Prerequisites: NA
Corequisites: NA
Instructor(s): A. Dahbura; L. Aronhime
Area: Engineering
NA.

EN.601.614. Computer Networks. 4.0 Credits.
This course considers intersystem communications issues. Topics covered include layered network architectures; the OSI model; bandwidth, data rates, modems, multiplexing, error detection/correction; switching; queuing models, circuit switching, packet switching; performance analysis of protocols, local area networks; and congestion control. Recommended Course Background: EN.601.220 and EN.601.229.
Students can only receive credit for EN.600.344 or EN.601.414, not both.
Prerequisites: NA
Corequisites: NA
Instructor(s): X. Jin
Area: Engineering
NA.

EN.601.615. Databases. 3.0 Credits.
Same material as 601.415, for graduate students. Introduction to database management systems and database design, focusing on the relational and object-oriented data models, query languages and query optimization, transaction processing, parallel and distributed databases, recovery and security issues, commercial systems and case studies, heterogeneous and multimedia databases, and data mining. [Systems] (www.cs.jhu.edu/~yarowsky/cs415.html) Recommended Course Background: EN.601.226
Prerequisites: Students may receive credit for only one of EN.600.315, EN.600.415, EN.601.315, EN.601.415, EN.601.615.
Corequisites: NA
Instructor(s): D. Yarowsky
Area: Engineering
NA.

EN.601.616. Databases. 3.0 Credits.
EN.601.617. Distributed Systems. 3.0 Credits.
Graduate version of 601.317 Systems. Students may receive credit for 601.317 or 601.417 but not both. Recommended Course Background: EN.601.220, EN.601.226
Prerequisites: Students may receive credit for only one of 417/617
Corequisites: NA
Instructor(s): Y. Amir
Area: Engineering
NA.

EN.601.618. Operating Systems. 3.0 Credits.
Same material as 601.418, for graduate students. This course covers fundamental topics related to operating systems theory and practice. Topics include processor management, storage management, concurrency control, multi-programming and processing, device drivers, operating system components (e.g., file system, kernel), modeling and performance measurement, protection and security, and recent innovations in operating system structure. Course work includes the implementation of operating systems techniques and routines, and critical parts of a small but functional operating system. [Systems] Recommended Course Background: 601.226 and 601.229.
Prerequisites: Students may receive credit for only one of EN.600.318, EN.600.418, EN.601.318, EN.601.418, EN.601.618.
Corequisites: NA
Instructor(s): P. Huang
Area: Engineering
NA.

EN.601.619. Cloud Computing. 3.0 Credits.
Clouds host a wide range of the applications that we rely on today. In this course, we study common cloud applications, traffic patterns that they generate, critical networking infrastructures that support them, and core networking and distributed systems concepts, algorithms, and technologies used inside clouds. We will also study how today’s application demand is influencing the network’s design, explore current practice, and how we can build future’s networked infrastructure to better enable both efficient transfer of big data and low-latency requirements of real-time applications. The format of this course will be a mix of lectures, discussions, assignments, and a project designed to help students practice and apply the theories and techniques covered in the course. [Systems] Prerequisites: EN.601.226 or permission. Students can only receive credit for one of 601.419/619. Recommended: a course in operating systems, networks or systems programming.
Prerequisites: Students may earn credit for EN.601.419 or EN.601.619, but not both.
Corequisites: NA
Instructor(s): S. Ghorbani Khaledi
Area: Engineering
NA.

EN.601.620. Parallel Programming. 3.0 Credits.
Graduate level version of EN.601.420. Recommended Course Background: EN.601.220 AND EN.601.229.
Prerequisites: NA
Corequisites: NA
Instructor(s): R. Burns
Area: Engineering
NA.

EN.601.621. Obj Orient Software Eng. 3.0 Credits.
Same material as EN.601.421, for graduate students. This course covers object-oriented software construction methodologies and their application. The main component of the course is a large team project on a topic of your choosing. Course topics covered include object-oriented analysis and design, UML, design patterns, refactoring, program testing, code repositories, team programming, and code reviews. [Systems or Applications] (http://pl.cs.jhu.edu/OOSE/index.shtml) Recommended Course Background: EN.601.226 and EN.601.220
Prerequisites: Students may receive credit for only one of EN.600.321, EN.600.421, EN.601.421, EN.601.621.
Corequisites: NA
Instructor(s): L. Facchinetti
Area: Engineering
NA.

EN.601.622. Software Testing & Debugging. 3.0 Credits.
Studies show that testing can account for over 50% of software development costs. This course presents a comprehensive study of software testing, principles, methodologies, tools, and techniques. Topics include testing principles, coverage (graph coverage, logic coverage, input space partitioning, and syntax-based coverage), unit testing, higher-order testing (integration, system-level, acceptance), testing approaches (white-box, black-box, grey-box), regression testing, debugging, delta debugging, and several specific types of functional and non-functional testing as schedule/interest permits (GUI testing, usability testing, security testing, load/performance testing, A/B testing etc.). For practical topics, state-of-the-art tools/techniques will be studied and utilized. [Systems]
Prerequisites: EN.601.290 OR EN.601.421 OR EN.601.621; Students can only take EN.601.422 or EN.601.622, but not both.
Corequisites: NA
Instructor(s): M. Darvish Darab
Area: Engineering
NA.

EN.601.626. Principles of Programming Languages. 3.0 Credits.
Same material as EN.601.426, for graduate students. Functional, object-oriented, and other language features are studied independent of a particular programming language. Students become familiar with these features by implementing them. Most of the implementations are in the form of small language interpreters. Some type checkers and a small compiler will also be written. The total amount of code written will not be overly large, as the emphasis is on concepts. The ML programming language is the implementation language used. [Analysis] Students may receive credit for only one of EN.601.426/626. Required course background: EN.601.226.
Prerequisites: NA
Corequisites: NA
Instructor(s): S. Smith
Area: Engineering, Quantitative and Mathematical Sciences
NA.
EN.601.627. Principles of Programming Languages II. 3.0 Credits.
This course is designed as a follow-on to Principles of Programming Languages. It will cover a wide array of fundamental topics in programming languages, including advanced functional programming, the theory of inductive definitions, advanced operational semantics, advanced type systems, program analysis, program verification, theorem provers and SAT solvers. [Analysis]
Prerequisites: EN.601.426 OR EN.601.626
Corequisites: NA
Instructor(s): S. Smith
Area: NA
NA.

EN.601.628. Compilers & Interpreters. 3.0 Credits.
Prerequisites: NA
Corequisites: NA
Instructor(s): P. Froehlich
Area: Engineering
NA.

EN.601.630. Combinatorics & Graph Theory in Computer Science. 3.0 Credits.
This is a graduate level course studying the applications of combinatorics and graph theory in computer science. We will start with some basic combinatorial techniques such as counting and pigeon hole principle, and then move to advanced techniques such as the probabilistic method, spectral graph theory and additive combinatorics. We shall see their applications in various areas in computer science, such as proving lower bounds in computational models, randomized algorithms, coding theory and pseudorandomness. [Analysis]
Recommended Course Background: probability theory and linear algebra
Prerequisites: Students may receive credit for only one of 430/630
Corequisites: NA
Area: Engineering, Quantitative and Mathematical Sciences
NA.

EN.601.631. Theory of Computation. 3.0 Credits.
This is a graduate-level course studying the theoretical foundations of computer science. Topics covered will be models of computation from automata to Turing machines, computability, complexity theory, randomized algorithms, inapproximability, interactive proof systems and probabilistically checkable proofs. Students may not take both EN.601.231 and EN.601.631, unless one is for an undergrad degree and the other for grad. [Analysis]
Recommended Course Background: EN.553.171 or instructor permission.
Prerequisites: NA
Corequisites: NA
Instructor(s): X. Li
Area: Engineering, Quantitative and Mathematical Sciences
NA.

EN.601.632. Introduction to Computational Complexity. 3.0 Credits.
Same material as EN.601.432, for graduate students. This course concentrates on the design of algorithms and the rigorous analysis of their efficiency. Topics include the basic definitions of algorithmic complexity (worst case, average case); basic tools such as dynamic programming, sorting, searching, and selection; advanced data structures and their applications (such as union-find); graph algorithms and searching techniques such as minimum spanning trees, depth-first search, shortest paths, design of online algorithms and competitive analysis. [Analysis]
Prerequisites: Students may receive credit for only one of EN.600.363, EN.600.463, EN.601.433, EN.601.633.
Corequisites: NA
Instructor(s): M. Dinitz
Area: Engineering, Quantitative and Mathematical Sciences
NA.

EN.601.633. Intro Algorithms. 3.0 Credits.
Same material as EN.601.433, for graduate students. This course concentrates on the design of algorithms and the rigorous analysis of their efficiency. Topics include the basic definitions of algorithmic complexity (worst case, average case); basic tools such as dynamic programming, sorting, searching, and selection; advanced data structures and their applications (such as union-find); graph algorithms and searching techniques such as minimum spanning trees, depth-first search, shortest paths, design of online algorithms and competitive analysis. [Analysis]
Prerequisites: Students may receive credit for only one of EN.600.363, EN.600.463, EN.601.433, EN.601.633.
Corequisites: NA
Instructor(s): M. Dinitz
Area: Engineering, Quantitative and Mathematical Sciences
NA.

EN.601.634. Randomized and Big Data Algorithms. 3.0 Credits.
Same material as 601.434, for graduate students. The course emphasizes algorithmic design aspects, and how randomization can be a helpful tool. The topics covered include: tail inequalities, linear programming relaxation & randomized rounding, de-randomization, existence proofs, universal hashing, markov chains, metropolis and metropolis-hastings methods, mixing by coupling and by eigenvalues, counting problems, semi-definite programming and rounding, lower bound arguments, and applications of expanders. [Analysis] (www.cs.jhu.edu/~cs464) Required course background: EN.600.363 or EN.601.433 or EN.601.633.
Prerequisites: Students may receive credit for only one of EN.600.464, EN.600.664, EN.601.434, EN.601.634.
Corequisites: NA
Instructor(s): V. Braverman
Area: Engineering
NA.

EN.601.635. Approximation Algorithms. 3.0 Credits.
Graduate version of EN.601.435. Recommended Background: EN.601.633 or equivalent. Students may receive credit for EN.601.435 or EN.601.635, but not both.
Prerequisites: NA
Corequisites: NA
Instructor(s): M. Dinitz
Area: NA
NA.

EN.601.636. Algorithmic Game Theory. 3.0 Credits.
Same material as EN.601.436, for graduate students. This course provides an introduction to algorithmic game theory: the study of games from the perspective of algorithms and theoretical computer science. There will be a particular focus on games that arise naturally from economic interactions involving computer systems (such as economic interactions between large-scale networks, online advertising markets, etc.), but there will also be broad coverage of games and mechanisms of all sorts. Topics covered will include a) complexity of computing equilibria and algorithms for doing so, b) (in)efficiency of equilibria, and c) algorithmic mechanism design. [Analysis]
Prerequisites: NA
Corequisites: NA
Instructor(s): M. Dinitz
Area: Engineering, Quantitative and Mathematical Sciences
NA.
EN.601.640. Web Security. 3.0 Credits.
This course begins with reviewing basic knowledge of the World Wide Web, and then exploring the central defense concepts behind Web security, such as same-origin policy, cross-origin resource sharing, and browser sandboxing. It will cover the most popular Web vulnerabilities, such as cross-site scripting (XSS) and SQL injection, as well as how to attack and penetrate software with such vulnerabilities. Students will learn how to detect, respond, and recover from security incidents. Newly proposed research techniques will also be discussed. [Systems] Required background: data structures and computer system fundamentals.
Prerequisites: Students may receive credit for only one of 340/440/640
Corequisites: NA
Instructor(s): Y. Cao
Area: NA

EN.601.641. Blockchains and Cryptocurrencies. 3.0 Credits.
Same as EN.601.441, for graduate students. This course will introduce students to cryptocurrencies and the main underlying technology of Blockchains. The course will start with the relevant background in cryptography and then proceed to cover the recent advances in the design and applications of blockchains. This course should primarily appeal to students who want to conduct research in this area or wish to build new applications on top of blockchains. It should also appeal to those who have a casual interest in this topic or are generally interested in cryptography. Students are expected to have mathematical maturity. Recommended Course Background: EN.601.226 AND (EN.553.310 OR EN.553.420) [Analysis]
Prerequisites: Students may receive credit for only one of EN.600.451 OR EN.601.441 OR EN.601.641
Corequisites: NA
Instructor(s): A. Jain; M. Green
Area: Engineering

EN.601.642. Modern Cryptography. 3.0 Credits.
Same material as 601.442, for graduate students. Modern Cryptography includes seemingly paradoxical notions such as communicating privately without a shared secret, proving things without leaking knowledge, and computing on encrypted data. In this challenging but rewarding course we will start from the basics of private and public key cryptography and go all the way up to advanced notions such as zero-knowledge proofs, functional encryption and program obfuscation. The class will focus on rigorous proofs and require mathematical maturity. [Analysis] Required course background: EN.601.231 or EN.601.631.
Prerequisites: Students may receive credit for only one of EN.600.442, EN.601.442, EN.601.642.
Corequisites: NA
Instructor(s): A. Jain
Area: Engineering, Quantitative and Mathematical Sciences

EN.601.643. Security & Privacy in Computing. 3.0 Credits.
Same material as 601.443, for graduate students. Lecture topics will include computer security, network security, basic cryptography, system design methodology, and privacy. There will be a heavy work load, including written homework, programming assignments, exams and a comprehensive final. The class will also include a semester-long project that will be done in teams and will include a presentation by each group to the class. [Applications] Required Background: A basic course in operating systems and networking, or permission of instructor.
Prerequisites: Students may receive credit for only one of EN.600.443, EN.601.443, EN.601.643.
Corequisites: NA
Instructor(s): A. Rubin
Area: Engineering

EN.601.644. Network Security. 3.0 Credits.
Same as EN.601.444, for graduate students. This course focuses on communication security in computer systems and networks. The course is intended to provide students with an introduction to the field of network security. The course covers network security services such as authentication and access control, integrity and confidentiality of data, firewalls and related technologies, Web security and privacy. Course work involves implementing various security techniques. A course project is required. [Systems] Recommended. Course Background: EN.601.220, EN.601.226 or equivalent
Prerequisites: Students may receive credit for only one of EN.600.424, EN.650.424, EN.601.444, EN.601.644.
Corequisites: NA
Instructor(s): S. Nielson
Area: Engineering

EN.601.645. Practical Cryptographic Systems. 3.0 Credits.
Same material as 601.445, for graduate students. This semester-long course will teach systems and cryptographic design principles by example: by studying and identifying flaws in widely-deployed cryptographic products and protocols. Our focus will be on the techniques used in practical security systems, the mistakes that lead to failure, and the approaches that might have avoided the problem. We will place a particular emphasis on the techniques of provable security and the feasibility of reverse-engineering undocumented cryptographic systems. [Systems]
Prerequisites: Students may receive credit for only one of EN.600.454, EN.650.454, EN.601.445, EN.601.645.
Corequisites: NA
Instructor(s): M. Green
Area: Engineering
EN.601.647. Computational Genomics: Sequences. 3.0 Credits.
Same material as 601.447, for graduate students. Your genome is the blueprint for the molecules in your body. It’s also a string of letters (A, C, G and T) about 3 billion letters long. How does this string give rise to you? Your heart, your brain, your health? This, broadly speaking, is what genomics research is about. This course will familiarize you with a breadth of topics from the field of computational genomics. The emphasis is on current research problems, real-world genomics data, and efficient software implementations for analyzing data. Topics will include: string matching, sequence alignment and indexing, assembly, and sequence models. Course will involve significant programming projects. [Applications] Recommended Course Background: EN.601.220 and EN.601.226
Prerequisites: Students may receive credit for only one of EN.600.439, EN.600.639, EN.601.447, EN.601.647.
Corequisites: NA
Instructor(s): B. Langmead; B. Solomon
Area: Engineering
NA.

EN.601.648. Computational Genomics: Data Analysis. 3.0 Credits.
Same material as EN.601.448, for graduate students. Genomic data has the potential to reveal causes of disease, novel drug targets, and relationships among genes and pathways in our cells. However, identifying meaningful patterns from high-dimensional genomic data has required development of new computational tools. This course will cover current approaches in computational analysis of genomic data with a focus on statistical methods and machine learning. Topics will include disease association, prediction tasks, clustering and dimensionality reduction, data integration, and network reconstruction. There will be some programming and a project component. [Applications] Recommended Course Background: EN.600.226 or other programming experience, probability and statistics, linear algebra or calculus.
Prerequisites: NA
Corequisites: NA
Instructor(s): A. Battle
Area: Engineering
NA.

EN.601.654. Augmented Reality. 3.0 Credits.
This course introduces students to the field of Augmented Reality. It reviews its basic definitions, principles and applications. It then focuses on Medical Augmented Reality and its particular requirements. The course also discusses the main issues of calibration, tracking, multi-modal registration, advance visualization and display technologies. Homework in this course will relate to the mathematical methods used for calibration, tracking and visualization in medical augmented reality. Students may also be asked to read papers and implement various techniques within group projects. Recommended Course Background: EN.601.220, EN.601.226, and AS.110.201. [Applications]
Prerequisites: Students may receive credit for only one of EN.600.484, EN.600.684, EN.601.454, EN.601.654.
Corequisites: NA
Instructor(s): N. Navab
Area: NA
NA.

EN.601.655. Computer Integrated Surgery I. 4.0 Credits.
Same material as 601.455, for graduate students. This course focuses on computer-based techniques, systems, and applications exploiting quantitative information from medical images and sensors to assist clinicians in all phases of treatment from diagnosis to preoperative planning, execution, and follow-up. It emphasizes the relationship between problem definition, computer-based technology, and clinical application and includes a number of guest lectures given by surgeons and other experts on requirements and opportunities in particular clinical areas. [Applications] Recommended Course Background: intermediate programming in C/C++, EN.601.457, EN.601.461, image processing.
Prerequisites: Students may receive credit for only one of EN.600.445, EN.600.645, EN.601.455, EN.601.655.
Corequisites: NA
Instructor(s): R. Taylor
Area: Engineering
NA.

EN.601.656. Computer Integrated Surgery II. 3.0 Credits.
Same material as EN.601.456, for graduate students. This weekly lecture/seminar course addresses similar material to EN.601.445, but covers selected topics in greater depth. In addition to material covered in lectures/seminars by the instructor and other faculty, students are expected to read and provide critical analysis/presentations of selected papers in recitation sessions. Students taking this course are required to undertake and report on a significant term project under the supervision of the instructor and clinical end users. Typically, this project is an extension of the term project from EN.601.445, although it does not have to be. Grades are based both on the project and on classroom recitations. Students wishing to attend the weekly lectures as a 1-credit seminar should sign up for EN.600.452. [Applications]
Prerequisites: EN.600.445 OR EN.600.645 OR PERMISSION OF INSTRUCTOR; Students may receive credit for only one of EN.600.445, EN.600.645, EN.601.456, EN.601.656.
Corequisites: NA
Instructor(s): R. Taylor
Area: NA
NA.

EN.601.657. Computer Graphics. 3.0 Credits.
Same material as 601.457, for graduate students. This course introduces computer graphics techniques and applications, including image processing, rendering, modeling and animation. [Applications] Permission of instructor is required for students not satisfying a pre-requisite. Required course background: EN.601.220 (C++), EN.601.226, linear algebra.
Prerequisites: Students may receive credit for only one of EN.600.357, EN.600.457, EN.601.457, EN.601.657.
Corequisites: NA
Instructor(s): M. Kazhdan
Area: Engineering, Quantitative and Mathematical Sciences
NA.
EN.601.658. Introduction to Geometry Processing. 2.0 Credits.
In this course we will look at fundamental techniques in geometry processing, including smoothing / sharpening, parameterization, and (if time allows) vector fields. The course will begin with a review of the discretization of the underlying concepts from differential geometry (e.g., normals, curvature, Laplacian) before proceeding to specific applications.

Prerequisites: EN.600.357 AND EN.600.457
Corequisites: NA
Instructor(s): F. Prada Nino; M. Kazhdan
Area: NA
NA.

EN.601.661. Computer Vision. 3.0 Credits.
Same material as 601.461, for graduate students. Students may receive credit for at most one of 601.461/661/761. This course gives an overview of fundamental methods in computer vision from a computational perspective. Methods studied include: camera systems and their modeling, computation of 3-D geometry from binocular stereo, motion, and photometric stereo; and object recognition. Edge detection and color perception are covered as well. Elements of machine vision and biological vision are also included. [Applications] Recommended Course Background: intro programming, linear algebra, prob/stat.

Prerequisites: Students may receive credit for only one of EN.600.361, EN.600.461, EN.600.661, EN.601.461, EN.601.661.
Corequisites: NA
Instructor(s): W. Shen
Area: Engineering
NA.

EN.601.663. Algorithms for Sensor-Based Robotics. 3.0 Credits.
Same material as 601.463, for graduate students. This course surveys the development of robotic systems for navigating in an environment from an algorithmic perspective. It will cover basic kinematics, configuration space concepts, motion planning, and localization and mapping. It will describe these concepts in the context of the ROS software system, and will present examples relevant to mobile platforms, manipulation, robotics surgery, and human-machine systems. [Analysis] Recommended Course Background: EN.600.226 and Linear Algebra and Probability

Prerequisites: Students may receive credit for only one of EN.600.336, EN.600.436, EN.600.636, EN.601.463, EN.601.663.
Corequisites: NA
Instructor(s): S. Leonard
Area: Engineering
NA.

EN.601.664. Artificial Intelligence. 3.0 Credits.
Same material as 601.464, for graduate students. The class is recommended for all scientists and engineers with a genuine curiosity about the fundamental obstacles to getting machines to perform tasks such as learning, planning and prediction. Materials will be primarily based on the popular textbook, Artificial Intelligence: A Modern Approach. Strong programming skills are expected, as well as basic familiarity with probability. For students intending to also take courses in Machine Learning (e.g., EN.601.475/675, EN.601.476/676), they may find it beneficial to take this course first, or concurrently. [Applications] Prereq: EN.600.226; Recommended: linear algebra, prob/stat.

Prerequisites: Students may receive credit for only one of EN.600.335, EN.600.435, EN.601.464, EN.601.664.
Corequisites: NA
Instructor(s): B. Van Durme
Area: Engineering
NA.

EN.601.665. Natural Language Processing. 3.0 Credits.
Same material as 601.465, for graduate students. This course is an in-depth overview of techniques for processing human language. How should linguistic structure and meaning be represented? What algorithms can recover them from text? And crucially, how can we build statistical models to choose among the many legal answers? The course covers methods for trees (parsing and semantic interpretation), sequences (finite-state transduction such as morphology), and words (sense and phrase induction), with applications to practical engineering tasks such as information retrieval and extraction, text classification, part-of-speech tagging, speech recognition and machine translation. There are a number of structured but challenging programming assignments. [Applications] Recommended Course Background: EN.601.226

Prerequisites: Students may receive credit for only one of EN.600.465, EN.601.465, EN.601.665.
Corequisites: NA
Instructor(s): K. Duh
Area: Engineering
NA.

EN.601.666. Information Retrieval and Web Agents. 3.0 Credits.
Same material as EN.601.466, for graduate students. An in-depth, hands-on study of current information retrieval techniques and their application to developing intelligent WWW agents. Topics include a comprehensive study of current document retrieval models, mail/news routing and filtering, document clustering, automatic indexing, query expansion, relevance feedback, user modeling, information visualization and usage pattern analysis. In addition, the course explores the range of additional language processing steps useful for template filling and information extraction from retrieved documents, focusing on recent, primarily statistical methods. The course concludes with a study of current issues in information retrieval and data mining on the World Wide Web. Topics include web robots, spiders, agents and search engines, exploring both their practical implementation and the economic and legal issues surrounding their use. [Applications]

Prerequisites: NA
Corequisites: NA
Instructor(s): D. Yarowsky
Area: Engineering
NA.

EN.601.667. Introduction to Human Language Technology. 3.0 Credits.
This course gives an overview of basic foundations and applications of human language technology, such as: morphological, syntactic, semantic, and pragmatic processing; machine learning; signal processing; speech recognition; speech synthesis; information retrieval; text classification; topic modelling; information extraction; knowledge representation; machine translation; dialog systems; etc. [Applications] Required Background: EN.601.226 Data Structures (not enforced for 6xx course number); knowledge of Python recommended.

Prerequisites: NA
Corequisites: NA
Instructor(s): P. Koehn
Area: NA
NA.
EN.601.668. Machine Translation. 3.0 Credits.
Same material as 601.468, for graduate students. Google translate can instantly translate between any pair of over fifty human languages (for instance, from French to English). How does it do that? Why does it make the errors that it does? And how can you build something better? Modern translation systems learn to translate by reading millions of words of already translated text, and this course will show you how they work. The course covers a diverse set of fundamental building blocks from linguistics, machine learning, algorithms, data structures, and formal language theory, along with their application to a real and difficult problem in artificial intelligence. [Applications] Recommended Course Background: prob/stat, EN.601.226; EN.601.465
Prerequisites: Students may receive credit for only one of EN.600.468, EN.601.468, EN.601.668.
Corequisites: NA
Instructor(s): P. Koehn
Area: Engineering

EN.601.675. Machine Learning. 3.0 Credits.
Same material as 601.475, for graduate students. Machine learning is a subfield of computer science and artificial intelligence, whose goal is to develop computational systems, methods, and algorithms that can learn from data to improve their performance. This course introduces the foundational concepts of modern Machine Learning, including core principles, popular algorithms and modeling platforms. This will include both supervised learning, which includes popular algorithms like SVMs, logistic regression, boosting and deep learning, as well as unsupervised learning frameworks, which include Expectation Maximization and graphical models. Homework assignments include a heavy programming component, requiring students to implement several machine learning algorithms in a common learning framework. Additionally, analytical homework questions will explore various machine learning concepts, building on the pre-requisites that include probability, linear algebra, multivariate calculus and basic optimization. Students in the course will develop a learning system for a final project. [Applications or Analysis] Recommended Course Background: multivariable calculus, probability, linear algebra.
Prerequisites: Students may receive credit for only one of EN.600.475, EN.601.475, EN.601.675.
Corequisites: NA
Instructor(s): M. Dredze
Area: Engineering

EN.601.676. Machine Learning: Data to Models. 3.0 Credits.
Same material as EN.601.476, for graduate students. How can robots localize themselves in an environment when navigating? Which factors predict whether patients are at greatest-risk for complications in the hospital? Can we reconstruct the brain’s “connectome” from fMRI data? Many such big data questions can be answered using the paradigm of probabilistic models in machine learning. This is the second course on machine learning which focuses on probabilistic graphical models. You will learn about directed and undirected graphical models, inference methods, sampling, structure learning algorithms, latent variables, and temporal models. There will be regular assignments, which include theory and some programming. Students will analyze real data for their final project, applying methods discussed in class and writing up a report of their results. [Analysis or Applications] Recommended Background: EN.600.475 or EN.601.675 or equivalent. Students may receive credit for EN.600.475 or EN.600.675, but not both.
Prerequisites: Students may receive credit for only one of EN.600.476, EN.601.476, EN.601.676.
Corequisites: NA
Instructor(s): D. Malinsky
Area: Engineering, Quantitative and Mathematical Sciences

EN.601.677. Causal Inference. 3.0 Credits.
Big data” is not necessarily “high quality data.” Systematically missing records, unobserved confounders, and selection effects present in many datasets make it harder than ever to answer scientifically meaningful questions. This course will teach mathematical tools to help you reason about causes, effects, and bias sources in data with confidence. We will use graphical causal models, and potential outcomes to formalize what causal effects mean, describe how to express these effects as functions of observed data, and use regression model techniques to estimate them. We will consider techniques for handling missing values, structure learning algorithms for inferring causal directionality from data, and connections between causal inference and reinforcement learning. [Analysis] Pre-requisites: familiarity with the R programming language, multivariate calculus, basics of linear algebra and probability.
Prerequisites: Students may receive credit for only one of EN.600.477, EN.600.677, EN.601.477, EN.601.677.
Corequisites: NA
Instructor(s): I. Shpitser
Area: Engineering, Quantitative and Mathematical Sciences
EN.601.681. Machine Learning: Optimization. 3.0 Credits.
Same material as EN.601.481, for graduate students. Optimization is at the heart of machine learning. Most machine learning problems can be posed as optimization problems. However, unlike mathematical optimization where the focus is on efficient algorithms for finding solutions with a high degree of accuracy as measured by optimality conditions, optimization for machine learning focuses on algorithms that are efficient and generalize well. In this course, we will focus on optimization for problems that arise in machine learning, design and analysis of algorithms for solving these problems, and the interplay of optimization and machine learning. The course work will include homework assignments and a final project focusing on applying optimization algorithms to real world machine learning problems. [Analysis or Applications] Recommended Course Background: EN.601.475 OR (EN.553.310 OR (EN.553.420 AND EN.553.430) AND AS.110.201 AND AS.110.202)
Prerequisites: Students may receive credit for only one of EN.601.481/681
Corequisites: NA
Instructor(s): R. Arora
Area: NA
NA.

EN.601.682. Machine Learning: Deep Learning. 3.0 Credits.
Deep learning (DL) has emerged as a powerful tool for solving data-intensive learning problems such as supervised learning for classification or regression, dimensionality reduction, and control. As such, it has a broad range of applications including speech and text understanding, computer vision, medical imaging, and perception-based robotics. The goal of this course is to introduce the basic concepts of deep learning (DL). The course will include a brief introduction to the basic theoretical and methodological underpinnings of machine learning, commonly used architectures for DL, DL optimization methods, DL programming systems, and specialized applications to computer vision, speech understanding, and robotics. Students will be expected to solve several DL problems on standardized data sets, and will be given the opportunity to pursue team projects on topics of their choice. [Applications] Students should also consider taking EN.601.382 Deep Learning Lab as a supplement. Students may choose to skip the lab course if they already have a strong programming background and are comfortable learning on their own using online resources and tutorials. Recommended Course Background: (AS.110.201 or AS.110.212 or EN.553.291) and (EN.553.310 OR EN.553.311 OR EN.553.420); numerical optimization recommended.
Prerequisites: NA
Corequisites: NA
Instructor(s): M. Unberath
Area: Engineering
NA.

EN.601.685. Probabilistic Models in the Visual Cortex. 3.0 Credits.
The course gives an introduction to computational models of the mammalian visual cortex. It covers topics in low-, mid-, and high-level vision. It briefly discusses the relevant evidence from anatomy, electrophysiology, imaging (e.g., fMRI), and psychophysics. It concentrates on mathematical modelling of these phenomena taking into account recent progress in probabilistic models of computer vision and developments in machine learning, such as deep networks. [Applications or Analysis] Programming experience (Python preferred).
Prerequisites: NA
Corequisites: NA
Instructor(s): A. Yuille
Area: Quantitative and Mathematical Sciences
NA.

EN.601.690. Introduction to Human-Computer Interaction. 3.0 Credits.
This course is designed to introduce undergraduate and graduate students to design techniques and practices in human-computer interaction (HCI), the study of interactions between humans and computing systems. Students will learn design techniques and evaluation methods, as well as current practices and exploratory approaches, in HCI through lectures, readings, and assignments. Students will practice various design techniques and evaluation methods through hands-on projects focusing on different computing technologies and application domains. This course is intended for undergraduate and graduate students in Computer Science/Cognitive Science/Psychology. Interested students from different disciplines should contact the instructor before enrolling in this course. [Applications] Recommended Background: Basic programming skills.
Prerequisites: Students can receive credit for either EN.601.490 or EN.601.690, but not both.
Corequisites: NA
Instructor(s): C. Huang
Area: Engineering
NA.

EN.601.691. Human-Robot Interaction. 3.0 Credits.
This course is designed to introduce graduate students to research methods and topics in human-robot interaction (HRI), an emerging research area focusing on the design and evaluation of interactions between humans and robotic technologies. Students will (1) learn design principles for building and research methods of evaluating interactive robot systems through lectures, readings, and assignments, (2) read and discuss relevant literature to gain sufficient knowledge of various research topics in HRI, and (3) work on a substantial project that integrates the principles, methods, and knowledge learned in this course. [Applications] Required course background: EN.601.220 and EN.601.226.
Prerequisites: NA
Corequisites: NA
Instructor(s): C. Huang
Area: NA
NA.

EN.601.714. Advanced Computer Networks. 3.0 Credits.
This is a graduate-level course on computer networks. It provides a comprehensive overview on advanced topics in network protocols and networked systems. The course will cover both classic papers on Internet protocols and recent research results. It will examine a wide range of topics, e.g., routing, congestion control, network architectures, datacenter networks, network virtualization, software-defined networking, and programmable networks, with an emphasize on core networking concepts and principles. The course will include lectures, paper discussions, programming assignments and a research project. Recommended Course Background: One undergraduate course in computer networks (e.g., EN.601.414/614 Computer Network Fundamentals or the equivalent), or permission of the instructor. The course assignments and projects assume students to be comfortable with programming.
Prerequisites: NA
Corequisites: NA
Instructor(s): X. Jin
Area: NA
NA.
EN.601.717. Advanced Distributed Systems & Networks. 3.0 Credits.
The course explores the state of the art in distributed systems, networks and Internet research and practice, trying to see what it would take to push the envelope a step further. The course is conducted as a discussion group, where the professor and students brainstorm and pick interesting semester-long projects with high potential future impact. Example areas include robust scalable infrastructure (distributed datacenters, cloud networking, scada systems), real-time performance (remote surgery, trading systems), hybrid networks (mesh networks, 3G/4G/Wifi/Bluetooth). Students should feel free to bring their own topics of interest and ideas. Recommended Course Background: a systems course (distributed systems, operating systems, computer networks, parallel programming) or permission of instructor.
Prerequisites: NA
Corequisites: NA
Instructor(s): Y. Amir
Area: NA

EN.601.718. Advanced Operating Systems. 3.0 Credits.
Students will study advanced operating system topics and be exposed to recent developments in operating systems research. This course involves readings on classic and new papers. Topics include virtual memory management, synchronization and communication, file systems, protection and security, operating system structure and extension techniques, fault tolerance, and history and experience of systems programming. [Systems]
Prerequisites: EN.600.318 OR EN.600.418 OR EN.601.318 OR EN.601.418 OR EN.601.618
Corequisites: NA
Instructor(s): P. Huang
Area: NA

EN.601.723. Advanced Topics in Data-Intensive Computing. 3.0 Credits.
The advent of cloud computing has lead to an explosion of storage system and data analysis software, including NoSQL databases, bulk-synchronous processing, graph computing engines, and stream processing. This course will explore scale-out software architectures for data-processing tasks. It will examine the algorithms and data-structures that underlie scalable systems and look at how hardware and networking trends influence the design and deployment of cloud computing. Recommended Course Background: EN.601.320/420 or permission of instructor. [Systems]
Prerequisites: EN.600.320 OR EN.600.420 OR EN.601.620
Corequisites: NA
Instructor(s): R. Burns
Area: NA

EN.601.720. Advanced Topics in Applied Cryptography. 3.0 Credits.
This reading and project based course will explore the latest research in the area of applied cryptography and cryptographic engineering. Topics covered will include zero knowledge, efficient multiparty computation, cryptocurrencies, and trusted computing hardware. Readings will be drawn from the latest applied cryptography and security conferences. The course will include both reading, critical analysis, presentations and a course programming project. [Analysis or Applications]
Prerequisites: EN.600.454 OR EN.601.445 OR EN.601.645 OR EN.600.442 OR EN.601.442 OR EN.601.642
Corequisites: NA
Instructor(s): M. Green
Area: NA

EN.601.730. Pseudorandomness and Combinatorial Constructions. 3.0 Credits.
Randomness is very useful in almost all areas of computer science, such as algorithms, distributed computing and cryptography. However, computers generally do not have access to truly uniform random bits. To deal with this, we rely on various pseudorandom objects to reduce either the quantity or the quality of the random bits needed. In this course, we will develop provably good pseudorandom objects for a variety of tasks. We will frequently require explicit combinatorial constructions. That is, we will want to efficiently and deterministically construct such objects. Along the way, we will also explore the close connections of such objects to many other areas in computer science and mathematics, such as graph theory, coding theory, complexity theory and arithmetic combinatorics. [Analysis] Recommended course background: EN.601.231 or EN.601.631, and probability.
Prerequisites: NA
Corequisites: NA
Instructor(s): X. Li
Area: NA

EN.601.743. Advanced Topics in Computer Security. 3.0 Credits.
Topics will vary from year to year, but will focus mainly on network perimeter protection, host-level protection, authentication technologies, intellectual property protection, formal analysis techniques, intrusion detection and similarly advanced subjects. Emphasis in this course is on understanding how security issues impact real systems, while maintaining an appreciation for grounding the work in fundamental science. Students will study and present various advanced research papers to the class. There will be homework assignments and a course project.
Prerequisites: NA
Corequisites: NA
Instructor(s): A. Rubin
Area: NA

EN.601.745. Advanced Topics in Applied Cryptography. 3.0 Credits.
EN.601.749. Computational Genomics: Applied Comparative Genomics. 3.0 Credits.
The goal of this course is to study the leading computational and quantitative approaches for comparing and analyzing genomes starting from raw sequencing data. The course will focus on human genomics and human medical applications, but the techniques will be broadly applicable across the tree of life. The topics will include genome assembly & comparative genomics, variant identification & analysis, gene expression & regulation, personal genome analysis, and cancer genomics. The grading will be based on assignments, a midterm & final exam, class presentations, and a significant class project. [Applications] Expected course background: familiarity with UNIX scripting and/or programming. 
Prerequisites: NA
Corequisites: NA
Instructor(s): M. Schatz
Area: NA
NA.

EN.601.760. FFT in Graphics & Vision. 3.0 Credits.
In this course, we will study the Fourier Transform from the perspective of representation theory. We will begin by considering the standard transform defined by the commutative group of rotations in 2D and translations in two- and three-dimensions, and will proceed to the Fourier Transform of the non-commutative group of 3D rotations. Subjects covered will include correlation of images, shape matching, computation of invariances, and symmetry detection. Recommended Course Background: AS.110.201 and comfort with mathematical derivations.
Prerequisites: NA
Corequisites: NA
Instructor(s): M. Kazhdan
Area: NA
NA.

EN.601.765. Machine Learning: Linguistic & Sequence Modeling. 3.0 Credits.
This course surveys formal ingredients that are used to build structured models of character and word sequences. We will unpack recent deep learning architectures that consider various kinds of latent structure, and see how they draw on earlier work in structured prediction, dimensionality reduction, Bayesian nonparametrics, multi-task learning, etc. We will also examine a range of strategies used for inference and learning in these models. Students will be expected to read recent papers and carry out a research project. [Applications] or [Analysis]
Prerequisites: EN.600.465/EN.601.465 or EN.601.665
Corequisites: NA
Instructor(s): J. Eisner
Area: NA
NA.

EN.601.769. Events Semantics in Theory and Practice. 3.0 Credits.
This course explores selected topics in the nature of event representations from the perspective of cognitive science, computer science, linguistics, and philosophy. These fields have developed a rich array of scientific theories about the representation of events, and how humans make inferences about them -- we investigate how (and if) such theories could be applied to current research topics and tasks in computational semantics such as inference from text, automated summarization, veridicality assessment, and so on. In addition to classic articles dealing with formal semantic theories, the course considers available machine-readable corpora, ontologies, and related resources that bear on event structure, such as WordNet, PropBank, FrameNet, etc.
The course is aimed to marry theory with practice: students with either a computational or linguistic background are encouraged to participate. [Applications]
Prerequisites: NA
Corequisites: NA
Instructor(s): B. Van Durme; K. Rawlins
Area: NA
NA.

EN.601.775. Statistical Machine Learning. 3.0 Credits.
This is a second graduate level course in machine learning. It will provide a formal and an in-depth coverage of topics at the interface of statistical theory and computational sciences. We will revisit popular machine learning algorithms and understand their performance in terms of the size of the data (sample complexity), memory needed (space complexity), as well as the overall computational runtime (computation or iteration complexity). We will cover topics including nonparametric methods, kernel methods, online learning and reinforcement learning, as well as introduce students to current topics in large-scale machine-learning and randomized projections. Topics will vary from year-to-year but the general focus would be on combining methodology with theoretical and computational foundations. [Analysis or Applications]
Prerequisites: NA
Corequisites: NA
Instructor(s): R. Arora
Area: NA
NA.

EN.601.778. Advanced Topics in Causal Inference. 3.0 Credits.
This course will cover advanced topics on all areas of causal inference, including learning causal effects, path-specific effects, and optimal policies from data featuring biases induced by missing data, confounders, selection, and measurement error; techniques for generalizing findings to different populations, complex probabilistic models relevant for causal inference applications, learning causal structure from data, and inference under interference and network effects. The course will feature a final project which would involve either an applied data analysis problem (with a causal inference flavor), a literature review, or theoretical work. [Analysis] Pre-requisite: EN.600.477/677 or permission.
Prerequisites: EN.600.477 OR EN.600.677
Corequisites: NA
Instructor(s): I. Shpitser
Area: NA
NA.
EN.601.779. Machine Learning: Advanced Topics. 3.0 Credits.
This course will focus on recent advances in machine learning. Topics will vary from year to year. The course will be project focused and involve presenting and discussing recent research papers.
Prerequisites: EN.600.475/EN.601.475 or EN.600.675/EN.601.675 or EN.600.775/EN.601.775 or EN.600.479/EN.601.479 or EN.600.679/EN.601.679 or EN.600.476/EN.601.476 or EN.600.676/EN.601.676 or permission.
Corequisites: NA
Instructor(s): R. Arora
Area: NA

EN.601.783. Vision as Bayesian Inference. 3.0 Credits.
This is an advanced course on computer vision from a probabilistic and machine learning perspective. It covers techniques such as linear and non-linear filtering, geometry, energy function methods, markov random fields, conditional random fields, graphical models, probabilistic grammars, and deep neural networks. These are illustrated on a set of vision problems ranging from image segmentation, semantic segmentation, depth estimation, object recognition, object parsing, scene parsing, action recognition, and text captioning. [Analysis or Applications] Required course background: calculus, linear algebra (AS.110.201 or equiv.), probability and statistics (AS.553.311 or equiv.), and the ability to program in Python and C++. Background in computer vision (EN.601.461/661) and machine learning (EN.601.475) suggested but not required.
Prerequisites: NA
Corequisites: NA
Instructor(s): A. Yuille
Area: NA

EN.601.801. Computer Science Seminar. 1.0 Credit.
Required for all full-time CS PhD students. Recommended for MSE students.
Prerequisites: NA
Corequisites: NA
Instructor(s): S. Smith
Area: NA

EN.601.803. Masters Research. 3.0 - 10.0 Credits.
Permission required. Independent research for masters or pre-dissertation PhD students.
Prerequisites: NA
Corequisites: NA
Instructor(s): Staff
Area: NA

EN.601.805. Graduate Independent Study. 1.0 - 3.0 Credits.
Permission required. Individual study in an area of mutual interest to a graduate student and a faculty member in the department.
Prerequisites: NA
Corequisites: NA
Instructor(s): Staff
Area: NA

EN.601.807. Teaching Practicum. 1.0 Credit.
PhD students will gain valuable teaching experience, working closely with their assigned faculty supervisor. Successful completion of this course fulfills the PhD teaching requirement. (grad students) Permission req’d.
Prerequisites: NA
Corequisites: NA
Instructor(s): J. Selinski; M. Schatz; S. Smith
Area: NA

EN.601.809. PhD Research. 3.0 - 20.0 Credits.
NA
Prerequisites: NA
Corequisites: NA
Instructor(s): Staff
Area: NA

EN.601.817. Selected Topics in Systems Research. 1.0 Credit.
This course covers latest advances in the research of computer systems including operating systems, distributed system, mobile and cloud computing. Students will read and discuss recent research papers in top systems conferences. Each week, one student will present the paper and lead the discussion for the week. The focus topics covered in the papers vary semester to semester. Example topics include fault-tolerance, reliability, verification, energy efficiency, and virtualization.
Prerequisites: NA
Corequisites: NA
Instructor(s): P. Huang
Area: NA

EN.601.826. Selected Topics in Programming Languages. 1.0 Credit.
This seminar course covers recent developments in the foundations of programming language design and implementation. Topics covered include type theory, process algebra, higher-order program analysis, and constraint systems. Students will be expected to present papers orally.
Prerequisites: NA
Corequisites: NA
Instructor(s): S. Smith
Area: NA

EN.601.831. CS Theory Seminar. 1.0 Credit.
Seminar series in theoretical computer science. Topics include algorithms, complexity theory, and related areas of TCS. Speakers will be a mix of internal and external researchers, mostly presenting recently published research papers.
Prerequisites: NA
Corequisites: NA
Instructor(s): M. Dinitz; X. Li
Area: NA

EN.601.833. Seminar in Algorithms. 1.0 Credit.
This course will explore algorithms and theoretical computer science with a focus on algorithms for massive data. Examples of topics include streaming algorithms, approximation algorithms, online algorithms. Students will be encouraged to select a paper and lead a discussion. External speakers will be invited to present current work as well. This course is a good opportunity for motivated students to learn modern algorithmic methods. Recommended Course Background: EN.601.433 or equivalent.
Prerequisites: NA
Corequisites: NA
Instructor(s): V. Braverman
Area: Engineering
NA.

EN.601.845. Selected Topics in Applied Cryptography. 1.0 Credit.
In this course students will read, discuss and present current research papers in applied cryptography. Topic coverage will vary each semester. Instructor approval required.
Prerequisites: NA
Corequisites: NA
Instructor(s): M. Green
Area: NA
NA.

EN.601.856. Seminar: Medical Image Analysis. 1.0 Credit.
This weekly seminar will focus on research issues in medical image analysis, including image segmentation, registration, statistical modeling, and applications. It will also include selected topics relating to medical image acquisition, especially where they relate to analysis. The purpose of the course is to provide the participants with a thorough background in current research in these areas, as well as to promote greater awareness and interaction between multiple research groups within the University. The format of the course is informal. Students will read selected papers. All students will be assumed to have read these papers by the time the paper is scheduled for discussion. But individual students will be assigned on a rotating basis to lead the discussion on particular papers or sections of papers. Co-listed with En.520.746.
Prerequisites: NA
Corequisites: NA
Instructor(s): J. Prince; R. Taylor
Area: NA
NA.

EN.601.857. Selected Topics in Computer Graphics. 1.0 Credit.
In this course we will review current research in computer graphics. We will meet for an hour once a week and one of the participants will lead the discussion for the week.
Prerequisites: NA
Corequisites: NA
Instructor(s): M. Kazhdan
Area: NA
NA.

EN.601.866. Selected Topics in Computational Semantics. 1.0 Credit.
This weekly reading group will review current research and survey articles on the topics of computational semantics, statistical machine translation, and natural language generation. Enrolled students will present papers and lead discussions.
Prerequisites: NA
Corequisites: NA
Instructor(s): B. Van Durme
Area: NA
NA.

EN.601.868. Selected Topics in Machine Translation. 1.0 Credit.
Students in this course will review, present, and discuss current research in machine translation. Permission of instructor.
Prerequisites: NA
Corequisites: NA
Instructor(s): P. Koehn
Area: NA
NA.

EN.601.875. Selected Topics in Machine Learning. 1.0 Credit.
This seminar is recommended for all students interested in data intensive computing research areas (e.g., machine learning, computer vision, natural language processing, speech, computational social science). The meeting format is participatory. Papers that discuss best practices and the state-of-the-art across application areas of machine learning and data intensive computing will be read. Student volunteers lead individual meetings. Faculty and external speakers present from time-to-time. Recommended Course Background: machine learning or permission of the instructor.
Prerequisites: NA
Corequisites: NA
Instructor(s): R. Arora
Area: NA
NA.

Cross Listed Courses

Cognitive Science
AS.050.375. Probabilistic Models of the Visual Cortex. 3.0 Credits.
The course gives an introduction to computational models of the mammalian visual cortex. It covers topics in low-, mid-, and high-level vision. It briefly discusses the relevant evidence from anatomy, electrophysiology, imaging (e.g., fMRI), and psychophysics. It concentrates on mathematical modeling of these phenomena taking into account recent progress in probabilistic models of computer vision and developments in machine learning, such as deep networks.
Prerequisites: AS.110.106 OR AS.110.108
Corequisites: NA
Instructor(s): A. Yuille
Area: Quantitative and Mathematical Sciences
NA.
The course gives an introduction to computational models of the mammalian visual cortex. It covers topics in low-, mid-, and high-level vision. It briefly discusses the relevant evidence from anatomy, electrophysiology, imaging (e.g., fMRI), and psychophysics. It concentrates on mathematical modelling of these phenomena taking into account recent progress in probabilistic models of computer vision and developments in machine learning, such as deep networks. Also offered as AS.050.375. Co-listed with Computer Science as EN.601.485.  
Prerequisites: NA  
Corequisites: NA  
Instructor(s): A. Yuille  
Area: NA  

AS.050.814. Research Seminar in Computer Vision. NA Credit.  
This seminar is based on topics in computational vision with the option of attending additional subgroup meetings on specific topics.  
Prerequisites: NA  
Corequisites: NA  
Instructor(s): A. Yuille  
Area: NA  
NA.

Physics Astronomy  
AS.171.205. Introduction to Practical Data Science: Beautiful Data. 3.0 Credits.  
The class will provide an overview of data science, with an introduction to basic statistical principles, databases, fundamentals of algorithms and data structures, followed by practical problems in data analytics. Recommend Course Background: Familiarity with principles of computing.  
Prerequisites: NA  
Corequisites: NA  
Instructor(s): S. Szalay  
Area: Natural Sciences, Quantitative and Mathematical Sciences  
NA.

Psychological Brain Sciences  
AS.200.313. Models of Mind and Brain. 3.0 Credits.  
This is a seminar surveying computational approaches to understanding mental and neural processes, including sensory and conceptual representation, categorization, learning and memory. The course will also develop familiarity with computational tools such as numerical simulation, linear transformation and data visualization. Enrollment limited to Juniors and Seniors. Recommended Course Background: AS.110.106 / Calculus I OR AS.110.108 Calculus I, AS.050.101 / Cognition OR AS.200.211 / Sensation & Perception OR AS.080.105 / Introduction to Neuroscience OR other introductory coursework in cognitive & neural sciences.  
Prerequisites: NA  
Corequisites: NA  
Instructor(s): C. Honey  
Area: Quantitative and Mathematical Sciences, Social and Behavioral Sciences  
NA.

General Engineering  
EN.500.112. Gateway Computing: JAVA. 3.0 Credits.  
This course introduces fundamental programming concepts and techniques, and is intended for all who plan to develop computational artifacts or intelligently deploy computational tools in their studies and careers. Topics covered include the design and implementation of algorithms using variables, control structures, arrays, functions, files, testing, debugging, and structured program design. Elements of object-oriented programming, algorithmic efficiency and data visualization are also introduced. Students deploy programming to develop working solutions that address problems in engineering, science and other areas of contemporary interest that vary from section to section. Course homework involves significant programming. Attendance and participation in class sessions are expected.  
Prerequisites: Students may not have earned credit in courses: EN.500.113 OR EN.500.114 OR EN.510.202 OR EN.520.123 OR EN.530.112 OR EN.580.200 OR EN.601.107.  
Corequisites: NA  
Instructor(s): I. Sekyonda; S. More  
Area: Engineering  
NA.

EN.500.132. Bootcamp: Java. 1.0 Credit.  
This on-line course provides students who have already achieved a basic understanding of programming and computational thinking in one programming language with an opportunity to apply these skills in another programming language. Students will be expected to complete projects to demonstrate proficiency in the new language. Satisfactory/unsatisfactory only.  
Prerequisites: Not open to students who have completed EN.601.107, EN.600.107, or EN.500.112;EN.500.113 OR EN.500.114 OR EN.510.202 OR EN.580.200 OR EN.530.112 OR EN.520.123  
Corequisites: NA  
Instructor(s): J. Kovba  
Area: Engineering  
NA.

EN.500.133. Bootcamp: Python. 1.0 Credit.  
This on-line course provides students who have already achieved a basic understanding of programming and computational thinking in one programming language with an opportunity to apply these skills in another programming language. Students will be expected to complete projects to demonstrate proficiency in the new language. Satisfactory/unsatisfactory only.  
Prerequisites: Not open to students who have completed EN.500.113 or EN.580.200;EN.500.112 OR EN.500.114 OR EN.601.107 OR EN.510.202 OR EN.530.112 OR EN.520.123  
Corequisites: NA  
Instructor(s): J. Kovba  
Area: Engineering  
NA.

EN.500.133. Bootcamp: Python. 1.0 Credit.  
This on-line course provides students who have already achieved a basic understanding of programming and computational thinking in one programming language with an opportunity to apply these skills in another programming language. Students will be expected to complete projects to demonstrate proficiency in the new language. Satisfactory/unsatisfactory only.  
Prerequisites: Not open to students who have completed EN.500.113 or EN.580.200;EN.500.112 OR EN.500.114 OR EN.601.107 OR EN.510.202 OR EN.530.112 OR EN.520.123  
Corequisites: NA  
Instructor(s): J. Kovba  
Area: Engineering  
NA.
EN.500.134. Bootcamp: MATLAB. 1.0 Credit.
This on-line course provides students who have already achieved a basic understanding of programming and computational thinking in one programming language with an opportunity to apply these skills in another programming language. Students will be expected to complete projects to demonstrate proficiency in the new language. Satisfactory/unsatisfactory only.
Prerequisites: Not open to students who have completed EN.500.114, EN.510.202, EN.520.123, EN.530.112, EN.540.111, EN.540.305, EN.580.200, or EN.553.281;EN.500.112 OR EN.500.113 OR EN.601.107
Corequisites: NA
Instructor(s): M. Rubanov
Area: Engineering
NA.

EN.500.745. Seminar in Computational Sensing and Robotics. 1.0 Credit.
Seminar series in robotics. Topics include: Medical robotics, including computer-integrated surgical systems and image-guided intervention. Sensor based robotics, including computer vision and biomedical image analysis. Algorithmic robotics, robot control and machine learning. Autonomous robotics for monitoring, exploration and manipulation with applications in home, environmental (land, sea, space), and defense areas. Bio robotics and neuro mechanics, including devices, algorithms and approaches to robotics inspired by principles in biomechanics and neuroscience. Human-machine systems, including haptic and visual feedback, human perception, cognition and decision making, and human-machine collaborative systems. Cross-listed Mechanical Engineering, Computer Science, Electrical and Computer Engineering, and Biomedical Engineering.
Prerequisites: NA
Corequisites: NA
Instructor(s): L. Whitcomb; P. Kazanzides
Area: NA
NA.

Electrical Computer Engineering

EN.520.434. Modern Biomedical Imaging Instrumentation and Techniques. 3.0 Credits.
An intermediate biomedical imaging course covering modern biomedical imaging instrumentation and techniques as applied to diagnostic radiology and other biomedical applications. It includes recent advances in various biomedical imaging modalities, multi-modality imaging and molecular imaging. The course is team taught by experts in the respective fields and provides a broad based knowledge of modern biomedical imaging to prepare students for graduate studies and research in biomedical imaging. Also, the course will offer tours and practical experience with modern biomedical imaging equipments in clinical and research settings. Co-listed with EN.580.473
Prerequisites: Students may not have taken EN.520.634
Corequisites: NA
Instructor(s): B. Tsui
Area: NA
NA.

EN.520.447. Information Theory. 3.0 Credits.
This course will address some basic scientific questions about systems that store or communicate information. Mathematical models will be developed for (1) the process of error-free data compression leading to the notion of entropy, (2) data (e.g. image) compression with slightly degraded reproduction leading to rate-distortion theory and (3) error-free communication of information over noisy channels leading to the notion of channel capacity. It will be shown how these quantitative measures of information have fundamental connections with statistical physics (thermodynamics), computer science (string complexity), economics (optimal portfolios), probability theory (large deviations), and statistics (Fisher information, hypothesis testing).
Prerequisites: EN.553.310 OR EN.553.420 OR EN.553.311;Students can earn credit for either EN.520.447 or EN.520.647, but not both.
Corequisites: NA
Instructor(s): J. Prince
Area: Engineering, Quantitative and Mathematical Sciences
NA.

EN.520.701. Current Topics in Language and Speech Processing. 1.0 Credit.
This biweekly seminar will cover a broad range of current research topics in human language technology, including automatic speech recognition, natural language processing and machine translation. The Tuesday seminars will feature distinguished invited speakers, while the Friday seminars will be given by participating students. A minimum of 75% attendance and active participation will be required to earn a passing grade. Grading will be S/U.
Prerequisites: NA
Corequisites: NA
Instructor(s): J. Trmal
Area: NA
NA.

EN.520.702. Current Topics in Language and Speech Processing. 1.0 Credit.
This biweekly seminar will cover a broad range of current research topics in human language technology, including automatic speech recognition, natural language processing and machine translation. The Tuesday seminars will feature distinguished invited speakers, while the Friday seminars will be given by participating students. A minimum of 75% attendance and active participation will be required to earn a passing grade. Cross-listed with Computer Science. Grading will be S/U.
Prerequisites: NA
Corequisites: NA
Instructor(s): J. Trmal
Area: Engineering
NA.
Mechanical Engineering
EN.530.707. Robot System Programming. 4.0 Credits.
This course seeks to introduce students to open-source software tools that are available today for building complex experimental and fieldable robotic systems. The course is grouped into sections, each of which building on the previous in increasing complexity and specificity: tools and frameworks supporting robotics research, robotics-specific software frameworks, integrating complete robotic systems, and culminates with an independent project of the student’s own design using small mobile robots or other robots in the lab. Students will need to provide a computer (with at least a few GB of memory and a several tens of GB of disc space) running Ubuntu (https://www.ubuntu.com or one of its variants such as Xubuntu) and ROS (http://ros.org/). Students should have an understanding of intermediate programming in C/C++ (including data structures and object-oriented programming). Familiarity with Linux programming. Familiarity with software version control systems such as Git, and linear algebra. Students should see the course homepage http://dscl.lcsr.jhu.edu/ME530707_2018 for more information and to get started with the course. Required Course Prerequisite/Corequisite: EN.530.646 and EN.600.436. Registration only by permission of the instructor. Please contact Prof. Louis Whitcomb at llw@jhu.edu. Please put ’530.707 Robot System Programming’ in the subject line of your email.
Prerequisites: NA
Corequisites: NA
Instructor(s): L. Whitcomb
Area: NA

Biomedical Engineering
EN.580.468. The Art of Data Science. 3.0 Credits.
In this course, we will cover the fundamentals of doing data science research, explaining “best practices” for each step, that collectively comprise an upward spiral. These steps include: (i) asking an interesting question, (ii) determining the degree to which the answer is known, (iii) assessing there currently exists data to likely obtain a satisfactory answer, (iv) exploring the data set, (v) cleaning up the dataset, (vi) formalizing a statistical inquiry, (vii) positing a statistical model which we hope will yield satisfactory answers, (viii) devising a test to assess the answer, (ix) building an estimator to assess the model, (x) checking the model, (xi) reporting the results, (xii) suggesting the next experiment to perform or question to answer to further enhance the model. Note that this course will largely be project based; each student will be expected to complete each of the above steps on some real data of interest to the student. Lectures will be minimal, giving introductory explanations one day, hopefully only part of the time. The rest of the time, we will work independently or in small groups to complete the weekly portion of the overall project. Please come ready to do science! If you don’t have questions that you want answered, you can work in small groups, but each student will need to write the code and reports on their own. Recommended Course Background: No courses are formally required, though students will need to write numerical code (in R, Python, or Julia), and make reports using LaTeX, knitr, or Jupyter notebooks or similar.
Prerequisites: NA
Corequisites: NA
Instructor(s): J. Vogelstein
Area: Engineering, Natural Sciences

EN.580.473. Modern Biomedical Imaging Instrumentation and Techniques. 3.0 Credits.
An intermediate biomedical imaging course covering modern biomedical imaging instrumentation and techniques as applied to diagnostic radiology and other biomedical applications. It includes recent advances in various biomedical imaging modalities, multi-modality imaging and molecular imaging. The course is team taught by experts in the respective fields and provides a broad based knowledge of modern biomedical imaging to prepare students for graduate studies and research in biomedical imaging. Also, the course will offer tours and practical experience with modern biomedical imaging equipment in clinical and research settings. Co-listed with EN.520.434 Recommended course background: EN.520.432 or EN.580.472
Prerequisites: Students may not have taken EN.580.773.
Corequisites: NA
Instructor(s): B. Tsui
Area: Engineering, Natural Sciences

EN.580.488. Foundations of Computational Biology and Bioinformatics. 4.0 Credits.
This course will introduce probabilistic modeling and information theory applied to biological sequence analysis, focusing on statistical models of protein families, alignment algorithms, and models of evolution. Topics will include probability theory, score matrices, hidden Markov models, maximum likelihood, expectation maximization and dynamic programming algorithms. Homework assignments will require programming in Python. Foundations of Computational Biology I is not a prereq. Recommended Course Background: Math through linear algebra and differential equations, EN.580.221 or equivalent, EN.601.226 or equivalent
Prerequisites: EN.601.220
Corequisites: NA
Instructor(s): R. Karchin
Area: Engineering, Natural Sciences
NA.
**EN.580.668. The Art of Data Science. 3.0 Credits.**

In this course, we will cover the fundamentals of doing data science research, explaining "best practices" for each step, that collectively comprise an upward spiral. These steps include: (i) asking an interesting question, (ii) determining the degree to which the answer is known, (iii) assessing there currently exists data to likely obtain a satisfactory answer, (iv) exploring the data set, (v) cleaning up the dataset, (vi) formalizing a statistical inquiry, (vii) positing a statistical model which we hope will yield satisfactory answers, (viii) devising a test to assess the answer, (ix) building an estimator to assess the model, (x) checking the model, (xi) reporting the results, (xii) suggesting the next experiment to perform or question to answer to further enhance the model. Note that this course will largely be project based; each student will be expected to complete each of the above steps on some real data of interest to the student. Lectures will be minimal, giving introductory explanations one day, hopefully only part of the time. The rest of the time, we will work independently or in small groups to complete the weekly portion of the overall project. Please come ready to do science! If you don't have questions that you want answered, you can work in small groups, but each student will need to write the code and reports on their own. Recommended background: No courses are formally required, though students will need to write numerical code (in R, Python, or Julia), and make reports using LaTeX, knitr, or Jupyter notebooks or similar.

**Prerequisites:** NA  
**Corequisites:** NA  
**Instructor(s):** J. Vogelstein  
**Area:** Engineering  
**NA.**

**EN.580.688. Foundations of Computational Biology and Bioinformatics. 4.0 Credits.**

This course will introduce probabilistic modeling and information theory applied to biological sequence analysis, focusing on statistical models of protein families, alignment algorithms, and models of evolution. Topics will include probability theory, score matrices, hidden Markov models, maximum likelihood, expectation maximization and dynamic programming algorithms. Homework assignments will require programming in Python. Recommended Course Background: Math through linear algebra and differential equations, EN.580.221 or equivalent, EN.601.226 or equivalent.

**Prerequisites:** NA  
**Corequisites:** NA  
**Instructor(s):** R. Karchin  
**Area:** NA  
**NA.**

**EN.580.745. Mathematics of Deep Learning. 1.5 Credits.**

The past few years have seen a dramatic increase in the performance of recognition systems thanks to the introduction of deep networks for representation learning. However, the mathematical reasons for this success remain elusive. For example, a key issue is that the training problem is nonconvex, hence optimization algorithms are not guaranteed to return a global minima. Another key issue is that while the size of deep networks is very large relative to the number of training examples, deep networks appear to generalize very well to unseen examples and new tasks. This course will overview recent work on the theory of deep learning that aims to understand the interplay between architecture design, regularization, generalization, and optimality properties of deep networks. Recommended background: machine learning (EN.601.475), deep learning (EN.520.438 or EN.601.482), graduate-level matrix analysis and linear algebra (EN.553.792) and graduate-level optimization (EN.553.762).

**Prerequisites:** NA  
**Corequisites:** NA  
**Instructor(s):** R. Vidal  
**Area:** NA  
**NA.**

**Information Security Institute**

**EN.650.624. Network Security. 3.0 Credits.**

This course focuses on advanced security topics and research in computer networks. It builds on the basic overview of network security covered in previous security courses. Beyond the basics of developing security network communications and applications, this advanced course dives deeper into the theory and practice behind network attack, the growing reality of weaponized zero-day vulnerabilities, and the current state-of-the-art responses. Course work includes reviewing contemporary security research papers, hands-on experiments in defending/attacking networks, and writing analyses.

**Prerequisites:** EN.650.424 OR EN.600.424 or permission of the instructor.  
**Corequisites:** NA  
**Instructor(s):** S. Nielson  
**Area:** Engineering  
**NA.**

**EN.650.724. Advanced Network Security. 3.0 Credits.**

This course focuses on advanced security topics and research in computer networks. It builds on the basic overview of network security covered in previous security courses. Beyond the basics of developing security network communications and applications, this advanced course dives deeper into the theory and practice behind network attack, the growing reality of weaponized zero-day vulnerabilities, and the current state-of-the-art responses. Course work includes reviewing contemporary security research papers, hands-on experiments in defending/attacking networks, and writing analyses.

**Prerequisites:** EN.601.644 or permission of the instructor.  
**Corequisites:** NA  
**Instructor(s):** S. Nielson  
**Area:** Engineering  
**NA.**