**Mission Statement:**

The mission of the Yeshiva College computer science program is to prepare students for employment in various fields of computer science and/or to pursue advanced studies in computer science. All tracks offered by the department cover the fundamentals of computer science theory and practice. Students who complete any of the tracks in the program should be well versed in the architectures of software and hardware systems, the construction and use of data structures and algorithms for the solution of computing problems, and the tools and practices needed to write high quality software. The department offers bachelors of science (B.Sc.) tracks that are designed to help students achieve an advanced level of expertise in a subspecialty of computer science. Students who complete one of the B.Sc. tracks should achieve a level of specialized expertise that should be sufficient to help them succeed at the highest levels of industry throughout their careers.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Program Goal** | **Objective** | **1300** | **1320** | **1504** | **2545** | **2546** | **3640** | **2113** | **3610** | **4580** | **3645** | **3800** | **3810** | **3820** | **3910** | **3920** | **3563** | **3580** |
| Students will be able to analyze real-world requirements/problems and identify, analyze, use, design, and implement software that includes appropriate abstractions and algorithms to meet the requirements, while being able to prove the software’s performance and correctness across a variety of metrics (e.g., time, space, parallel vs. sequential implementation) | Students are able to apply computational thinking to map problems, and associated solution requirements, into programs that can be implemented and solved on a computers | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Students are able to write programs that meet a set of requirements | **X** | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Students will be able to analyze a computational problem from an algorithmic perspective, and identify and employ appropriate algorithms and data-structures to solve these problems |  |  |  | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Students will be able to analyze the complexity of algorithms as well as be able to apply different algorithm design strategies and select the appropriate strategy for a range of problems |  |  | **X** | **X** | **X** |  |  |  |  |  |  |  |  |  |  |  |  |
| Students will be able to write complete, high-quality software solutions using a range of software engineering best practices, languages, and tools | Students are able to test and debug small scale programs | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Students will be able to apply software engineering skills in order to build large software systems |  | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Students will be able to implement programs that solve real-world problems making appropriate use of data structures and algorithms |  | **X** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Students can make well informed choices when deciding what language, and language features, to use when writing a given program to solve a given problem |  |  |  |  |  | **X** |  |  |  |  |  |  |  |  |  |  |  |
| Students can write programs in a range of languages and using a range of language features |  |  |  |  |  | **X** |  |  |  |  |  |  |  |  |  |  |  |
| Students will be able to write software which leverages the strengths and takes into account the pitfalls present at multiple layers of the software stack (system, operating system, and application layers) | Students understand how code gets transformed into machine instructions, and can apply that knowledge to make good software design choices |  |  |  |  |  |  | **X** |  |  | **X** |  |  |  |  |  |  |  |
| Students understand the strengths, weaknesses, and impacts of caches, memory management, process management, and file systems on software |  |  |  |  |  |  | **X** | **X** |  | **X** |  |  |  |  |  |  |  |
| Students are able to write system level software that uses or replicates (as appropriate) common features of modern operating systems |  |  |  |  |  |  |  | **X** |  |  |  |  |  |  |  |  |  |
| Students understand the behavior and architecture of the networks protocols that make up the internet |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Students will be able to write secure software which takes into account potential security threats at multiple layers of the software stack (system, operating system, and application layers) | Students will understand security vulnerabilities of X86\_64 processors and C programs |  |  |  |  |  |  | **X** |  |  |  |  |  |  |  |  |  |  |
| Students understand, and can configure and use, the security-related features of modern operating systems |  |  |  |  |  |  |  | **X** |  |  |  |  |  |  |  |  |  |
| Students will be able to write secure software, as well as revise poorly written software to mitigate common security vulnerabilities |  |  |  |  |  |  |  |  | **X** |  |  |  |  |  |  |  |  |
| Students understand, and will be able to function effectively, both on the offensive (“Black Hat”) and defensive (“White Hat”) sides of Cybersecurity |  |  |  |  |  |  |  |  | **X** |  |  |  |  |  |  |  |  |
| Students will be able to both write and use high quality modern software | Students are able to write and debug properly synchronized multi-threaded code at both the process and thread levels in Linux, preventing both deadlock and starvation |  |  |  |  |  |  |  | **X** |  |  |  |  |  |  |  |  |  |
| Students understand Machine Learning such and can use it when building software applications |  |  |  |  |  |  |  |  |  |  |  |  |  | **X** | **X** |  |  |
| Students will understand and be able to apply a chosen combination of the following: 1) distributed systems 2) database technologies 2) parallel programming |  |  |  |  |  |  |  |  |  |  | **X** | **X** | **X** |  |  | **X** | **X** |