Minor in Computer Science

23 units

A minor in computer science equips mathematically minded students specializing in computer programming. The minor comprises a fundamental understanding of the use, knowledge, function, installation, programming, and maintenance of computers, and provides graduates with a variety of technological skills needed in today's workplace.

Requirements

Please consult with the department for each semester's offerings since courses are not necessarily offered every semester.

Computer Science students are required to have a laptop for classroom work.

Code	Title	Units
CS/ENGR 120	Introduction to Computer Science I [1]	4
CS/ENGR 125	Introduction to Computer Science II	4
CS/ENGR 160	Discrete Structures [2]	3
CS/ENGR 260	Algorithms and Data Structures	3
MATH 165	Calculus I	3
Computer Science Minor Electives		6
Select two of the following:		
CS 230	Systems Programming and Operating Systems	
CS 290	Database Management Systems [1]	
CS 315	Fundamentals of Network Administration	
CS 325	Telecommunications and Interfacing	
CS 360	Computer Architecture and Organization	
CS 363	Web Programming	
CS 430	Artificial Intelligence	
CS 440	Mobile App Development	
CS/ENGR 452	Internet of Things	
CS 495	Topics in Computer Science	
Total Units		**23**

1 Meets 1 unit of the General Education Oral Communication requirement (taking CS 120, CS 290, and CS 480—or CS 120, ENGR 240, and ENGR 480—satisfies the General Education Oral Communication requirement).

2 MATH 280 may be substituted for CS 160.

There are a number of benefits to adding a minor in computer science to related fields such as mathematics. Students should consult their department advisor or an advisor in computer science to determine how adding a computer science minor might further their educational or professional goals.

Program Learning Outcomes

Program Learning Outcomes

Students who successfully complete this program shall be able to:

1. Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions.
2. Design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline.
3. Apply computer science theory and software development fundamentals to produce computing-based solutions.