HARVEY MUDD COLLEGE

CS005x: CS FOR ALL

INTRODUCTION TO COMPUTER SCIENCE AND PYTHON PROGRAMMING

Course Information and Syllabus

COURSE AIMS AND OBJECTIVES

Welcome to CS 5! This course is designed to give you a broad introduction to the exciting, and sometimes misunderstood, field of computer science.

This course has two central aims, each with a number of associated objectives:

Aim 1: To give students the tools to take a computational problem through the process of design, implementation, documentation, and testing.

Objectives:

- Break a broad problem down into specific sub-problems
- Write an algorithm to solve a specific problem, and then translate that algorithm into a program in a specific programming language (Python)
- Write clear, concise documentation
- Develop test cases that reveal programming bugs

Aim 2: To give students an understanding of the breadth of Computer Science as a discipline and how it exists in the world.

Objectives:

- Identify applications of computer science in society
- Describe the big questions in computer science
- Describe the relationship between a number of major sub-disciplines within computer science, including functional and imperative programming, computer architecture, and theoretical computer science

Based on these objectives, the curriculum is driven by exercises that tie the skills of programming and design directly into relevant applications in computer science.

The only prerequisite is a strong foundation in Algebra; everything else will be explained in the course.

WHY PYTHON?

This course teaches the skills of Python programming as a way to accomplish the aims and objectives above. We believe Python is a great language to start to learn to program; it can feel a lot like writing instructions in English, and therefore is often quick to learn, but is also popularly used by many computer science professionals.

Programming, and Python programming in particular, is a valuable tool that computer scientists use, but it is only one facet of the broad and exciting field of CS. We will be tackling much more than the basics of programming in this course, and for several assignments will stray completely from Python in order to explore deeper concepts of computer science.

COURSE STRUCTURE

While students may take this course at any pace, it has been designed to last 14 weeks. Each week of content includes the following:

- **Lessons** – 1-5 short informational videos describing key concepts for the week, created by Harvey Mudd undergraduate T.A.s. These only appear in the first part of the course, as lesson content is incorporated in homework assignments after the midterm. These can be supplemented through use of the CS for All textbook, which is available as a link above the courseware and can be downloaded.

- **Homework** – *The core of the coursework:* 2-6 longer problems building on the lecture content, usually using Python programming.
    - **Readings** within homework assignments provide an opportunity to reflect on and respond to articles explaining applications of course concepts. These are graded by self-assessment.
    - **Programming Assignments** will usually be auto-graded by copying and pasting your code solutions into a window provided at the bottom of the page. Submissions are limited in order to avoid overburdening the autograder; please be cautious to debug code before submitting.
- **Discussions** – opportunities to share questions about key concepts, homework assignments, and more.

The core topics of each week are below.

Week 1	Introduction to computation: CS, Python, and Picobot
Week 2	From data to information: strings, structures, and slicing
Week 3	CS's fundamental building blocks: functions
Week 4	Self-similarity as design strategy: recursion
Week 5	Top-down vs. bottom-up problem solving: higher-level functions
Week 6	Computation's physical building blocks: circuit design
Week 7	The nonliving world's native tongue: assembly language
Week 8	Iteration in Python: repetition through loops **Midterm Exam**
Week 9	Serious structures: nested loops and dictionaries
Week 10	Self-defined structures: objects and classes
Week 11	Piecing everything together: large-scale problem solving
Week 12	Theoretical CS: state-machines and uncomputable functions **Final Project & Final Exam**

Exams: The exams resemble homework assignments in many ways – they will primarily consist of Python programming exercises, though these will be designed to be completed fairly quickly. Please note in the Grading section that these are weighted

more heavily than homework assignments, and have fewer chances to submit code; make sure your code works in Trinket or IDLE before submitting.

Final Project: You will perform a self-assessed

As this is a self-paced course, the choice of using one week per unit is recommended, but not required. Some assignments may take longer than others, so it may be worthwhile to give a bit of extra time for some units.

GRADING

Grades are determined by the following:

- **Homework (50%)** (12 assignments total; 2 dropped*)
  *Two assignments require downloadable software. Those unable to download additional software may wish to drop these assignments.
- **Midterm Exam (15%)**
- **Final Project (15%)**
- **Final Exam (20%)**

To earn a certificate in this course (honor code or verified), students must earn a minimum grade of 60% in the course by the course end date: October 4th, 2015, at 23:59 UTC. This is the only deadline for the course; all work may be submitted at any time before that date to be factored into the grade.

IMPORTANT DATES

This course is self-paced. We recommend that the course be taken over a period of about fourteen weeks; however, the course may be started at any time after the start date of **June 2, 2015, 0:00 UTC** and pursued at any pace.

Again, if you wish to receive a certificate in this course, a minimum grade of **60%** must be achieved by the course's end date of **October 4th, 2015, 23:59 UTC**. Those looking to complete a *Verified Certificate* will need to upgrade by **September 25th, 2015, 23:59 UTC.**

The courseware will still be available after the course end date; any registered users of the course should be able to access the archived courseware indefinitely, including videos, assignments, and exams. Discussion boards will not be moderated after the course end date.