CSCI 161: Introduction to Computer Science

Course Syllabus – Fall 2021

Instructor: David Chiu
Email: dchiu@pugetsound.edu
Course Page: https://canvas.pugetsound.edu/courses/6520
Office Hours (available in office and zoom):

- Mon/Wed/Fri 9:00am - 10:00am
- Tues 2:00pm - 4:00pm
- By appointment

Lecture Meeting Times: Mon/Wed/Fri 11:00pm - 11:50pm in TH 409
Lab Meeting Times: Thu 1:00pm - 2:50pm in TH 409
Final Exam: Wed, Dec 15, 12:00pm - 2:00pm

1 Course Information

This course is an introduction to computer science and programming. The programming language Java is used to illustrate concepts in computer science. The course emphasizes the use of the computer as a problem-solving tool and the development of good programming style. CS 161 is the introductory course for students planning to major or minor in computer science. A weekly lab is required.

Prerequisites

Three years of high-school mathematics, MATH 110 Pre-Calculus, or equivalent. Students with transfer credit for CSCI 161 may not take this course.

Course Topics

- Object-oriented programming
- Arithmetic manipulation
- Conditional logic
- Loops and recursion
- Methods
- Basic data structures
- Input/output: file and user interaction
- Elementary complexity analysis

2 Grading

The following grade cutoffs are upper bounds. They might come down, but will not be set higher: A = 95, A- = 90, B+ = 87, B = 83, B- = 80, C+ = 77, C = 73, C- = 70, D+ = 67, D = 64, D- = 60, F = $<$ 60. Your overall grade will be composed as follows:

	% Weight
Participation	5
Lab Assignments	10
Homework Assignments	35
Midterm I	15
Midterm II	15
Final Exam	20

Table 1: Breakdown of Grades

Assignments

- **Lab Assignments (Paired)** – You will pair up with another student and switch roles throughout the lab. Lab assignments are downloadable from the class page and are always due the Friday after lab at 23:59, unless stated otherwise. Labs are graded on a 10/5/0 point scale. 10 = complete, 5 = some effort made, 0 = minimal effort or no attendance.
- **Homework Assignments (Work Alone!)** – You will work alone on all homework assignments. Collaboration among students is encouraged for problem interpretation, brainstorming, etc., but in general, I expect every student to submit their own work. **Do not help write or share each other's programs!** Code duplications are *extremely easy* to catch, and the penalty for cheating is severe: course failure and a permanent record on your transcript.
- **Late Work** – For each day either a homework or project assignment is late (includes weekends), a 10% deduction will be assessed, and no late work will be accepted one week after the due date.

Exams

There will be two midterms and a final exam. They will cover topics discussed in the lectures, readings from the assigned textbook, and assignments. The exams are cumulative. Study guides are provided and selected problems are reviewed on the lecture preceding the day of the exam. You are allowed a calculator and a page of notes (front and back) on all exams.

Discretionary

Discretionary points will be given based on:

- Lecture and lab attendance
- Classroom participation
- Timeliness
- Refraining from activities that can disrupt others, *e.g.*, texting, playing games on your laptop, *etc*.

3 Community Statement

Students taking this course range from those with no prior background in programming to those with substantial experience and prior coursework. Because this course is an introduction to computer science, I do not expect students to have had any prior experience in programming. The course will work best if we respect and welcome each other no matter what level of "readiness" we are at, and we all support one another in learning. I will not tolerate behaviors that could negatively affect another students classroom experience. Such behaviors might include: making rude or condescending comments, snickering at others questions or comments, talking over other individuals, and so on. I reserve the right to withdraw a student from this class who is repeatedly exhibiting such behavior.

The goals of this course can only be accomplished in a setting of mutual respect, where ideas, questions, and misconceptions can be discussed with civility. As your instructor, I am committed to creating a classroom environment that welcomes all students, regardless of their identities (e.g., race, class, gender, sexual orientation, religious beliefs). I firmly believe that everyone in the class is fully capable of engaging and grasping the material, and that the world of computing is stronger when it includes the broadest possible set of perspectives. We all have unconscious biases, and I will try to continually examine my judgments, words, and actions to keep my biases in check and treat everyone fairly. I hope that you will do the same. If you feel comfortable, please let me know if there is anything I can do to make sure everyone is encouraged to succeed in this class.

4 Course Policies

Laptops and Phones

Laptops: Laptop computers have proved to be a distraction during my lectures. Except on lab days and for those who can provide documentation of need from the office of Student Accessibility and Accommodation (SAA), please don't bring your laptops to lectures. **Phones:** Please put your phones on silent during class.

Academic Integrity

You should be aware of the *Student Integrity Code* at the university. Any suspected cheating (*e.g.*, plagiarizing code, copying homework solutions, etc.) will be reported to the Registrar, which may result in possible suspension/expulsion. See this link for more info:
http://www.pugetsound.edu/student-life/personal-safety/student-handbook/academic-handbook/academic-integrity

Student Accessibility and Accommodation

If you have a physical, psychological, medical or learning disability that may impact your course work, please contact Peggy Perno, Director of the Office of Accessibility and Accommodation, 105 Howarth, 253.879.3395. She will determine with you what accommodations are necessary and appropriate. All information and documentation is confidential.

Classroom Emergency Response Guidance

Please review university emergency preparedness and response procedures posted at . There is a link on the university home page. Familiarize yourself with hall exit doors and the designated gathering area for your class and laboratory buildings.

If building evacuation becomes necessary (*e.g.* earthquake), meet your instructor at the designated gathering area so she/he can account for your presence. Then wait for further instructions. Do not return to the building or classroom until advised by a university emergency response representative.

If confronted by an act of violence, be prepared to make quick decisions to protect your safety. Flee the area by running away from the source of danger if you can safely do so. If this is not possible, shelter in place by securing classroom or lab doors and windows, closing blinds, and turning off room lights. Lie on the floor out of sight and away from windows and doors. Place cell phones or pagers on vibrate so that you can receive messages quietly. Wait for further instructions.

Student Bereavement Policy

The University of Puget Sound recognizes that a time of bereavement can be difficult for a student. Therefore, the university provides a Student Bereavement Policy for students facing the loss of a family member. Students are normally eligible for, and faculty members are expected to grant, three consecutive weekdays of excused absences, without penalty, for the death of a family member, including parent, grandparent, sibling, or persons living in the same household. Should the student feel that additional days are necessary, the student must request additional bereavement leave from the Dean of Students or the Dean's designee. In the event of the death of another family member or friend not explicitly included within this policy, a bereaved student may petition for grief absence through the Dean of Students office for approval.

5 Course Schedule

The following course schedule is tentative and subject to change.

Week	Topics	Reading
1	Computer science and algorithms	Notes provided
2	Classes and objects, variables and types	BlueJ: Chap 1-2
3	Writing methods and classes	Chap 3
4	Conditionals	BlueJ: Chap 3
5	Typing: primitives vs. classes	BlueJ: Chap 3-4
6	*Review and Midterm I*	—
7	Loops	BlueJ: Chap 4
8	Arrays	Notes provided
9	ArrayList and Scanner	BlueJ: Chap 4
10	HashMap and File I/O	BlueJ: Chap 6
11	*Review and Midterm II*	—
12	Basic complexity analysis	Notes provided
13	Sorting	Notes provided
14	Recursion	Notes provided
15	*Review and reading period*	—

String API (Note: Strings are *immutable* objects, unlike most others)

Method	Description
`public char charAt(int position)`	Returns the character at the specified position
`public boolean equals(String other)`	Checks content equality with the given String
`public boolean equalsIgnoreCase(String other)`	Same as above, but case-insensitive
`public int indexOf(String str)`	Returns starting position of the search string `str` if found, or -1 if not found
`public int length()`	Gets the length of the String
`public String replace(String target, String rep)`	Returns this String resulting from replacing `target` with `rep`
`public String substring(int begin, int end)`	Returns this String starting from position `begin`, ending at position `end` -1
`public String toLowerCase()`	Returns this String in lower case
`public String toUpperCase()`	Returns this String in upper case
`public String[] split(String delimiter)`	Splits the string by the `delimiter` and returns an array containing the splits

ArrayList API

Method	Description
`public ArrayList<E>()`	Constructor. Creates a new ArrayList that holds objects of type E
`public boolean add(E e)`	Inserts element e to end of this list
`public boolean add(int index, E e)`	Inserts element e at the specified position in the list
`public void clear()`	Removes all elements from list
`public boolean contains(E e)`	Searches for element e in list, returns true if found, false otherwise
`public E get(int index)`	Returns the element at given index
`public int indexOf(E e)`	Searches for e in the list, returns the index of the first occurrence if found, or -1 if not found.
`public E remove(int index)`	Removes the element at given index. Returns the deleted element
`public E set(int index, E e)`	Replaces the element at specified index.
`public int size()`	Returns the number elements in the current list
`public String toString()`	Returns the String representation of the current list

Scanner API

Method	Description
`public Scanner(File source)`	Constructor. Creates a scanner on the given file.
`public Scanner(InputStream source)`	Constructor. Creates a scanner on the given stream.
`public boolean hasNext()`	Returns true if the scanner has another token.
`public String nextLine()`	Returns an entire line of input as a String.
`public void close()`	Closes the scanner.

PrintWriter API

Method	Description
`public PrintWriter(File destination)`	Constructor. Opens the given file for writing.
`public void close()`	Closes the file after you're done writing to it. **(This is a must after you're done writing!)**
`public void print(String s)`	Prints the given string to the opened file
`public void println(String s)`	Prints the given string and a newline to the opened file

HashMap API

`public V put(K key, V value)`	Associates given value with given key in the map
`public void clear()`	Removes all elements from list
`public boolean containsKey(K key)`	Searches for the given key in the map
`public boolean containsValue(V value)`	Searches for the given value in the map
`public V get(K key)`	Returns the element at given key, or null if no such key exists
`public V remove(K key)`	Removes the element at given key. Returns the deleted element
`public Set<K> keySet()`	Returns the set of keys in the map
`public int size()`	Returns the number elements in the current map