SABANCI UNIVERSITY
IF 100 - Computational Approaches to Problem Solving
Spring 2017-2018 / Top Hat 640093

Instructors

Duygu Karaoğlan Altop, FENS G001-C, duygukaltop@sabanciuniv.edu
İnanç Arın, FENS G001-C, inancarin@sabanciuniv.edu

Course Description

The course is an introduction to the key concepts in computational thinking such as algorithmic thinking, abstraction and decomposition. The students will also gain basic programming skills in order to apply computational thinking concepts in practice.

Through the lectures, homeworks, and interactive recitations specific to different disciplines, the students will learn how to design algorithms, how to divide a problem into subproblems, and how to build a solution by means of compositions. Evaluation of the solutions in terms of correctness and efficiency will also be covered.

In order to enable students apply computational thinking skills in practice, basic programming concepts, such as variables, statements, conditionals, iteration, and functions will be introduced by using a simple programming language such as Python.

Grading Policy

Small Homeworks (Exercises)[1]	4%
Regular Homeworks[2]	14%
In-class Assignments[3]	12%
Midterm Exam	30%
Final Exam	40%

Please note that the weighted average is not the only criterion in letter grading; exam average may also be taken into consideration.

[1] There will be 8-12 small homeworks. Randomly selected 4 homeworks will be graded.
[2] There will be 4 regular homeworks.
[3] There will be 5-6 in-class assignments, best 4 in-class assignment grades will be considered.

NA Policy

Students who fail to take the midterm and/or the final exam will get NA grade at the end of the semester.

Average of each of the grading items must be greater than 0 (zero). In other words, you have to attend at least 1 in-class assignment, you have to submit at least 1 meaningful small homework that is to be graded and you have to submit at least 1 meaningful regular homework. Otherwise, you will get NA grade.

Make-up Policy

No make-ups are allowed for in-class and small/regular homework assignments. Students automatically take zero from the respective assignments' grade if any of these assignments are missed.

Make-ups are only allowed for the midterm and final examinations to those with an official report from the University Health Center and to those with an official permission notice from the university for participation in a university event on the date of the exam in question. Students must submit their reports/notices to one of the instructors. The ones having other excuses should contact to the instructors within the day of the exam to be missed and then the instructors will decide whether these students are allowed to take the make-up exam or not. No exceptions will be made to this rule. Any excuses that will be taken into instructors' account after the exam will not be considered.

The dates of the make-up examinations will be announced later. ***The make-up examinations will be written and/or oral***. The details will be announced prior to the corresponding examination.

Please note that the make-up exams are expected to be ***much harder*** *than the regular exams, due to fairness reasons.*

Objection Policy

Students are allowed to object to their midterm and final examinations, as well as their in-class and small/regular homework assignments. Grade bargaining is absolutely not tolerated.

Plagiarism Policy (Academic Integrity)

Although we encourage the students to work and study together; exams, in-class and small/regular homework assignments are expected to be students' own works. Students need to understand the **difference between *helping and cheating*** concepts. You may share your ideas and knowledge, but you should not (and never) share your script or code. Allowing friends to copy part of an exam or an assignment from your work is not helping. In such a case, both parties will be considered as submitting a plagiarised work, and ***such behaviour will have disciplinary consequences for all parties involved***. Additionally, it is the student's responsibility to make sure that the assignment in question is never in publicly accessible locations.

If the submitted assignment (either in-class or small/regular homework) is not done by the student himself/herself, (s)he will get **0** (zero) from that particular assignment. If a student repeats it again, (s)he will ***fail the class***. No exceptions will be made to this rule.

It is the student's responsibility to ensure that (s)he completely understands any material that (s)he submits and that (s)he is actively engaged in the production of the solution. The instructors, TAs and LAs of this course reserve the right to ask the students to explain the reasoning behind their work without the presence of any collaborators. The students should know that the ***written submitted work is not the only material that will be graded***. The instructors, TAs or LAs might **request a viva** (oral exam), and **grade it instead of the written submitted work**.

Additionally, cases of plagiarism will be directly referred to the Dean's Office for disciplinary action. This course does not tolerate any breach of academic integrity (more info on http://www.sabanciuniv.edu/en/academic-integrity-statement).

Course Components

Lectures: Lectures are the main building blocks of this course. The content delivered in the lectures will be tested in the midterm and final examinations. Attendance is not mandatory but the students are recommended and encouraged to attend to the lectures for their own good.

Recitation Sessions: Recitation sessions are student-centered learning hours conducted by the TAs and the LAs of this course. Students must participate their own sections. There will be a total of twelve regular and two exam preparation recitation sessions. Every recitation session will consist of two hours. Each one of the exam preparation recitation sessions will be held one week prior to the related examination. In those recitation sections, TAs/LAs will be solving some sample questions to help students get ready for the examinations. Students are expected to bring their laptops to all recitations. Recitations are quite important to practice the topics taught in lectures. We strongly recommend students to participate the recitations for understanding/practicing topics in a better way.

Small/Regular Homeworks: Students are expected to do all of the assigned homeworks. Homework assignments are designed to bring the students the ability of transferring their knowledge into practice. The only way for the students to submit their homeworks is through SuCourse. Submissions by any other means will not be accepted. If a student fails to submit his/her homework via SuCourse on time, the corresponding homework will be graded as zero. The deadline for the homework assignments (except for the "Homework 0" which will not be graded) will always be by Wednesday 23:55 of the corresponding week. Late submissions are not allowed. The answer sheets for the homeworks will be made available to the students by 23:59 of the submission day, at the latest. Besides, successful submission is one of the requirements of the homeworks. If, for some reason, students cannot successfully submit their homeworks, the corresponding homework will be graded as zero. Our experiences show that doing homeworks with a good understanding is the best way to prepare for the midterm and final examinations.

There will be 8-12 small homeworks; 4 of them will be randomly selected and graded. On the other hand, there will be 4 regular homeworks and each of them will be graded.

In-class Assignments: In-class assignments will take exactly 10 minutes. Students who turn up late to the class, i.e. after the in-class assignment starts, can take the in-class assignment but they won't be given any extra time to finish their work. Additionally, every student must take the in-class assignments on their own recitation sections; otherwise his/her grade will be 0 (zero) regardless of what the provided answers deserve. No exceptions will be made to this rule.

There will be 5-6 in-class assignments in total, and the best 4 of these in-class assignment grades will be considered.

Midterm and Final Examinations: There will be only one midterm and one final examinations in this course. Students will be graded based on their comprehension of the content and their ability to transfer their knowledge gathered from the lectures and recitations into practice. The examination dates will be announced later.

Additional Notes

Students are responsible for every announcement made in lecture/SuCourse or sent via e-mail. Students are expected to check their Sabanci University mail inboxes regularly as important announcements will be sent them via e-mail. Not attending the class, not following SUCourse, not checking e-mails regularly is not an excuse, in case they miss something.

Tentative Course Outline

Introduction & Motivation We will introduce the course to the students and talk about our motivations. The reason behind this course being a university course (not for only CS students, but everyone) will be discussed and the importance of this course will be emphasized. Then, we will start with the general idea of computational thinking and provide some examples from different disciplines.

From Puzzles to Real World Problems Computational thinking includes 4 important concepts: (i) decomposition, (ii) pattern recognition, (iii) abstraction, and (iv) algorithm design. We will explain these concepts in detail. It is very important to understand the features of each aspect and their roles in solving a problem computationally. Several examples will be provided for these aspects, from different disciplines and daily life.

There are 10 types of people; those who understand binary and those who don't We will talk about computer components, data representation and basic information on data storage/retrieval. These will be a good practice for the "Abstraction" aspect of computational thinking. Additionally, Python will be mentioned for the first time to be used in the following weeks. We will be using Python to realize each one of the aspects of computational thinking separately in different topics.

Let the Coding Begin We will introduce the first Python program to the students. Syntax, variables, operators, standard input/outputs, basic problems with simple calculations, strings and lists will be covered. Besides, we will discuss using different libraries, especially for graph drawing purposes.

To Be or Not To Be: Making Decisions Some problems can be solved with basic operators and calculations, however some problems need decision making. Decision making is required when we want to execute a code only if a certain condition is satisfied, which is examined through Boolean expressions. Students will learn to determine which action to take and which statements to execute depending on the outcome. Decision making is a crucial point of "Algorithm Design", thus various examples from different disciplines will be covered.

Civilization with Modularity A function is a block of organized, reusable code that is used to perform a single, related action. Functions provide better modularity for our applications and a high degree of code reusing. In this week, students will learn to implement their own "user-defined" functions. Functions will be a good practice for decomposition and abstraction aspects of computational thinking. Additionally, we will discuss the scope of the defined and referred variables.

String Like a Bee Strings are amongst the most popular types in Python. We will discuss about using some of Python's built-in methods in order to be able to manipulate strings. It will be a good practice for the "Abstraction" aspect of computational thinking.

The Copying Game In general, statements are executed sequentially. However, some statements may be executed depending on a decision, and there may also be a situation when a block of code needs to be executed several times. A loop statement allows us to execute a statement or group of statements multiple times. Loops are also a crucial point of "Algorithm Design", thus various examples from different disciplines will be covered.

Let’s Get Organized The most basic data structure in Python is the sequence. Each element of a sequence is assigned a number - its position or index. There are certain things you can do with all sequence types. These operations include indexing, slicing, adding, multiplying, and checking for membership. Some built-in functions of lists as finding the length of a sequence or finding its largest /smallest elements will also be covered. Additionally, tuples and its differences will be mentioned. Besides, Python dictionary is a container of the unordered set of objects. It is aimed to show the role of the dictionaries in specific problems. Data structures have an important role for the abstraction of problems, thus should be emphasized under Computational Thinking approaches.

Verba Volant Scripta Manent (The faintest ink is more powerful than the strongest memory) Sometimes we need to read/write from/to files. Python provides basic functions and methods necessary to manipulate files by default. Students will learn these details.

Winter is Coming Some futuristic and eye-opening topics will be covered.