CS201 – Spring 2015-2016 - Sabancı University

Homework #3: Mirror Mirror

Due March 9, Wednesday, 19:00 (Sharp Deadline)

Brief Description

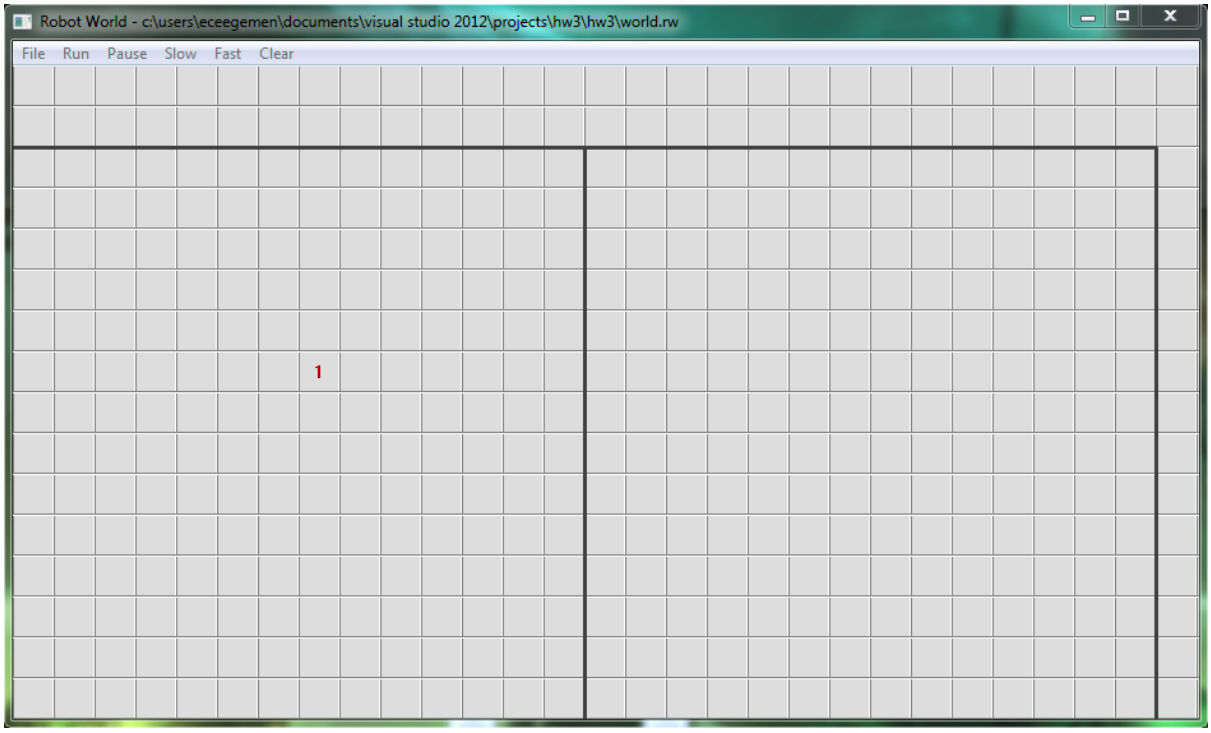
In this homework, you will write a robot application that uses the Robot class discussed in the lecture. This class is also described in a document called "RobotWorld.pdf" that is in the robot.zip package available in CS201 website. Please examine the RobotWorld.pdf file and the example programs. You will practice on the robot class during recitations this week. In order to do this homework, it is to your benefit to attend recitations because Robot World is something quite different considering what you have learned until now.

The C++ program that you are going to write will create a robot (the player robot) in a location of which the x and y coordinates are inputs. Then, your program will create a reflection robot in the symmetrical position according to the line (mirror) in the robot world. Then, the player robot will visit a target cell at the following coordinates: (7, 8). Reflection robot should follow your player robot's movements symmetrically. Reflection robot's movements will be explained in a later section. Other details related to initial orientation, input checks etc. will also be explained later in the document.

Details and Rules

First of all, the position of the player robot will be entered by the user. The coordinates must be non-negative integers between 0 and 13 inclusively. Otherwise, your program should display an error message immediately and should not prompt for another input (and no robot should be created).

The world is a rectangle (28x14) and it is divided into two squares (14x14) with a line shown in the figure below. The reflection robot should be created in the right square and it should be symmetric of the player robot in the left square with respect to the dividing line in the middle of two squares.



The initial orientation (the direction that the robot faces when created) is NOT an input and it should be east for the player robot and west for the reflection robot. Remember that the default value for the direction of the robot is east. The player robot should be created **facing east** and the reflection robot should be created **facing west**. After creating both robots, your program should set colors of both robots; the color of the player robot must be set to **blue**, and the color of the reflection robot must be set to **red**.

After both robots are created and their colors are set, the player robot will travel to the target cell. The reflection robot will mimic its movements symmetrically. In order to travel to the target cell, the player robot need to make two moves; horizontal and vertical. You may choose your player robot to begin with doing either of them. The reflection robot moves when the player robot moves, such that its movement is symmetric with respect to the mirror wall. You may refer to the following table for its movements.

Player	Reflection
East	West
West	East
North	North
South	South

Keep in the mind that the robots can't move at the same time. The player robot should move towards a direction first, and then the reflection robot should make its mimic movement before the player robot makes its next movement.

After all movements have been completed, two robots should swap colors, meaning that the color of the player robot will be set to red, and the color of the reflection robot will be set to blue.

The Program Flow

- Ask the user to input x coordinate of the player robot. Check the input value according to the following rule:
    - x-coordinate of the robot: $0 \leq x < 14$

  If the value is correct, proceed, otherwise terminate the program. Inputs are assumed to be integers.
- Ask the user to input y coordinate of the player robot. Check the input value according to the following rule:
    - y-coordinate of the robot: $0 \leq y < 14$

  If the value is correct, proceed, otherwise terminate the program. Inputs are assumed to be integers.
- Create the player robot at the given coordinates, facing east.
- Create the reflection robot at the symmetrical position according to the mirror, facing west.
- Set their colors; the player robot should be blue, the reflection robot should be red.
- The player robot will travel to the target cell (7, 8) making horizontal and vertical moves. Remember that, the reflection robot should mimic its movements symmetrically with respect to the mirror line, after each move.
- Swap their colors; the player robot should be red, the reflection robot should be blue.

Please keep in mind that, if the given coordinates for the player robot are the same with the target cell's, both robots should be created and even though they will not need to move in this special case, they are still required to change their colors according to the given steps.

IMPORTANT!

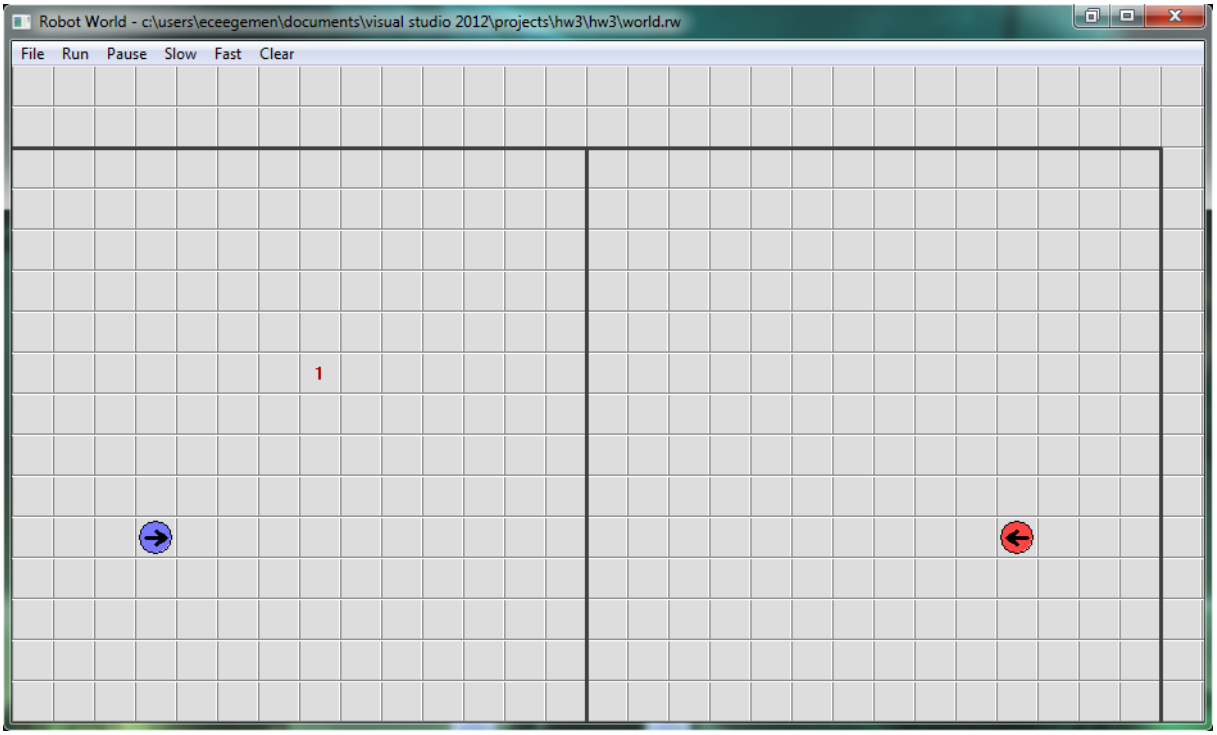
If your code does not compile, you will get **zero**. Please be careful about this and double check your code before submission.

Sample Runs

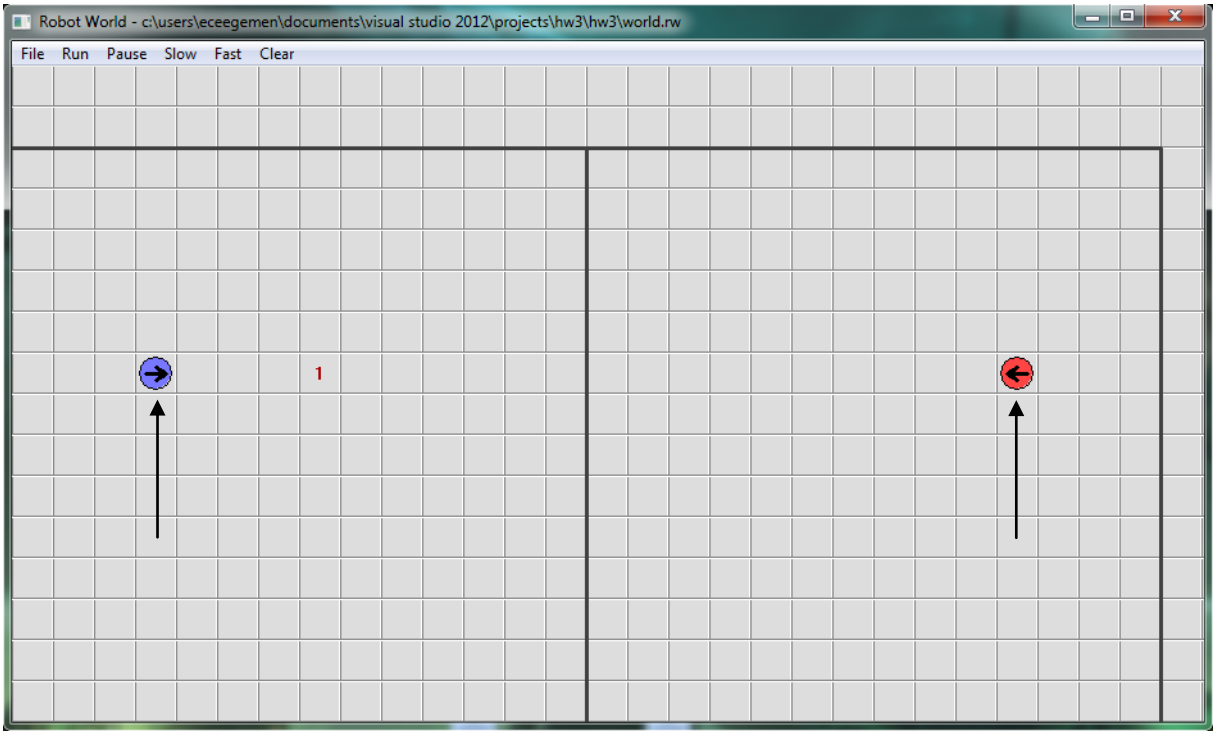
For your convenience, we provide a robot world file called "world.rw" in the same zip package as this document. In this rw file, the target cell is marked as 1. You may run your program on this robot world. Although the program should also work on an empty robot world, running it on world.rw would help you to check the correctness of your program since you visualize the targets on the robot world. To open world.rw file, first copy it in your local folder. After that execute your program and use File → Open menu item of the robot world.

Please examine the following screenshots for some examples. Here please remark that following screenshots show only some example cases; not all possible cases for relative positions of the player robot with respect to the target cell. Your program should work properly in all possible cases.

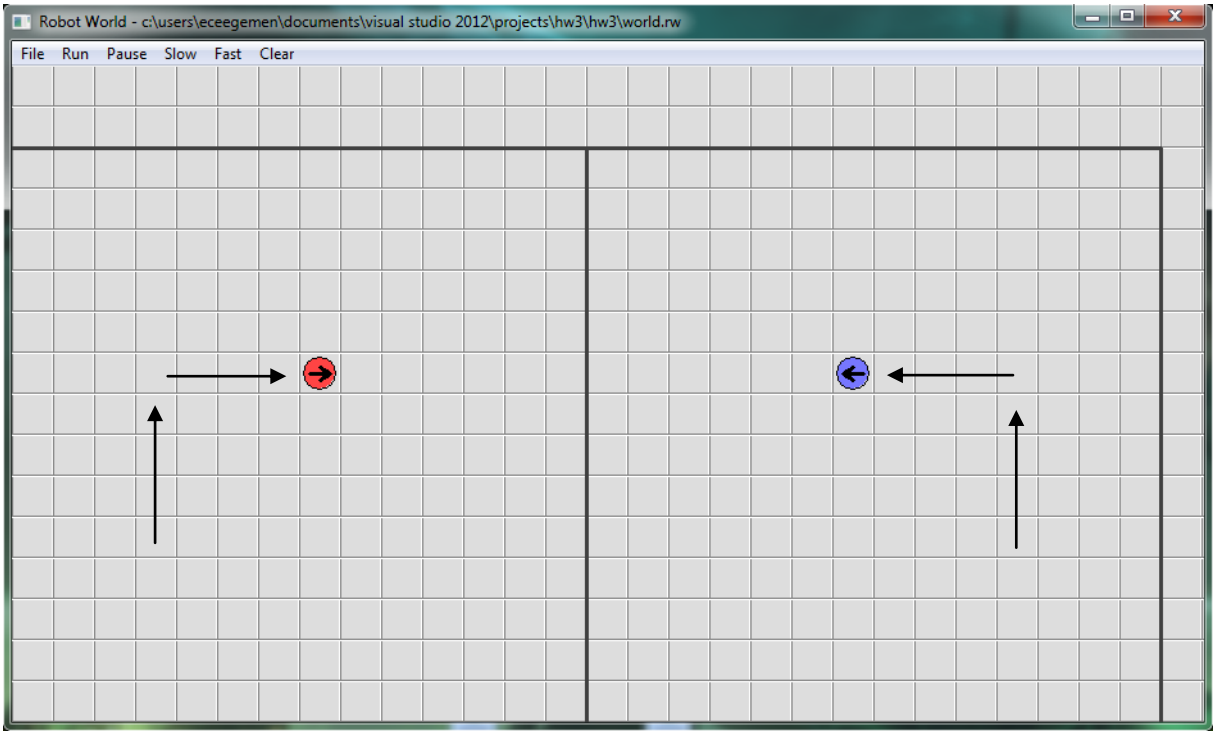
For example, if the input coordinates for the player robot are x=3 and y=4, your world may look something like the world given in the following figure when the robots are first created:



Following figure shows both robots in the middle of their travel:



Finally, you may see in the following figure how both robots will look after they finish their travel:



Please refer to the provided demo application to understand movements of the robots and try different cases.

Input and Output in Robot Environment

You cannot use `cin` and `cout` for input and output in robot environment. The reasons for this fact and remedies are explained in the RobotWorld.pdf file. Please read this document carefully and examine the example program given in recitation materials this week in order to understand the I/O functionality that you are going to use in this homework.

Use of Functions and Other Rules

Unlike the second homework, we will not specify any functions here. But you are expected to use functions to avoid code duplication and to improve the modularity of your program. **If your main function or any user-defined function is too long and if you do everything in main or in another user-defined function, your grade may be lowered.**

AND PLEASE DO NOT WRITE EVERYTHING IN MAIN AND THEN TRY TO SPLIT THE TASK INTO SOME FUNCTIONS JUST TO HAVE SOME FUNCTIONS OTHER THAN MAIN. THIS IS TOTALLY AGAINST THE IDEA OF FUNCTIONAL DESIGN AND NOTHING BUT A TRICK TO GET SOME POINTS. INSTEAD PLEASE DESIGN YOUR PROGRAM BY CONSIDERING NECESSARY FUNCTIONS AT THE BEGINNING.

Try to use parametric and non-void functions wherever appropriate. Do NOT use any global variables (variables defined outside the functions) to avoid parameter use.

Demo Application

Due to interactive feature of this homework, we are not able to provide sample outputs, but we provide an executable file that can be run to understand how your program should work.

We have already written the program for this homework and the corresponding executable file (demohw3.exe) is given to you within the same zip package as this homework document. Before pressing "run" button on the menu, either you have to open a sample robot world (files with .rw extension), or you have to create an environment specified in this homework.

No abrupt program termination please!

You may want to stop the execution of the program at a specific place in the program. Although there are ways of doing this in C++, it is not a good programming practice to abruptly stop the execution in the middle of the program. Therefore, your program flow should continue until the end of the main function and finish there.

General Rules and Guidelines about Homeworks

The following rules and guidelines will be applicable to all homeworks, unless otherwise noted.

How to get help?

You may ask questions to TAs (Teaching Assistants) of CS201. Office hours of TAs are at the class website. Recitations will partially be dedicated to clarify the issues related to homework, so it is to your benefit to attend recitations.

What and Where to Submit

Please see the detailed instructions below/in the next page. The submission steps will get natural/easy for later homeworks.

Grading and Objections

Careful about the semi-automatic grading: Your programs will be graded using a semi-automated system. Therefore you should follow the guidelines about input and output order; moreover you should also use same prompts as given in the Sample Runs. Otherwise semi-automated grading process will fail for your homework, and you may get a zero, or in the best scenario you will lose points.

Grading:

- Late penalty is 10% off of the full grade and only one late day is allowed.
- **Having a correct program is necessary, but not sufficient to get the full grade. Comments, indentation, meaningful and understandable identifier names, informative introduction and prompts, and especially proper use of required functions, unnecessarily long program (which is bad) and unnecessary code duplications (which is also bad) will also affect your grade.**
- Please submit your own work only (even if it is not working). It is really easy to find out "similar" programs!
- For detailed rules and course policy on plagiarism, please check out http://myweb.sabanciuniv.edu/gulsend/su_current_courses/cs-201-spring-2008/plagiarism/

Plagiarism will not be tolerated!

Grade announcements: Grades will be posted in SUCourse, and you will get an Announcement at the same time. You will find the grading policy and test cases in that announcement.

Grade objections: It is your right to object to your grade if you think there is a problem, but before making an objection please try the steps below and if you still think there is a problem, contact the TA that graded your homework from the email address provided in the comment section of your announced homework grade or attend the specified objection hour in your grade announcement.

- Check the comment section in the homework tab to see the problem with your homework.
- Download the .zip file you submitted to SUCourse and try to compile it.
- Check the test cases in the announcement and try them with your code.
- Compare your results with the given results in the announcement.

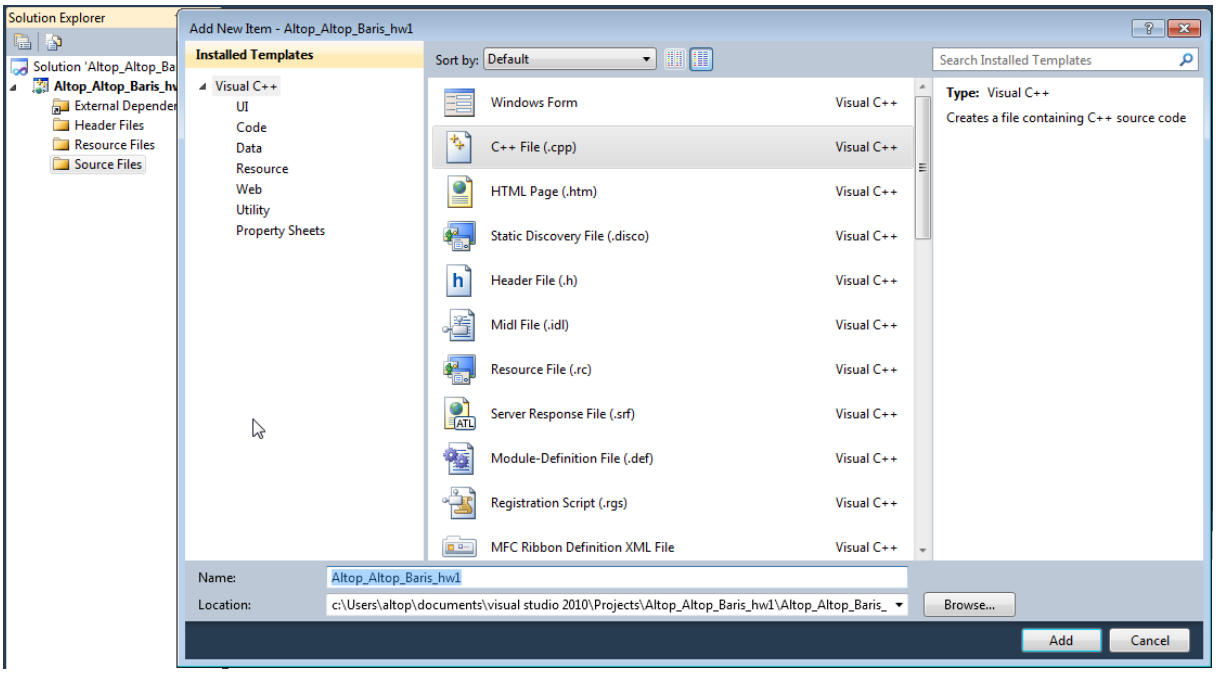
What and where to submit (IMPORTANT)

Submissions guidelines are below. Most parts of the grading process are automatic. Students are expected to strictly follow these guidelines in order to have a smooth grading process. If you do not follow these guidelines, depending on the severity of the problem created during the grading process, 5 or more penalty points are to be deducted from the grade.

Add your name to the program: It is a good practice to write your name and last name somewhere in the beginning program (as a comment line of course).

Name your submission file:

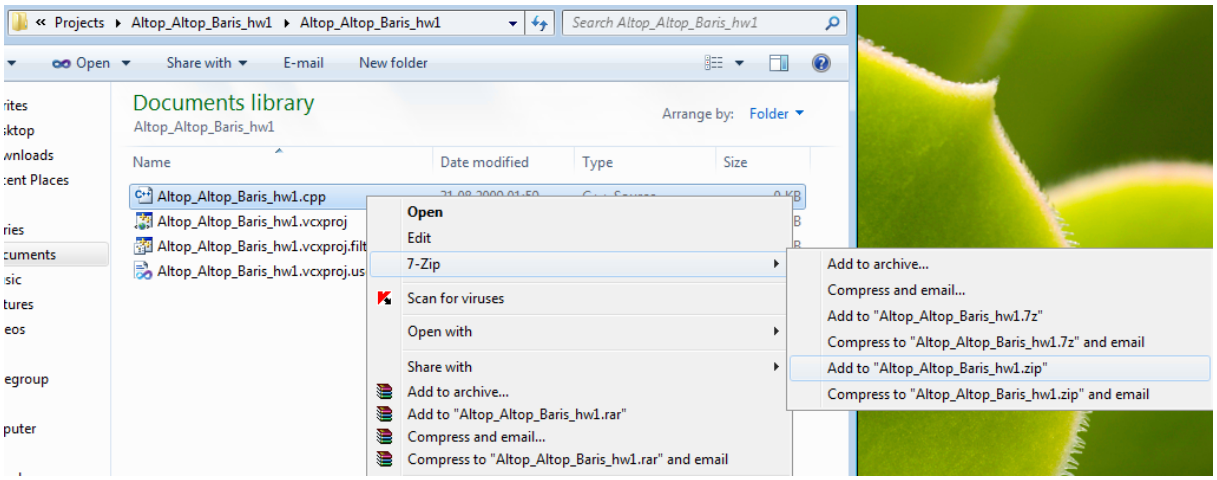
- ❑ Use only English alphabet letters, digits or underscore in the file names. Do not use blank, Turkish characters or any other special symbols or characters.
- ❑ Name your cpp file that contains your program as follows.
  "**SUCourseUserName_YourLastname_YourName_HWnumber.cpp**"



- Your SUCourse user name is actually your SUNet user name which is used for checking sabanciuniv e-mails. Do NOT use any spaces, non-ASCII and Turkish characters in the file name. For example, if your SUCourse user name is cago, name is Çağlayan, and last name is Özbugsızkodyazaroğlu, then the file name must be:

Cago_Ozbugsizkodyazaroglu_Caglayan_hw3.cpp

- Do not add any other character or phrase to the file name.
- Make sure that this file is the latest version of your homework program.
- Compress this cpp file using WINZIP or WINRAR programs. Please use "**zip**" compression. "rar" or another compression mechanism is NOT allowed. Our homework processing system works only with zip files. Therefore, make sure that the resulting compressed file has a zip extension.



- Check that your compressed file opens up correctly and it contains your **cpp** file. You will receive no credits if your compressed zip file does not expand or it does not contain the correct file.
- The naming convention of the zip file is the same as the cpp file (except the extension of the file of course). The name of the zip file should be as follows.

"**SUCourseUserName_YourLastname_YourName_HWnumber.zip**"

For example zubzipler_Zipleroglu_Zubeyir_hw3.zip is a valid name, but hw3_hoz_HasanOz.zip, HasanOzHoz.zip are NOT valid names.

Submission:

- Submit via SUCourse ONLY! You will receive no credits if you submit by other means (e-mail, paper, etc.).
  1) Click on "Assignments" at CS201 SUCourse (not the CS201 web site).
  2) Click Homework 3 in the assignments list.
  3) Click on "Add Attachments" button.
  4) Click on "Browse" button and select the zip file that you generated.
  5) Now, you have to see your zip file in the "Items to attach" list.
  6) Click on "Continue" button.
  7) Click on "Submit" button. We cannot see your homework if you do not perform this step even if you upload your file.

Resubmission:

- After submission, you will be able to take your homework back and resubmit. In order to resubmit, follow the following steps.
    1) Click on "Assignments" at CS201 SUCourse.
    2) Click Homework 3 in the assignments list.
    3) Click on "Re-submit" button.
    4) Click on "Add/remove Attachments" button
    5) Remove the existing zip file by clicking on "remove" link. This step is very important. If you do not delete the old zip file, we receive both files and the old one may be graded.
    6) Click on "Browse" button and select the new zip file that you want to resubmit.
    7) Now, you have to see your new zip file in the "Items to attach" list.
    8) Click on "Continue" button.
    9) Click on "Submit" button. We cannot see your homework if you do not perform this step even if you upload your file.

Successful submission is one of the requirements of the homework. If, for some reason, you cannot successfully submit your homework and we cannot grade it, your grade will be 0.

Good Luck!
Ece Egemen and Gülşen Demiröz