CS201 – Spring 2017-2018 - Sabancı University

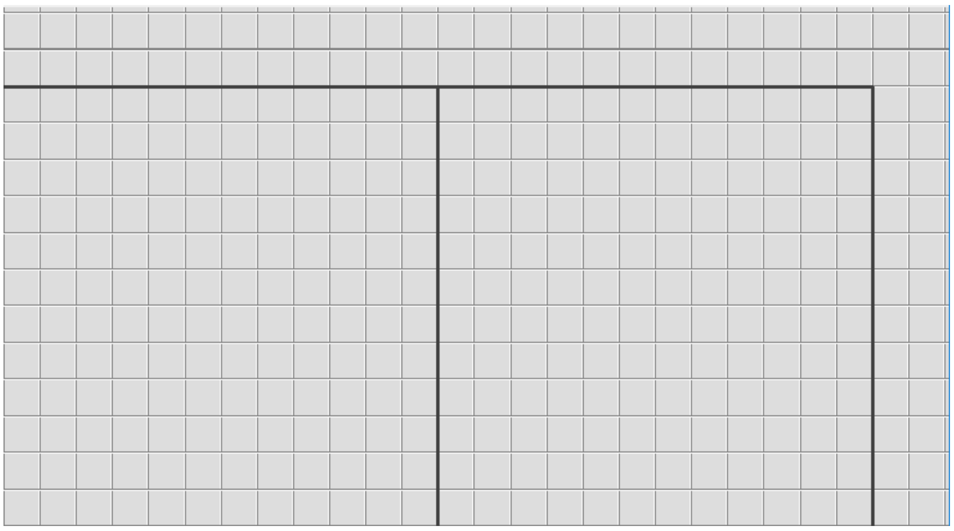
Homework #3 – The Spiral

Due March 14, 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 lecture materials. All the material needed for the robot class is also in the robot.zip (at the google drive of CS201). Please examine the RobotWorld.pdf file and the example programs. You will practice on the robot class during recitations this week. 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 in a location of which the x and y coordinates are inputs from the user. There are some limitations for these input x and y coordinates, which will be explained in a later section. Then, your program will create a mirror robot in the symmetrical position with respect to the middle line (mirror) in the robot world. Then, the original robot will begin a clockwise spiral movement starting from going to south by 1 cell. After completing 2 cycles (making a total of 8 moves) it will stop. Mirror robot will follow your original robot's movements symmetrically in a counter-clockwise spiral movement starting from going to south by 1 cell. Just like the original robot, the mirror robot will also stop after completing 2 cycles (making a total of 8 moves). A screenshot of the empty world is given below:



Details and Rules

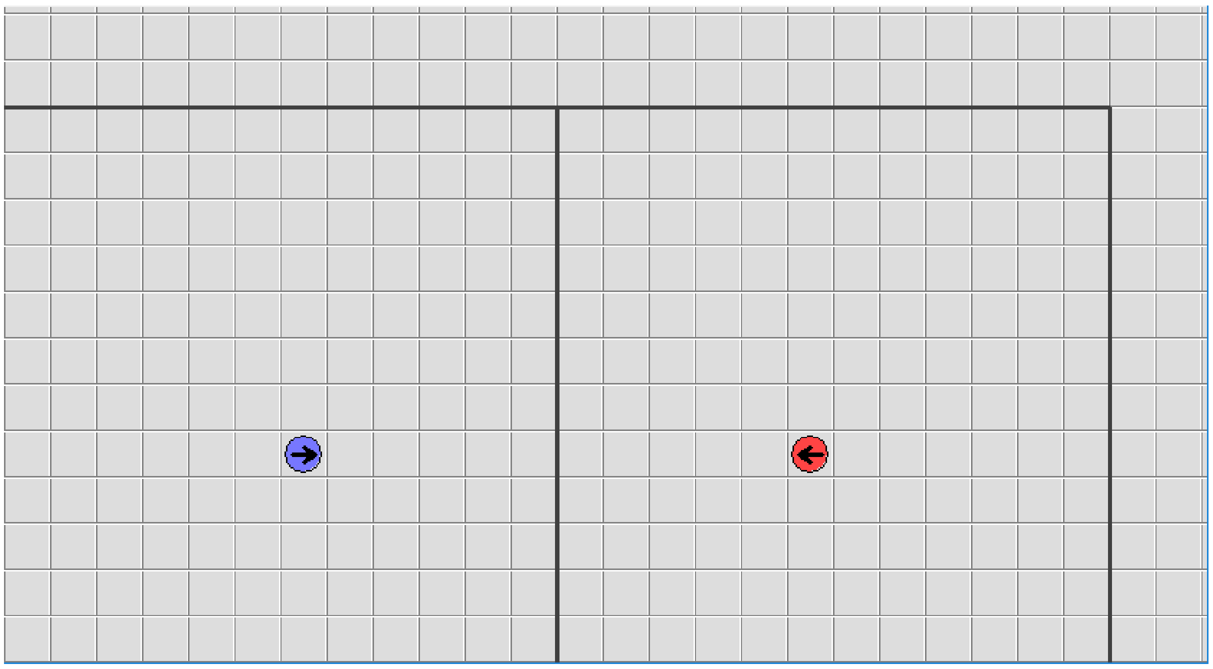
First, the position of the original robot will be entered by the user. The coordinates of this robot must be non-negative integers between 0 and 11 inclusively (may be equal to 0 or 11). Otherwise, your program should display an appropriate error message immediately and should not even prompt for the other inputs and of course **no** robot should be created.

The world is a rectangle (24x12) and it is divided into two squares (12x12) with a line in the middle shown in the figure below. The mirror robot should be created in the right square and it should be symmetric of the original 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 original robot and **west** for the mirror robot.

Then your program should set the colors of both robots; the color of the original robot must be set to **blue**, and the color of the mirror robot must be set to **red**.

Your world may look something like this when the robots are first created for input coordinates (6,4) entered for the original robot:



At this point, the original robot will begin its spiral movement starting from going to south by 1 cell. Spiral movement of the original robot should be in a clockwise fashion whereas the mirror robot in a counter-clockwise fashion. Which means after going to south, the original robot should turn to west whereas the mirror robot should turn to east before moving again. After moving to west or east by 1 cell, the robots then should turn to north. In order to make a proper spiral movement, the distance the original robot will travel should be increased to 2 at this point. After completing the first cycle (4 moves) we expect robots to change colors; the original robot should be **green** and the mirror robot should be **purple** from this point on. We expect the both robots to move a total of 8 times, you should determine the necessary distances to achieve spiral movement.

If robots' movement is blocked by a wall during the spiral movement, they must not move into the wall, instead you should display an appropriate message and end the program. But keep in mind that, even though the robots must not move into the wall, they should move until they are next to the wall.

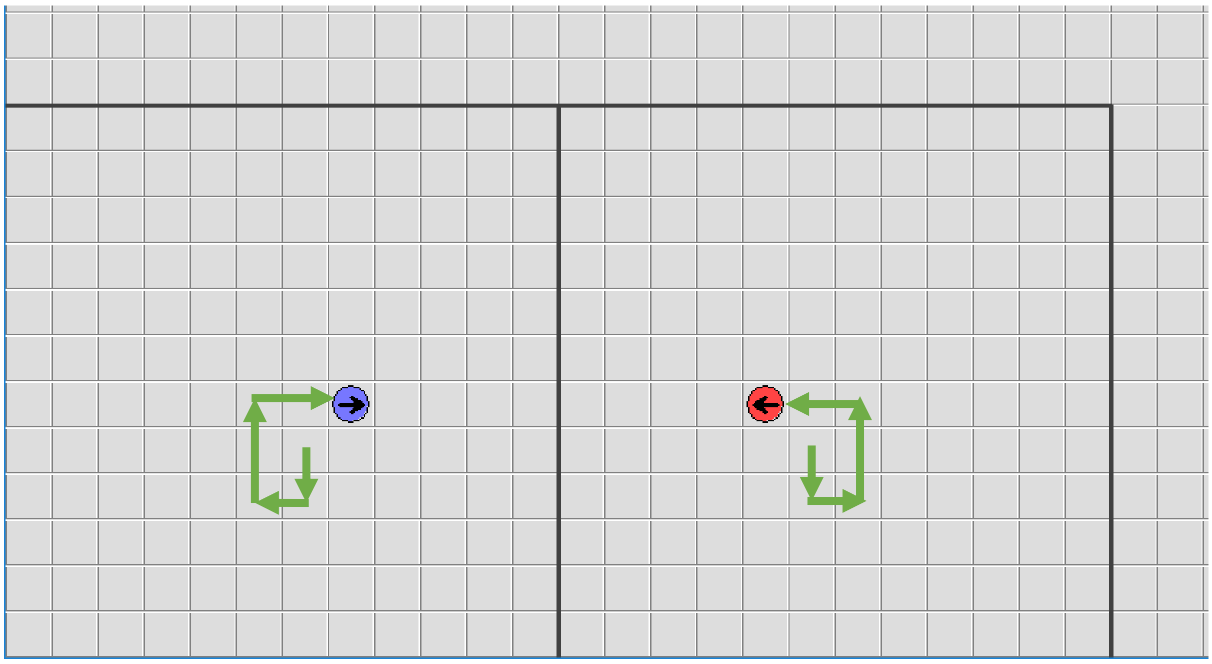
While the original robot makes its moves, the mirror robot will mimic its movements symmetrically. Which means; the mirror robot moves when the original robot moves, such that its movement is symmetric with respect to the mirror wall. You may refer to the following table for movements:

Original Robot	**Mirror Robot**
East	West
West	East
North	North
South	South

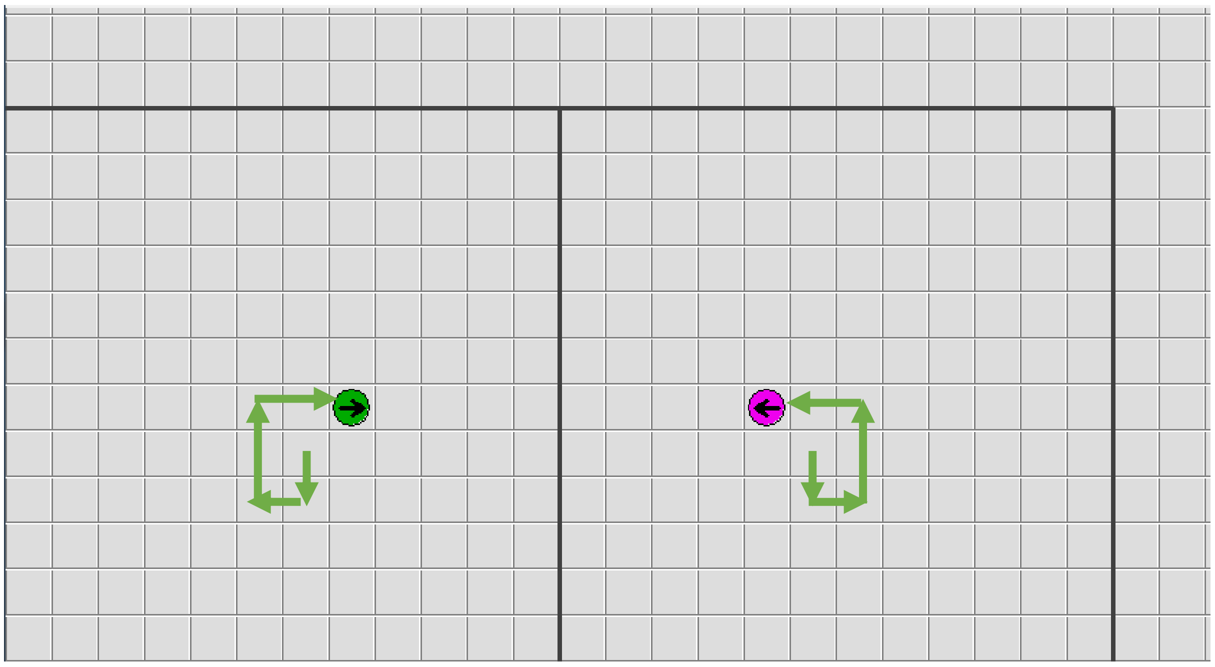
Keep in the mind that the robots cannot move at the same time. The original robot should move towards a direction first, and then the mirror robot should make its mimic movement before the original robot makes its next movement.

In the screenshot below, you may see an example movement flow for the robot created at (6,4).

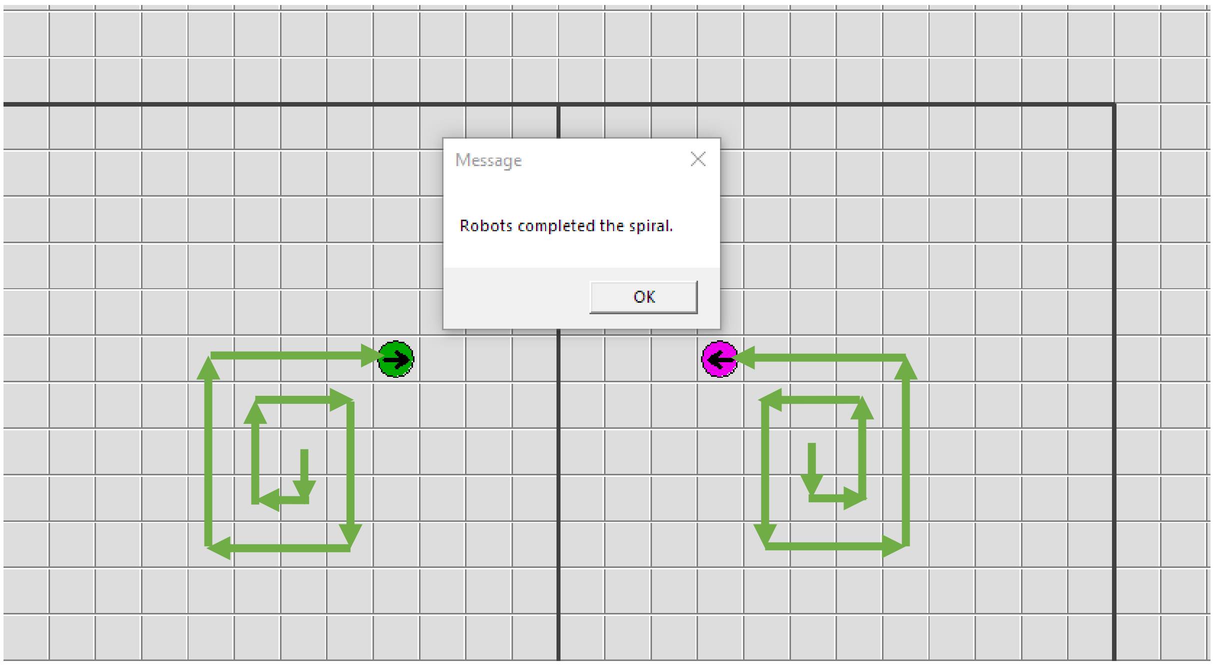
After completing the first cycle, but before changing the colors:



After changing the colors:



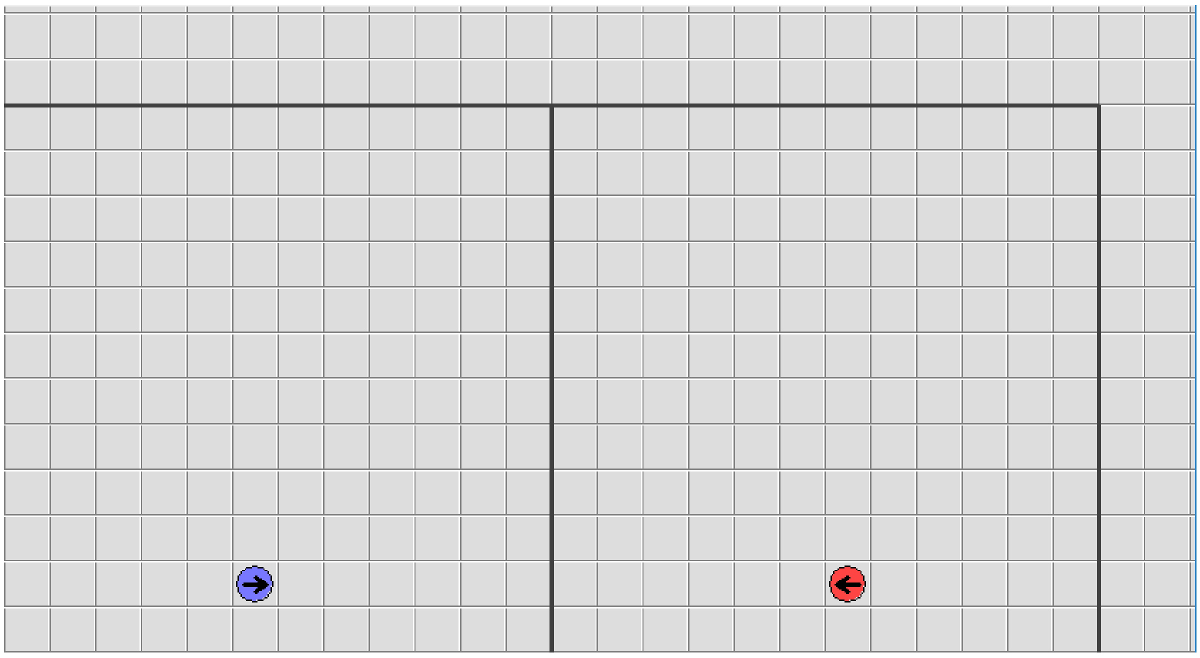
End of the spiral movement:


Message

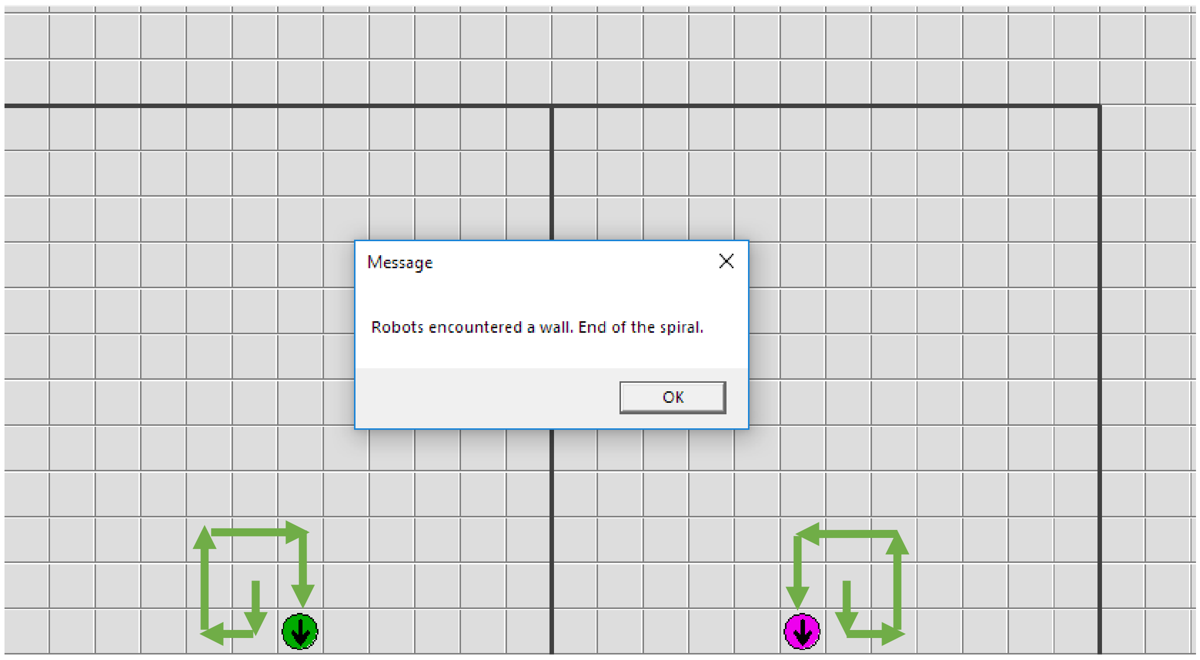
Robots completed the spiral.

Below you will find another example with the original robot created at (5,1), please notice that the spiral movement cannot be completed because of the wall but the robots do **not** die, instead an appropriate message is shown.

When robots are created:



The end result:


Message

Robots encountered a wall. End of the spiral.

The Program Flow

- Ask the user to input x coordinate of the original robot. Check the input value according to the following rule:
    - x-coordinate of the robot: $0 \leq x < 12$

  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 original robot. Check the input value according to the following rule:
    - y-coordinate of the robot: $0 \leq y < 12$

  If the value is correct, proceed, otherwise terminate the program. Inputs are assumed to be integers.
- Create the original robot at the given coordinates, facing east.
- Create the mirror robot at the symmetrical position according to the mirror, facing west.
- Set their colors; the original robot should be **blue**, the mirror robot should be **red**.
- The original robot will begin its spiral movement by going 1 cell to south. Remember that, the mirror robot should mimic its movements symmetrically with respect to the mirror line, after each move of the original robot.
- After completing a cycle (4 moves), robots should change colors; the original robot should be **green** and the mirror robot should be **purple**.
- After completing two cycles (8 moves), robots should stop their movements.
- If robots' movement is blocked by a wall during the spiral movement, they must not move into the wall, instead you should display an appropriate message saying "*Robots encountered a wall. End of spiral.*" and end the program. Even though the robots must not move into the wall, they should move until they are next to the wall.
- If robots complete the spiral movement, then your program should display an appropriate message saying "*Robots completed the spiral.*" before ending.

IMPORTANT!

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

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 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 (hw3_demo.exe) as well as the world file (mirror.rw) is given to you within the same zip package of this homework document.

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 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/courses/cs201/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)

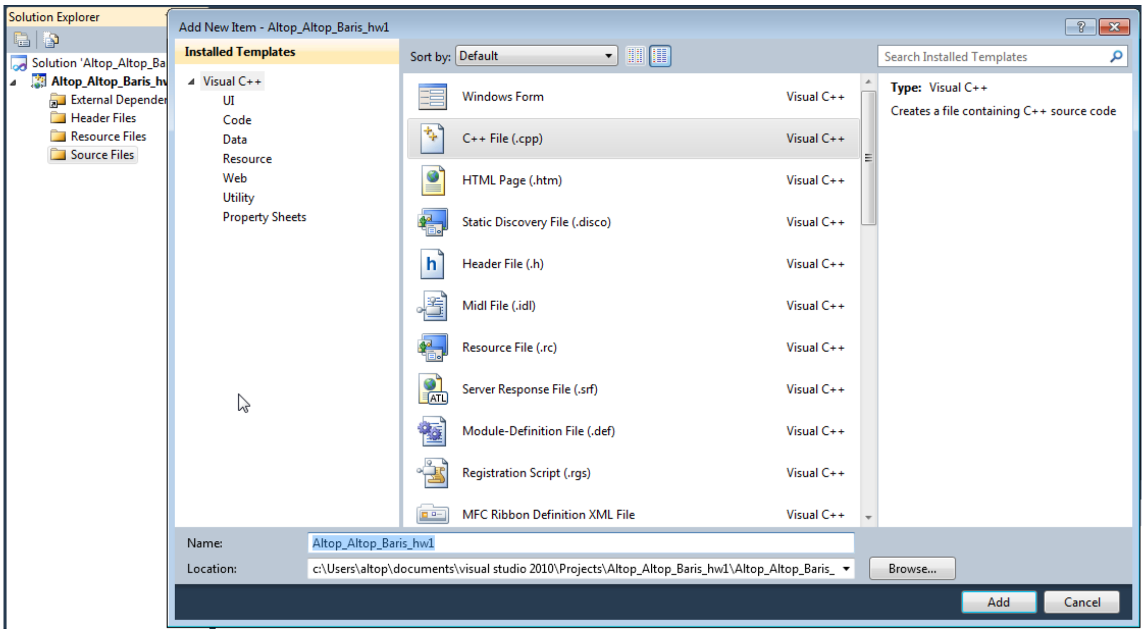
Submissions guidelines are below. Most parts of the grading process are automatic. Students are expected to strictly follow these guidelines in order to have a smooth grading process. If you do not follow these guidelines, depending on the severity of the problem created during the grading process, 5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

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

<u>Name your submission file:</u>

- <u>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.</u>
- Name your cpp file that contains your program as follows.

  “**SUCourseUserName_yourLastname_yourName_HWnumber.cpp**”


Solution Explorer
Solution 'Altop_Altop_Ba
Altop_Altop_Baris_hw
External Depende
Header Files
Resource Files
Source Files
Add New Item - Altop_Altop_Baris_hw1
Installed Templates
Visual C++
UI
Code
Data
Resource
Web
Utility
Property Sheets
Sort by: Default
Search Installed Templates
Windows Form Visual C++
C++ File (.cpp) Visual C++
HTML Page (.htm) Visual C++
Static Discovery File (.disco) Visual C++
Header File (.h) Visual C++
Midl File (.idl) Visual C++
Resource File (.rc) Visual C++
Server Response File (.srf) Visual C++
Module-Definition File (.def) Visual C++
Registration Script (.rgs) Visual C++
MFC Ribbon Definition XML File Visual C++
Browse...
Add
Cancel

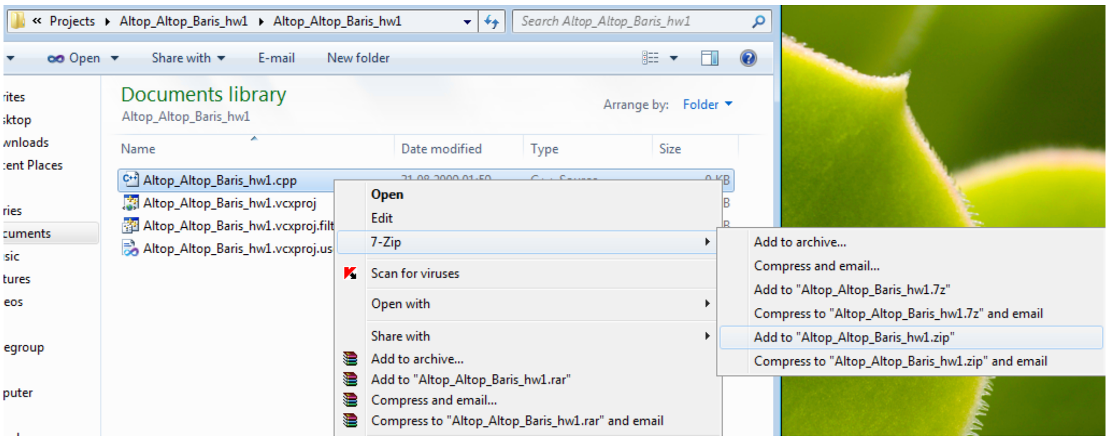
Type: Visual C++
Creates a file containing C++ source code

Name: Altop_Altop_Baris_hw1
Location: c:\Users\altop\documents\visual studio 2010\Projects\Altop_Altop_Baris_hw1\Altop_Altop_Baris_

- 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.
- You need to submit ALL .cpp and .h files including the robot and minifw files in addition to your main.cpp in your VS solution. Compress all your necessary files 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.


« Projects ▸ Altop_Altop_Baris_hw1 ▸ Altop_Altop_Baris_hw1
Search Altop_Altop_Baris_hw1
Open
Share with
E-mail
New folder
Documents library
Altop_Altop_Baris_hw1
Arrange by: Folder
Name
Date modified
Type
Size
Altop_Altop_Baris_hw1.cpp
Altop_Altop_Baris_hw1.vcxproj
Altop_Altop_Baris_hw1.vcxproj.filt
Altop_Altop_Baris_hw1.vcxproj.us
Open
Edit
7-Zip
Scan for viruses
Open with
Share with
Add to archive...
Add to "Altop_Altop_Baris_hw1.rar"
Compress and email...
Compress to "Altop_Altop_Baris_hw1.rar" and email
Add to archive...
Compress and email...
Add to "Altop_Altop_Baris_hw1.7z"
Compress to "Altop_Altop_Baris_hw1.7z" and email
Add to "Altop_Altop_Baris_hw1.zip"
Compress to "Altop_Altop_Baris_hw1.zip" and email

- ❑ Check that your compressed file opens up correctly and it contains your **cpp** file. You will receive no credits if your 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 don’t delete the old zip file, we get 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, İnanç Arın and Gülşen Demiröz