CS201 – Fall 2023-2024
Homework 3

– MasterMind Game –

Due November 24, Friday, 22:00 (Sharp Deadline)

Introduction

The aim of this homework is to practice on strings and loops. Of course, you will need to use the topics that we have seen before such as if-else statements and functions, as needed.

Description

In this homework, you will write a C++ program that simulates the MasterMind board game, which is a code-breaking game played by two players. Normally, the game is played using a *decoding board*, *code pegs*, and *key pegs*. Decoding board, which is shown below, is the board of the game that has a shield at one end covering a row of four large holes, and there are twelve additional holes next to a set of four small holes. The larger holes are to be filled by the code pegs, which are used to construct the secret and the guess codes, while the smaller holes are to be filled by the key pegs, which are used to provide feedback to the player who tries to guess the secret code. Code pegs have six different colors and key pegs have two different colors.

The two players decide in advance how many games they will play, which must be an even number. One player becomes the *code-maker* and the other becomes the *code-breaker*. The code-maker chooses a pattern of four code pegs, where the players decide whether to allow duplicates and blanks or not at the beginning of the game. The chosen pattern is placed in the four holes covered by the shield, so it is only visible to the code-maker but not to the code-breaker.

The code-breaker tries to guess the code-maker's pattern, in both order and color, within at most twelve turns. Each guess is made by placing a row of code pegs on the decoding board. Once placed, the code-maker provides feedback by placing the matching key pegs in the small holes of the row with the guess. A **red** key peg is placed for each code peg from the guess which is correct in both color and position, while a **white** key peg is placed for each code peg from the guess which is correct in color but wrong in position.

Once feedback is provided, another guess can be made by the code-breaker. This guessing and providing feedback process continue to alternate until either the code-breaker guesses the secret code correctly, or twelve incorrect guesses are made.

The code-maker gets one point for each incorrect guess a code-breaker makes, and also an extra point if the code-breaker cannot guess the pattern exactly in the last guess. On the other hand, if the code-breaker can guess the pattern exactly in some round of the game, (s)he earns one point. The winner is the one who has the most points after the agreed number of runs of the game.

Please see below for the modifications we made to this described game.

Your homework will be automatically graded using SUCourse, so it is very important to satisfy the exact same outputs given in the example test cases of SUCourse. Please submit your assignment by writing your main source (cpp) file content into the Answer field. You can utilize the **Check** button under the code editor at SUCourse to check whether your implementation is working in the expected way. After you check your solution code, you will see your grade with the example test cases used; however your homework will then be graded with **different** test cases.

To submit your homework, you must hit the **"Finish attempt..."** and **"Submit all and finish"** buttons. **Just a reminder of a character ↩ which refers to a newline in your expected output.**

Inputs, Flow of the Program and Outputs

Inputs

There are four different types of inputs for this program:

1. Names of the players (of type string),

2. Total number of runs for the game (of type integer),
3. Secret code (of type string), and
4. Guess code (of type string).

Input Check

Following input checks must be performed by your program and in case that the user enters an invalid value, your program should continuously prompt for the respective input until the user enters a valid value.

1. Total number of runs for the game must be a positive even number,
2. Secret/guess code must consist exactly of four characters,
3. Each character in the secret/guess code must be one of the followings: "R" "G", "B", "P", "Y" or "M".
4. No duplicate letters are allowed in secret and guess codes.

Please see Sample Runs section to better understand the displayed prompts and messages, and input/outputs.

Flow of the Program

In the program that you will implement, there will be two players. The user will first be prompted to enter the names of the players and display a welcome message by referring to the players with their names. Then, your program will ask for the number of runs of the game. The value to be entered by the user for the number of runs must be a positive even number. If the user enters an incorrect value, then your program should display an error message and prompt for another number of runs until the user enters a positive even number.

Once the number of runs is specified, the simulation of the MasterMind game will start. In each run of the game, one of the players will be the code-maker and the other will be the code-breaker, in turns. In the first run, the code-maker will be the player whose name was given as the first input and the code-breaker will be the player whose name was given as the second input. Then the players will change their roles in the second run of the game, i.e. the player whose name was given as the second input will be the code-maker, while the other player will be the code-breaker. This will continue until the game is played specified number of runs.

Initially, both of the players will have zero points. In each run of the game, the code-breaker will have **six chances** to guess the secret code correctly. For each incorrect guess made by the code-breaker, the code-maker will earn one point. If the code-breaker can guess the secret code in at most **six rounds**, then (s)he earns one point; otherwise, i.e. if the code-breaker cannot guess the secret code correctly in the last round, then the code-maker will earn one point.

In a particular run of the game, your program should first display who is the code-maker and prompt for a secret code. The code must consist of four **different** letters and those letters can only be "R" "G", "B", "P", "Y" and "M", which stand for "Red", "Green", "Purple", "Yellow" and "Magenta", respectively. Duplicate letters in a code are **NOT** allowed. Notice that these letters correspond to the code pegs in the original game described above. If the user enters an invalid value, then your program should display an error message and prompt for another secret code (by referring to the code-maker with his/her name) until the user enters a valid one.

Once the secret code is entered correctly by the code-maker, the code-breaker can start with his/her guesses. The code-breaker will have **at most six** chances to make a correct guess. In each round of this guessing process, your program should prompt for a guess by referring to the code-breaker with his/her name. The guess code entered by the code-breaker must consist of four different letters and those letters can only be "R" "G", "B", "P", "Y" and "M", as in the case of the secret code. If the user enters an invalid value, then your program should display an error message and prompt for another guess code (by referring to the code-breaker with his/her name) until the user enters a valid one.

When a valid guess is made by the code-breaker, your program should check this guess against the secret code. If the guess code is exactly the same as the secret code, then the code-breaker wins this run of the game, an informative message is displayed and the code-breaker gets one point. Otherwise, the code-maker will get one point and your program will provide some feedback to the code-breaker. This feedback can be of length from two to four and it must consist only of the letters "W" and/or "R", which stand for "White" and "Red" key pegs in the original game, respectively. In the feedback to be provided to the code-breaker, "R"s must be displayed before "W"s, if there are any. Here, "W" means that one of the characters within the guess code also appears to be within the secret code but at a different location, while "R" means that one of the characters within the guess code is at the exact same location in the secret code. For each character of the guess code that also appears in the secret code, the feedback for that character should be either "W" or "R", depending on whether the location of this character within the secret code matches that of the guess code.

Once the feedback is provided, another guess can be made by the code-breaker. This guessing and feedbacking process alternates until either the code-breaker guesses the secret code correctly, or **six** incorrect guesses are made. Thereafter, the game should continue with the next run, if there are any left.

When all the runs are played, your program should first display the scores of the players and then display who the winner is, if there is any. If the scores of the players are equal to each other, then your program should display that there is a tie between the players. If the score of the first player is greater than that of the other player, then the first player wins the overall game; otherwise, the second player wins the overall game.

Please see Sample Runs section to better understand the flow of the program.

IMPORTANT!

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

Note: Please avoid using cin.ignore() and cin.get() in your code for your assignment. We are using CodeRunner, and these functions may cause unexpected behaviour.

VERY IMPORTANT!

Your programs will be compiled, executed and evaluated automatically; therefore you should definitely follow the rules for prompts, inputs and outputs. You can check the example test case outputs from SUCourse to get more information about the expected output.

Order of inputs and outputs must be in the mentioned format.

Following these rules is crucial for grading, otherwise, our software will not be able to process your outputs and you will lose some points in the best scenario.

Sample Runs

Below, we provide only one sample run of the program that you will develop, for more sample runs please check the SUCourse example test cases.
The *italic*, **bold** and yellow highlighted phrases are inputs taken from the user.
NOTE THAT these inputs and the newlines after the inputs are missing at SUCourse in the outputs expected from you, so please ignore this as you copy/paste your C++ code from VS/XCode to SUCourse, the same will happen

to your code too. You can see samples of SuCourse output in your assignment on SuCourse.

Visual Studio/XCode Outputs

Sample Run 1

```
Please enter the names of the players: Duygu Gulsen
Welcome Duygu and Gulsen

Please enter the number of runs: -2
Invalid number of runs!
Please enter the number of runs: 0
Invalid number of runs!
Please enter the number of runs: 1
Invalid number of runs!
Please enter the number of runs: 3
Invalid number of runs!
Please enter the number of runs: 11
Invalid number of runs!
Please enter the number of runs: 2

Duygu, it is your turn to select the secret!
Valid colors: R - Red, G - Green, B - Blue, P - Purple, Y - Yellow,
M - Magenta
Duygu, please enter your secret: QWET
Invalid secret!
Duygu, please enter your secret: UIOK
Invalid secret!
Duygu, please enter your secret: RGBT
Invalid secret!
Duygu, please enter your secret: rgBY
Invalid secret!
Duygu, please enter your secret: RGBR
Invalid secret!
Duygu, please enter your secret: RGBP

Gulsen, please enter your guess: rgbm
Invalid guess!
Gulsen, please enter your guess: MYPB
WW
Gulsen, please enter your guess: MYP
Invalid guess!
Gulsen, please enter your guess: YMPGR
Invalid guess!
```

```
Gulsen, please enter your guess: GMYP
RW
Gulsen, please enter your guess: RMYP
RR
Gulsen, please enter your guess: RBYP
RRW
Gulsen, please enter your guess: RBGP
RRWW
Gulsen, please enter your guess: RBBY
Invalid guess!
Gulsen, please enter your guess: RPGB
RWWW
Gulsen, you're out of chances, you lost this run!

Gulsen, it is your turn to select the secret!
Valid colors: R - Red, G - Green, B - Blue, P - Purple, Y - Yellow,
M - Magenta
Gulsen, please enter your secret: YMPG

Duygu, please enter your guess: MPGR
WWW
Duygu, please enter your guess: BPMY
WWW
Duygu, please enter your guess: PMYB
RWW
Duygu, please enter your guess: PM
Invalid guess!
Duygu, please enter your guess: YMBP
RRW
Duygu, please enter your guess: YMPB
RRR
Duygu, please enter your guess: MYGP
WWWW
Duygu, you're out of chances, you lost this run!

The game has ended!
Duygu: 7 - Gulsen: 7

Duygu and Gulsen, there is a tie!
```

Sample Run 2

```
Please enter the names of the players: Gulsen Duygu
Welcome Gulsen and Duygu

Please enter the number of runs: 4

Gulsen, it is your turn to select the secret!
Valid colors: R - Red, G - Green, B - Blue, P - Purple, Y - Yellow,
M - Magenta
Gulsen, please enter your secret: GPYB

Duygu, please enter your guess: GPYB
Duygu, you won this run!

Duygu, it is your turn to select the secret!
Valid colors: R - Red, G - Green, B - Blue, P - Purple, Y - Yellow,
M - Magenta
Duygu, please enter your secret: BPYM

Gulsen, please enter your guess: GPMR
RW
Gulsen, please enter your guess: GMPY
WWW
Gulsen, please enter your guess: MPRB
RWW
Gulsen, please enter your guess: BPMR
RRW
Gulsen, please enter your guess: BPMY
RRWW
Gulsen, please enter your guess: BPYM
Gulsen, you won this run!

Gulsen, it is your turn to select the secret!
Valid colors: R - Red, G - Green, B - Blue, P - Purple, Y - Yellow,
M - Magenta
Gulsen, please enter your secret: YMGB

Duygu, please enter your guess: YGMP
RWW
Duygu, please enter your guess: YMPG
RRW
Duygu, please enter your guess: YMGP
```

```
RRR
Duygu, please enter your guess: YMGB
Duygu, you won this run!

Duygu, it is your turn to select the secret!
Valid colors: R - Red, G - Green, B - Blue, P - Purple, Y - Yellow,
M - Magenta
Duygu, please enter your secret: PBYG

Gulsen, please enter your guess: GBYP
RRWW
Gulsen, please enter your guess: PBYG
Gulsen, you won this run!

The game has ended!
Gulsen: 5 - Duygu: 8

Congrats Duygu, you have won the game!
```

General Rules and Guidelines about Homework

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

How to get help?

You may ask questions to TAs (Teaching Assistants) or LAs (Learning Assistants) of CS201. Office hours of TAs/LAs are at the SUCourse.

What and Where to Submit

You can prepare (or at least test) your program using MS Visual Studio 2019 C++ (Windows users) or using XCode (macOS users).

- Your code will be automatically graded using SUCourse. Therefore, it is essential that you ensure your output matches the exact same outputs given in the example test cases provided by SUCourse.
- After writing your code, use the "Check" button located under the code editor in SUCourse to see your grade based on the example test cases used. This grade will give you an idea of how well your code is performing.

- Note that the example test cases used for checking your code are not the same as the ones used for grading your homework. Your final grade will be based on different test cases. Therefore, it is important that you carefully follow the instructions and ensure that your code is working correctly to achieve the best possible grade on your homework assignment.
- To submit your homework, click on the "Finish attempt..." button and then the "Submit all and finish" button. If you wish to submit again before the due date, you can press the "Re-attempt quiz" button.
- Submit your work **through SUCourse only**! You will receive no credits if you submit by any other means (email, paper, etc.).

Grading, Review and Objections

Be careful about the automatic grading: Your programs will be graded using an automated system. Therefore, you should follow the guidelines on the input and output order. Moreover, It is important to use the exact same text as provided in the example test case outputs from SUCourse. Otherwise, the automated grading process will fail for your homework, and you may get a zero, or in the best scenario, you will lose points.

Grading:

- There is NO late submission. You need to submit your homework before the deadline. Please be careful that SUCourse time and your computer time may have 1-2 minute differences. You need to take this time difference into consideration.
- 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.
- If your code does not work because of a syntax error, then we cannot grade it; and thus, your grade will be 0.
- Please submit your **own** work only. It is really easy to find "similar" programs!
- Plagiarism will not be tolerated. Please check our plagiarism policy given in the Syllabus.

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.
- Check the test cases in the announcement and try them with your code.
- Compare your results with the given results in the announcement.

Good Luck!
CS201 Instructors