CS201 – Spring 2021-2022

Take-Home Exam 2

Due April 17th, Sunday, 23:55 (Sharp Deadline)

Introduction

The aim of this homework is to practice on functions and if-else statements.

Description

In this homework, you will write a C++ program that calculates the calories the user has burned, compares it to their goal, and recommends exercises depending on how far they are from their goal.

The user will first input his/her name in the main function of your program. Other inputs will be asked in a different function, where a second prompt will be received that wants the user to enter his/her weight in terms of kilograms. Then, the user will be prompted to enter the average speed and how many minutes (s)he spent walking, running, and cycling in a week, respectively. The last input will be how many calories the user aims to burn in that week. All in these prompts, your program should refer to the user with his/her name.

After the inputs are completed, in another function, your program will check whether these inputs are valid or not. The weight, speed and minute inputs have some predefined ranges that they are supposed to be in. If they are out of range, your program should display certain messages. The ranges and messages are given below:

Range	Message
weight < 30	Weight out of range!
speed < 4 OR speed > 30	Speed out of range!
minute < 0 OR minute > 2000	Minute out of range!

Note that the order of the input checks is important. Firstly, weight should be checked. If it is out of range, you should not continue with the calculations. If weight is valid, then the speed should be checked. Again, if it is out of range, then your program should not continue with the calculations. If speed is valid, then the minute should be checked. If all of them are valid, only then your program should proceed to the calculations (HINT: Use if-else statements). Also, note that you should first check the inputs for walking, then running, then cycling.

After the input check is done, your program should continue with the calculations in a separate function. In that function, you should first calculate how many calories have been burned for each exercise. For that, first you should determine the MET values of each exercise, which are as indicated in the table below.

Exercise	**Speed**	**MET**
Walking	>= 4, < 6.5	4
Walking	>= 6.5	6.5
Running	>=4, < 11	9
Running	>= 11	12.5
Cycling	>= 4, < 20	6
Cycling	>= 20, < 25	8
Cycling	>= 25	10.5

You should also have a separate function to determine the MET values. This function should determine the MET values using if-else statements and at the end, it should return the corresponding MET value for a given exercise.

After all the MET values for all exercises are determined, in the function where your program is supposed to do all the calculations, you will calculate how many calories have been burned for each exercise.

calorie burn = weight x MET

The above formula gives the calorie burn for a specific exercise with a given MET value for one hour. The important point here is that your program will get the number of minutes the user spent on an exercise. So, after you have calculated the calorie burn for an hour, you need to convert it to one minute, and then multiply that with the number of minutes that was given as input. Then, your program should sum all the calorie burns to get the total and afterwards, it should get the difference between the goal calories and total calories.

After everything is calculated, your program will make use of another function. This function will be used to display the output to the user. Firstly, your program should display the calorie burn separately for each exercise, and then it should display the total calorie burn. Thereafter, what your program should display depends on the goal and the total calorie values.

If the goal is the same as the total, then your program should print the following:

"Congratulations! You have reached your goal!"

If the total is greater than the goal, then your program should print the following, where the bold and italicized "***difference***" will be calculated by your program:

"You have surpassed your goal! You can eat something worth *difference* calories :)"

If the goal is greater than the total, then your program should print the following, where the bold and italicized ***variables*** will be calculated by your program:

"You did not reach your goal by *difference* calories.
You need to walk *minuteCalculation* minutes more to reach your goal or,
You need to run *minuteCalculation* minutes more to reach your goal or,
You need to do cycling *minuteCalculation* minutes more to reach your goal."

All of steps described above should also be done again for a second user. The functions are explained more in detail below.

Your take-home exams will be automatically graded using GradeChecker, so it is very important to satisfy the exact same output given in the sample runs. You can utilize GradeChecker (http://learnt.sabanciuniv.edu/GradeChecker/) to check whether your implementation is working in the expected way. To be able to use GradeChecker, you should upload all of your files used in the take-home exam (**only** *your_main_cpp* file for this take-home exam). Additionally, you should submit all of your files to SUCourse (**only** *your_main_cpp* file for this take-home exam) **without zipping** them. **Just a reminder, you will see a character ¶ which refers to a newline in your expected output.**

The name of your main source (cpp) file should be in the expected format: "SUCourseUsername_THEnumber.cpp" (all lowercase letters). Please check the submission procedures of the take-home exam, which are listed at the end of this document.

Use of Functions (EXTREMELY IMPORTANT)

You have to follow the specifications below for function declaration and callings. The grading criteria will include proper use of these parametric functions. Do NOT use any global variables (variables defined outside the functions) to avoid parameter use. Unnecessary code duplication will cause grade reduction as well.

In the first homework, you were not supposed to implement any functions. However, in this homework you are expected to (actually you have to) use some function(s). The guidance about using functions in this homework is below.

A total of five user-defined functions (other than the main) must be implemented in this homework. On top of these functions, you may of course use other functions, if you want. In the main function, your program will ask the name of the user, and then call the ***getInput*** function. After that, your program will ask again for a second user's name, and call the ***getInput*** function again for that user. The function explanations are stated as below:

1. ***getInput*****:** A void function that takes *username* as its only parameter, and asks the user for their weight, speed and minutes spent on three exercises (walking, running, cycling), and weekly calorie burn goal. This function will call the ***inputCheck*** function in if-else statements, and then call the function ***computeResults*** if ***inputCheck*** returns true for all arguments.

2. ***inputCheck*****:** A non-void function (returns true/false result to the caller) that takes *speed*, *weight* and *minute* as its parameters, and checks whether they are in the range that they are supposed to be or not. If they're not in the range, the function will display an appropriate message and return false. Otherwise, it will return true.

3. ***calculateMET*****:** A non-void function (returns a result to the caller) that takes *speed* and *exercise* as its parameters, and determines what the MET value is depending on the speed. The parameter exercise is of type string and it is the name of the exercise. In if-else statements, you can first check the exercise, then speed, and then determine the MET. At the end of the function, it should return the calculated MET value.

4. ***computeResults*****:** A void function that takes *weight, goal, speedWalk, speedRun, speedCycling, minWalk, minRun* and *minCycling* as its parameters, and does all the necessary computations. First of all, your program will need the MET values for each exercise. So, you need to call the ***calculateMET*** function from this function, separately for each exercise. After

all three MET values are determined, your program will calculate the calories burned for each of them. Then, your program should compute the total calories burned, and also calculate the difference between the goal calories and total calories as it will be needed later. Then you will call ***`displayResults`*** function from this function.

5. ***<u>`displayResults`</u>*:** A void function that takes *`difference, total, weight, walkMET, runMET, cyclingMET, walkCalorie, runCalorie`* and *`cycleCalorie`* as its parameters and displays the required stuff. Your program will first print calories burned for each exercise and the total calories, and then according to the difference, your program will print appropriate messages as explained above.

No abrupt program termination please!

Especially during the input check, 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.

<u>VERY IMPORTANT!</u>

Your programs will be compiled, executed and evaluated automatically; therefore you should definitely follow the rules for prompts, inputs and outputs. See **<u>Sample Runs</u>** section for some examples.

<u>Order</u> of inputs and outputs must be in the abovementioned format.

<u>Prompts</u> for inputs and outputs must be **<u>exactly the same</u>** with examples.

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

<u>IMPORTANT!</u>

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

Sample Runs

Below, we provide some sample runs of the program that you will develop. The *italic* and **bold** phrases are inputs taken from the user. You should follow the input order in these examples and the prompts that your program will display **must** be **exactly the same** as given in the following examples.

Sample Run 1

```
Please enter your name: Naz
Welcome Naz, please enter your weight(kg): 55
Naz, please enter speed(km/h) and minutes spent in a week for the
activities below.
Walking: 6 200
Running: 12 160
Cycling: 22.5 190
Naz, please enter your weekly calorie burn goal: 3960
From walking, you burned 733.333 calories.
From running, you burned 1833.33 calories.
From cycling, you burned 1393.33 calories.
You burned 3960 calories.
Congratulations! You have reached your goal!

Please enter your name: Berk
Welcome Berk, please enter your weight(kg): 72.5
Berk, please enter speed(km/h) and minutes spent in a week for the
activities below.
Walking: 7.5 220
Running: 9 180
Cycling: 26 150
Berk, please enter your weekly calorie burn goal: 5000.5
From walking, you burned 1727.92 calories.
From running, you burned 1957.5 calories.
From cycling, you burned 1903.13 calories.
You burned 5588.54 calories.
You have surpassed your goal! You can eat something worth 588.042
calories :)
```

Sample Run 2

```
Please enter your name: Burak
Welcome Burak, please enter your weight(kg): 75
Burak, please enter speed(km/h) and minutes spent in a week for the
activities below.
Walking: 5 100
Running: 12 120
Cycling: 21 200
Burak, please enter your weekly calorie burn goal: 5500
From walking, you burned 500 calories.
From running, you burned 1875 calories.
From cycling, you burned 2000 calories.
You burned 4375 calories.
You did not reach your goal by 1125 calories.
You need to walk 225 minutes more to reach your goal or,
You need to run 72 minutes more to reach your goal or,
You need to do cycling 112.5 minutes more to reach your goal.

Please enter your name: Ece
Welcome Ece, please enter your weight(kg): 25
Ece, please enter speed(km/h) and minutes spent in a week for the
activities below.
Walking: 5 160
Running: 7.7 200
Cycling: 22 150
Ece, please enter your weekly calorie burn goal: 4000
Weight out of range!
```

Sample Run 3

```
Please enter your name: Berkay
Welcome Berkay, please enter your weight(kg): 70
Berkay, please enter speed(km/h) and minutes spent in a week for the
activities below.
Walking: 50 150
Running: 10 3000
Cycling: 25 150
```

```
Berkay, please enter your weekly calorie burn goal: 3000
Speed out of range!

Please enter your name: Defne
Welcome Defne, please enter your weight(kg): 60
Defne, please enter speed(km/h) and minutes spent in a week for the
activities below.
Walking: 6 90
Running: 8 100
Cycling: 25 2100
Defne, please enter your weekly calorie burn goal: 3500
Minute out of range!
```

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/LAs (Teaching/Learning Assistants) of CS201. Office hours schedule of all TAs/LAs can be found at the course's google site. 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

Be careful about the full-automatic grading: Your programs will be graded using an automated system. Therefore, you should follow the guidelines about input and output order. Moreover, you should also use the same prompts as given in the Sample Runs. 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.

To get help using GradeChecker you may ask questions to the list of your grader TAs: cs201gchelp@lists.sabanciuniv.edu.

Grading:

- ❏ **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 programs (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 "similar" programs!
- ❏ For detailed rules and course policy on plagiarism, please check out https://sites.google.com/sabanciuniv.edu/cs201 and keep that in mind.

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 announcement.

- Check the comment section in the homework tab to see the problem with your homework.
- Download the files 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)

Submission guidelines are given below. Students **MUST** strictly follow these guidelines to have a smooth grading process. If you don't follow these guidelines, depending on the severity of the problem created during the grading process, 5 or more penalty points will 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). Do not use Turkish characters in your program (not even in the comments).

Name your submission file:

❑ Use only English alphabet lowercase 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_lastname_name_hwnumber.cpp"

❑ Your SUCourse user name is actually your SUNet username, 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_hw2.cpp

❑ Do not add any other characters or phrases to the file name.

❑ Make sure that this file is the latest version of your homework program.

❑ You have to submit all files that are used in your homework without zipping them.

Submission:

➢ Submit via SUCourse ONLY! No credits if you submit by other means (e-mail, paper, etc.).
  - Click on "Assignments" at CS201 SUCourse (not the CS201 web site).
  - Click Homework 2 in the assignments list.
  - Click on the "Add Attachments" button.
  - Click on the "Browse" button and select all files used in homework.
  - Now, you have to see your file(s) in the "Items to attach" list.
  - Click on the "Continue" button.
  - Click on the "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.
  - Click on "Assignments" at CS201 SUCourse.
  - Click Homework 2 in the assignments list.
  - Click on the "Re-submit" button.
  - Click on the "Add/remove Attachments" button.
  - Remove the existing files by clicking on "remove" link. This step is very important. If you do not delete the old files, we receive both files and the old one may be graded.
  - Click on the "Browse" button and select the new files that you want to resubmit.
  - Now, you have to see your new files in the "Items to attach" list.
  - Click on the "Continue" button.
  - Click on the "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!
E. Beyza Çandır & CS201 Instructors