CS201 – Spring 2020-2021 - Sabancı University
Take Home Exam #1
-Exercise and Calorie Calculator-
Due March 24th, Wednesday, 23:50 (Sharp Deadline)

Motivation

The aim of this Take-Home Exam is to practice on functions and if-else statements.

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.

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

Description

In this Take-Home Exam, 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. The inputs that define the exercise (minutes spent in that exercise, how many repetitions done and for weight lifting only, the amount of weight that person lifted) will be asked in a different function. After necessary inputs for each exercise are gotten from the user, a second prompt will be received that wants the user to enter his/her weight in terms of kilograms in the main function. 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, lifted weight, repetitions 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!
Lifted weight <=0 or >35	Lifted weight out of range!
reps <0 OR reps > 50	Reps out of range!
minute < 0 OR minute > 2000	Minute out of range!

Note that the order of the input checks is important. Firstly, the weight should be checked. If it is out of range, you should not continue with the calculations. If the weight is valid, then the lifted weight should be checked. Again, if it is out of range, then your program should not continue with the calculations. If the lifted weight is valid, then the reps should be checked. And lastly if the reps are correct, minutes 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 doing lunges, then push ups, pull ups and weight lifting.

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(Metabolic Equivalent for Task) values of each exercise, which are as indicated in the table below:

Exercise	**Reps**	**MET**
Lunges	<15	3
Lunges	>=15, <30	6.5
Lunges	>=30	10.5
Push-ups	<15	4
Push-ups	>=15	7.5
Pull-ups	< 25	5
Pull-ups	>= 25	9

We use a different table and calculation for the weight lifting, since the calories burned is also related to how much weight a person is lifting. Although the table is a little different than other activities, the formula to calculate the calories burned is the same.

Exercise	**Lifted Weight**	**Reps**	**MET**
Weight Lifting	<5kg	<20	3
Weight Lifting	<5kg	>=20, <40	5.5
Weight Lifting	<5kg	>=40	10
Weight Lifting	>=5kg, <15kg	<20	4
Weight Lifting	>=5kg, <15kg	>=20, <40	7.5
Weight Lifting	>=5kg, <15kg	>=40	12
Weight Lifting	>=15kg	<20	5
Weight Lifting	>=15kg	>=20, <40	9
Weight Lifting	>=15kg	>=40	13.5

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.

calories burned = Duration (in minutes)*(MET*3.5*weight in kg)/200

The above formula gives the calorie burn for a specific exercise with a given MET value. After calculating calories burned in each exercise, 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 do lunges ***minuteCalculation*** minutes more to reach your goal or,
You need to do push ups ***minuteCalculation*** minutes more to reach your goal or,
You need to do pull ups ***minuteCalculation*** minutes more to reach your goal.
You need to do weight lifting ***minuteCalculation*** minutes more to reach your goal."

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

VERY IMPORTANT!

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

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

Prompts for inputs and outputs must be **exactly the same** 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.

IMPORTANT!

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

See Sample Runs section for input/output examples.

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-0, you were not supposed to implement any functions. However, in this THE you are expected to (actually you have to) use some function(s). The guidance about using functions in this THE is below.

A total of 6 user-defined functions (other than the main) **MUST** be implemented in this THE. On top of these functions, you may of course use other functions, if you want. The function explanations are stated as below:

1. ***main:*** In your main function, first prompt the user for their name and weight. After this, call the ***getInput*** function for each exercise type which are mentioned in the tables above. When you have every value needed, then call the ***inputCheck*** function for each exercise. If all of the input checks are successful, lastly call the ***computeResults*** function.

2. ***getInput*:** A void function that takes exercise (string, denoting the type of exercise), mins, reps and liftedWeight as parameters and asks the user to enter the mins, reps and liftedWeight (if applicable) values. Since this is a void function, the entered values must be returned to the main function using **reference** parameters.

3. ***inputCheck*:** A non-void function (returns true/false result to the caller) that takes the ***reps***, ***weight***, ***minute*** and the ***liftedweight*** 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. The ***liftedweight*** parameter uses a default value of 1, for the exercises except for weight lifting, so that in the function call you don't have to pass this parameter to the function.

4. ***calculateMET*:** A void function (returns the MET value using reference) that takes ***reps, exercise*** and ***MET*** as its parameters, and determines what the MET value is depending on the repetitions. 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 reps, and then determine the MET.

5. ***calculateWeightLiftMET***: A void function (returns MET value using reference) that takes ***reps***, ***liftedWeights*** and ***MET*** as its parameters, and determines what the MET value is depending on these parameters. The parameter reps is of type `int` and represents repetitions of exercise, parameter ***liftedweight*** is of type `double`. In if-else statements, you can first check the weight range, then reps, and then determine the MET.

6. ***computeResults*****:** A void function that takes ***weight***, ***goal***, ***repsLunge***, ***repsPushUp***, ***repsPullUp***, ***repsWeightLift***, ***minLunge***, ***minPushUp***, ***minPullUp***, ***minWeightLift*** and ***liftedWeight*** 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*** or ***calculateWeightLiftMET*** function from this function, separately for each exercise. After all four 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.

7. ***displayResults*****:** A void function that takes ***difference***, ***total***, ***weight***, ***lungeMET***, ***pushupMET***, ***pullupMET***, ***weightliftMET***, ***lungeCalorie***, ***pushupCalorie***, ***pullupCalorie*** and ***weightliftCalorie*** as its parameters and displays the required outputs. 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.

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: Gulsen
Welcome Gulsen, please enter your weight(kg): 60
Gulsen, please enter minutes and repetitions in a week for the
activities below.
Lunges: 40 20
Push Ups: 30 15
Pull Ups: 30 20
Gulsen, please enter minutes, repetitions and lifted weight in a week
for the activities below.
Weight Lifting: 30 50 15
Gulsen, please enter your weekly calorie burn goal: 1092
From lunges, you burned 273 calories.
From push ups, you burned 236.25 calories.
From pull ups, you burned 157.5 calories.
From weight lifting, you burned 425.25 calories.
You burned 1092 calories.
Congratulations! You have reached your goal!
```

Sample Run 2

```
Please enter your name: Baris
Welcome Baris, please enter your weight(kg): 80
Baris, please enter minutes and repetitions in a week for the
activities below.
Lunges: 30 40
Push Ups: 35 50
Pull Ups: 25 50
Baris, please enter minutes, repetitions and lifted weight in a week
for the activities below.
Weight Lifting: 35 45 15
Baris, please enter your weekly calorie burn goal: 1500
From lunges, you burned 441 calories.
From push ups, you burned 262.5 calories.
From pull ups, you burned 441 calories.
From weight lifting, you burned 661.5 calories.
You burned 1806 calories.

You have surpassed your goal! You can eat something worth 306 calories
:)
```

Sample Run 3

Please enter your name: **Sevval**
Welcome Sevval, please enter your weight(kg): **70**
Sevval, please enter minutes and repetitions in a week for the activities below.
Lunges: **20 40**
Push Ups: **30 45**
Pull Ups: **20 30**
Sevval, please enter minutes, repetitions and lifted weight in a week for the activities below.
Weight Lifting: **20 50 8**
Sevval, please enter your weekly calorie burn goal: **1500**
From lunges, you burned 257.25 calories.
From push ups, you burned 183.75 calories.
From pull ups, you burned 330.75 calories.
From weight lifting, you burned 294 calories.
You burned 1065.75 calories.

You did not reach your goal by 434.25 calories.
You need to do lunges 33.7609 minutes more to reach your goal or,
You need to do push ups 47.2653 minutes more to reach your goal or,
You need to do pull ups 39.3878 minutes more to reach your goal or,
You need to do weight lifting 29.5408 minutes more to reach your goal.

Sample Run 4

Please enter your name: **Berfin**
Welcome Berfin, please enter your weight(kg): **25**
Ece, please enter minutes and repetitions in a week for the activities below.
Lunges: **45 15**
Push Ups: **35 20**
Pull Ups: **20 20**
Ece, please enter minutes, repetitions and lifted weight in a week for the activities below.
Weight Lifting: **20 20 5**
Ece, please enter your weekly calorie burn goal: **1400**
Weight out of range!

Sample Run 5

```
Please enter your name: Furkan
Welcome Furkan, please enter your weight(kg): 75
Furkan, please enter minutes and repetitions in a week for the
activities below.
Lunges: 2500 40
Push Ups: 75 50
Pull Ups: 35 20
Furkan, please enter minutes, repetitions and lifted weight in a week
for the activities below.
Weight Lifting: 10 10 10
Furkan, please enter your weekly calorie burn goal: 1800
Minute out of range!
```

Sample Run 6

```
Please enter your name: Nasim
Welcome Nasim, please enter your weight(kg): 55
Sevval, please enter minutes and repetitions in a week for the
activities below.
Lunges: 20 45
Push Ups: 33 10
Pull Ups: 20 44
Sevval, please enter minutes, repetitions and lifted weight in a week
for the activities below.
Weight Lifting: 40 50 40
Sevval, please enter your weekly calorie burn goal: 1500
Lifted weight out of range!
```

Sample Run 7

```
Please enter your name: Kenan
Welcome Kenan, please enter your weight(kg): 85
Sevval, please enter minutes and repetitions in a week for the
activities below.
Lunges: 20 100
Push Ups: 25 50
Pull Ups: 15 40
Sevval, please enter minutes, repetitions and lifted weight in a week
for the activities below.
Weight Lifting: 20 45 3
Sevval, please enter your weekly calorie burn goal: 1300
Reps out of range!
```

General Rules and Guidelines about THEs

The following rules and guidelines will be applicable to all take-home exams, unless otherwise noted.

How to get help?

You can use GradeChecker (http://learnt.sabanciuniv.edu/GradeChecker/) to check your expected grade. Just a reminder, you will see a character ¶ which refers to a newline in your expected output.

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

What and Where to Submit

You should prepare (or at least test) your program using MS Visual Studio 2012 C++ (Windows users) or using Xcode (macOS users).

It'd be a good idea to write your name and last name in the program (as a comment line of course). Do not use any Turkish characters anywhere in your code (not even in comment parts). If your name and last name is "İnanç Arın", and if you want to write it as comment; then you must type it as follows:

// Baris Altop

Submission guidelines are below. Since the grading process will be automatic, students are expected to strictly follow these guidelines. If you do not follow these guidelines, your grade will be 0.

- Name your submission file as follows:
  - 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_THEnumber.cpp"
  - Your SUCourse user name is actually your SUNet username, which is used for checking sabanciuniv emails. Do NOT use any spaces, non-ASCII and Turkish characters in the file name (**use only lowercase letters**). For example, if your SUCourse username is "**altop**", then the file name should be: **altop_the1.*cpp*** (please only use lowercase letters).
  - Do not add any other character or phrase to the file name.
- Please make sure that this file is the latest version of your take-home exam program.
- Submit your work **through SUCourse only**! You can use GradeChecker only to see if your program can produce the correct outputs both in the correct order and in the correct format. It will not be considered as the official submission. You must submit your work to SUCourse. You will receive no credits if you submit by any other means (email, paper, etc.).
- If you would like to resubmit your work, you should first remove the existing file(s). This step is very important. If you do not delete the old file(s), we will receive both files and the old one may be graded.

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, you should also use the same text as given in the "Sample Runs" section. Otherwise, the automated grading process will fail for your take-home exam, 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 take-home exam before the deadline. Please be careful that SUCourse time and your computer time may have 1-2 minutes differences. You need to take this time difference into consideration.
- Successful submission is one of the requirements of the take-home exam. If, for some reason, you cannot successfully submit your take-home exam 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 or on the course website.

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 take-home exam from the email address provided in the comment section of your announced take-home exam grade or attend the specified objection hour in your grade announcement.

- Check the comment section in the take-home exam tab to see the problem with your take-home exam.
- Download the 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.

Good Luck!
Şevval Şimşek & Gülşen Demiröz & Barış Altop