

Hacettepe University
Department of Computer Engineering
BBM104 Introduction to Programming Laboratory II
Programming Assignment 2

Submission Date : 21.03.2018
Due Date : 11.04.2018
Programing Language : JAVA
Title : Calorie Calculation for Healthy Life
Advisor : Dr. Gönenç Ercan, Dr. Öner Barut, Dr. Cumhuriyet Özcan, Dr. Ali Seydi Keçeli,
R.A. Nebi Yılmaz

INTRODUCTION

In this experiment you are expected to gain knowledge on basic JAVA programming. The program you are going to develop will deal with variables, loops, string operations, file read and write operations. Besides the programming task, you will also learn to comply with coding standards.

1. Problem Definition

In this experiment you are expected to write Java code that calculates the calories by considering taken and burned calories during the day for the healthy life of the people. You will be given three text files as follows:

1.1 Text for information of people (people.txt)

This text file includes personal information of each person, which are person ID (**personID**), name (**name**), gender (**gender**), weight (**weight**), height (**height**) and date of birth (**dateOfBirth**) as shown in following table. Every item in the file separated with a **tab** character. This text file contains up to 50 items.

| |
|---|
| [person ID] tab [name] tab [gender] tab [weight] tab [height] tab [date of birth] newline |
| [person ID] tab [name] tab [gender] tab [weight] tab [height] tab [date of birth] newline |

Example content of person.txt

| | | | | | |
|-------|-------|--------|----|-----|------|
| 12345 | ahmet | male | 78 | 175 | 1987 |
| 12346 | ahmet | male | 92 | 189 | 1990 |
| 12378 | gizem | female | 61 | 172 | 1986 |
| | | | | | |

1.2 Text for food (food.txt)

This text file includes information of foods, which are food ID (**foodID**), name of food (**nameOfFood**) and calorie count (**calorieCount**) as shown in the following table. Every item in the file separated with a **tab** character. For each food, 1 portion is 100 grams and the calorie count in the table is calculated for 1 portion. ID of fruits groups start with 10.., ID of meal groups start with 11.., ID of dessert groups start with 12.. and they consist of a 4-digit number. This text file contains up to 100 items.

| | | | | | |
|-----------|-----|----------------|-----|-----------------|---------|
| [food ID] | tab | [name of food] | tab | [calorie count] | newline |
| [food ID] | tab | [name of food] | tab | [calorie count] | newline |

Example content of food.txt

| | | |
|-------|-----------|-----|
| 1001 | apple | 57 |
| 1101 | spaghetti | 131 |
| 1102 | lahmacun | 185 |
| | | |
| 1201 | baklava | 521 |
| | | |

1.3 Text for sport activities (sport.txt)

This text file includes information of sport, which are sport ID (**sportID**), name of sport (**nameOfSport**) and calorie burned (**calorieBurned**) as shown in the following table. Every item in the file separated with a **tab** character. The calories burned for each sport are calculated for 60 minutes. ID of sport activities start with 20.. and they consist of a 4-digit number. This text file contains up to 100 items.

| | | | | | |
|------------|-----|-----------------|-----|------------------|---------|
| [sport ID] | tab | [name of sport] | tab | [calorie burned] | newline |
| [sport ID] | tab | [name of sport] | tab | [calorie burned] | newline |

Example content of sport.txt

| | | |
|-------|----------|-----|
| 2001 | swimming | 400 |
| 2002 | running | 300 |
| | | |
| 2013 | tennis | 275 |
| | | |

2. Calculation of daily calorie needs

The daily calorie needs (**dailyCalorieNeeds**) of people vary by gender, age, height and weight. Therefore, it will be calculated separately for men and women as follows:

$$\text{Calculation for Men} = 66 + (13.75 \times \text{weight (kg)}) + (5 \times \text{height (cm)}) - (6.8 \times \text{age})$$

$$\text{Calculation for Women} = 665 + (9.6 \times \text{weight (kg)}) + (1.7 \times \text{height (cm)}) - (4.7 \times \text{age})$$

The daily calorie needs (**dailyCalorieNeeds**) should always be rounded to the closest integer value.

3. Text for input (command.txt)

Each line of the input file named as command.txt consists of either person ID (**personID**), food ID (**foodID**) and the number of portions (**numberOfPortion**), or person ID (**personID**), sport ID (**sportID**) and sport duration (**sportDuration**) as shown in the table below. During day, a person may add food ID that is eaten and sport ID that is done into this file. The **print(personID)** command should write the current calorie status of the specified person in command.txt file to monitoring.txt file. The **printList** command should write calorie statuses of all people given in command.txt file to monitoring.txt file. The expected output format is given in section 4.

| |
|---|
| [person ID] tab [food ID] tab [number of portions] newline |
| [person ID] tab [sport ID] tab [sport duration] newline |
| |
| print (personID) newline |
| [person ID] tab [sport ID] tab [sport duration] newline |
| printList newline |
| |

Example content of command.txt

```
12345 1001 2
12378 1002 3
.....
print (12345)
12345 2001 45
12378 1001 1
printList
.....
```

4. Text for output (monitoring.txt)

You are expected to write output of your program to a text file named as monitoring.txt for persons specified in command.txt file. This text file should include the following information for each person in order as shown in the following table: name (**name**), age (**age**), daily calorie needs (**dailyCalorieNeeds**), calories taken (**caloriesTaken**), calories burned (**caloriesBurned**) and result (**result**) for print (personID) and printList. If the result is a number less than zero, it means that a person has taken less calories than they should take during a day. On the other hand, if the result is greater than zero, a person has taken more calories than they should take during a day. The result (**result**) should always be rounded to the closest integer value. Also, the output file should include **person ID (personID)**, calories taken, name of food (**nameOfFood**), calories burned and name of sport (**nameOfSport**) to keep track calories burned and taken for a given person in input file. Every item in the file separated with a **tab** character

```
[person ID] tab has tab taken tab [calories taken]kcal tab from tab [name of food] newline
***** (There will be 15 stars ) newline
[person ID] tab has tab burned tab [calories burned]kcal tab thank to tab [name of sport] newline
***** (There will be 15 stars ) newline
[name] tab [age] tab [daily calorie needs] tab [calories taken] tab [calories burned] tab [result] newline
***** (There will be 15 stars ) newline
[name] tab [age] tab [daily calorie needs] tab [calories taken] tab [calories burned] tab [result] newline
***** (There will be 15 stars ) newline
... ..
```

Example content of monitoring.txt

```
12345 has taken 200kcal from apple
*****

12356 has burned 100kcal thanks to tennis
*****

ahmet 27 1897kcal 2300kcal 400kcal +3kcal
*****

ahmet 27 1897kcal 2300kcal 400kcal +3kcal
gizem 25 1789kcal 1900kcal 430kcal -319kcal
*****

.....
```

5. Example content of input and output file

In this experiment, you will be given an input file (command.txt) as below and you are expected to create an output file as shown below (monitoring.txt) by considering this given input file. The values in the example content of files given above (section 1.1, 1.2 and 1.3) are taken into consideration in this input and output files.

command.txt

```
12345 1102 4
12378 1101 3
print (12345)
12345 2001 4
printList
12378 1001 1
```

monitoring.txt

```
12345 has taken 740kcal from lahmacun
*****
12378 has taken 393kcal from spaghetti
*****
ahmet 31 1803kcal 740kcal 0kcal -1063kcal
*****
12345 has burned 300kcal thanks to swimming
*****
ahmet 31 1803kcal 740kcal 300kcal -1363kcal
gizem 32 1393kcal 393kcal 0kcal -1000kcal
*****
12378 has taken 57kcal from apple
```

Execution and Test

You will use the Java Platform as described in the. The input files (command.txt) should be given as an argument. Upload your java files to your server account (dev.cs.hacettepe.edu.tr)

- Upload your java files to your server account (dev.cs.hacettepe.edu.tr)
- Compile your code (javac *.java)
- Run your program (java Main command.txt)
- Control your output file (monitoring.txt) and format.

Submit Format

File hierarchy must be zipped before submitted (Not .rar, only .zip files are supported by the system)

<student id>.zip

<src>

- Main.java, *.java

<report>

- Report.pdf (It will have same format with homework 1)

Late Policy

You have three days for late submission. You will lose 10 points from maximum evaluation score for each day (your submitted study will be evaluated over 90, 80 and 70 for each late submission day). You have to submit your solution in deadline date + three days, otherwise it will not be evaluated.

Notes and Restrictions

- Save all your work until the assignment is graded.
- Using collection classes (arraylist, vector, hashmap, etc) are not allowed. Only use of array is allowed.
- The names of classes, attributes and methods should obey to Java naming convention.
- All assignments must be original, individual work. Duplicate or very similar assignments are both going to be considered as cheating.