CS201 – Fall 2021-2022

Take-Home Exam 4
– Market Database –
Due December 15, Wednesday, 23:55 (Sharp Deadline)

Introduction

The aim of this take-home exam is to make students comfortable with streams (operating on file streams and string streams using library functions) in C++ as well as using computational approaches to solve analytical questions.

Description

First let us start with a brief definition of the Database concept. A database is a collection of data records stored in a computer in a systematic way. In this way, a computer program can be used to answer queries about the data stored in the database by searching the records. In this homework, you will write a console application that would take two database files and a shop list file as input and **display the output on console** according to the user inputs. Your program will take the database file names as input to open and read the files according to the shop list file. These two database files will contain QR codes of items, their names and their prices. This is similar to a supermarket database in a primitive way. Your shop list file refers to the amount of goods that will be bought. Its name is **"shopList.txt"** and it contains QR codes of items and their amounts. You can read and use it directly or try different amounts to check if your code works properly. While reading each line in "shopList.txt" your program will search two database files for matching QR code and retrieve the item name and price. As a result your program will print the item names, number of items and prices on the console and also calculate the tax and total value of the items, which will also be printed into the console.

You **cannot** write all of your code under the main function, i.e. you **must** write user-defined functions and use them.

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 (**all** your .cpp and .h files for this take-home exam). Additionally, you should submit all of your files to SUCourse (**all** your .cpp and .h files 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, e.g. gulsend_THE4.cpp). Please check the submission procedures of the take-home exam, which are listed at the end of this document.

Structure of Database Files and Shop List

The database files are same in organization, but hold different information. The two files will both contain the exact same QR codes for each item. First file will have the matching item name for each QR code. The second file will contain the matching prices of each QR coded item. For the shop list, it will have the matching amounts of each QR coded item. However, a QR code in the shop list may not have an exact match in database files. In that case, you will dismiss that line and continue processing.

Name of both database files will be entered by the user of your program. At the beginning of your program, user will be asked to enter an input file name for the item name database records. If user enters a name for a file that does not exist, your program should display an appropriate message and ask for the name of a new input file name until the file is successfully opened. After the first input file is opened successfully, your program will ask for another input file name for the item price records file. The same file opening control will be performed for this file too. If a user enters a name for a file that does not exist, your program should display an appropriate message and ask for the name of a new input file name until the file is successfully opened.

Item Name Database File

This file contains the name of each item with its corresponding QR code. Each record line contains these two information, however the item name can be a combination of multiple words. Below you can see a structure of the file:

```
QR_Code     Item_Name
QR_Code     Item_Name Can_Be Multiple_Words
```

The rules for this file are as follows:

- Record line starts with the QR code. QR code can contain letters and/or numbers

- The second part of the record line stores the item name, which can be a combination of multiple words.

Item Price Database File

This file will contain the price of each item with its corresponding QR code. Each record line contains these two pieces of information. Below you can see a structure of the file:

```
QR_Code      Price_of_Item
```

The rules for this file are as follows:

- Record line starts with the QR code.
- The order of the QR codes can be different from the Item Name File, but it will contain the same codes.
- The second part of the record line stores the item's price, which is a real number.

Shop List File

This file will contain the amount of each item that should be bought with its corresponding QR code. Each record line contains these two pieces of information. Below you can see a structure of the file:

```
QR_Code      Amount_of_Item
```

The rules for this file are as follows:

- Each line starts with the QR code.
- QR codes may or may not match with other database files.
- The second part of the line stores the corresponding item's amount, which is an integer.

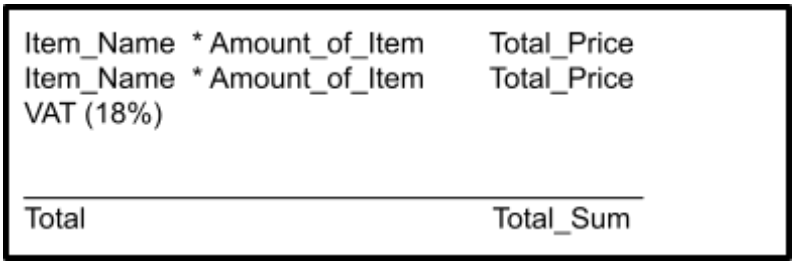
There might be any number of white spaces (i.e. blanks) between and after each data item in all files. You should handle it. There is no specific order of records in both input files. That means you cannot assume that the records of the input files are ordered according to QR_Code , Item_Name or any other data attribute. The structure of the input files and the associated rules and assumptions explained above are fixed such that you cannot change them or make any other assumptions. No format check for the contents of the input files is needed. You may assume that all data are in the correct format. You may

examine the sample input files (shopList1.txt, shopList2.txt, shopList3.txt, qr.txt, prices.txt) provided in the zip package for this homework.

Program Flow and Outputs

First of all, your program will ask for the name of the item name file. It should continue to ask the name filename until the file can be opened successfully. Once the name file is successfully opened, your program will ask for the price list filename. It should also continue to ask the price filename until the file can be opened successfully. After both filenames are correctly entered your program will ask for the name of the third file for the shopping list, you may assume the shop list file name will always be entered correctly. You will write all the outputs according to the content of this file on the console window.

Once all three files are opened correctly your program will read the shop list as it gives QR codes together with the count. After you read a QR code, your program should check if this code exists in the item name database. If it does not exist you must skip that line and continue processing. If the QR code exists in the item name database then your program should then take the item name and its total price (using price and count) and print it to the console window. Your output on the console window will look like this:



The total price of each item will be calculated according to the number of items and its price. In other words, *Total_Price = Item_Price * Item_Count*.

Database Search

Database search will basically be searching the both files iteratively against the read QR codes from the shop list. Your program should read each record line into a string file and should then parse the string into its parts. In each file the first part of a record line will be the QR code and you should search the matching QR code. If the QR exists you must retrieve the remaining part as item name from the item name database file. Only after that you must search the second file and

calculate the total price, the price list file with the same QR code. Finally, you will have to do three important steps.

- You must write the item name, item count and total price to the console.
- You must keep track of the sum of all bought items.
- You must return to the first line of both item name and item price files.
- You must set the spaces of the output correctly. All prices and total sum should be aligned.

Input Entry and Input Check

- Item name filename input: The name filename must be entered correctly.
- Item price filename input: The price filename must be entered correctly.
- QR code can have both numbers and uppercase letters. The shop list may contain lowercase letters, your program should handle it.
- The shop list can contain wrong QR codes, you should dismiss it.
- Number of items bought will be an integer. Assume only integers will be entered.
- Number of each item must be between 1-25.

Some Hints

Maybe you have already realized that the simplest way of processing the input file is reading the data line by line and parsing each line. In order to perform such a processing, the idealm echanism is to use a combination of getline function and input string streams (substr, find and at member functions do not help you too much in this homework). We have given/will give examples of such a processing in the lecture and in recitations. When you finish searching and processing a QR code, you must return to the first record line of both input files. To achieve this you must use both the clear() and seekg(int) functions.

Important Remarks

We will not specify the number of functions for this THE. But you are expected to use functions to avoid code duplication and 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. Please do not write everything in main and**

then try to split the task into some functions just to have some functions other than mail. This is totally against the idea of functional design and nothing but a dirty trick to get some points. Instead please design your program by considering the necessary functions at the beginning.

Try to use parametric and non-void functions wherever appropriate. Do NOT use any global variables (variables defined outside the functions) to avoid parameter use.
In this homework (and in the coming ones) you are not allowed to use instructions such as "exit" and "goto". These cause difficulties to control the flow of your programs. Thus, we do not approve of using them. You are also not encouraged to use "break" and "continue". The use of "break" and "continue" prevent you from forming good readable loop conditions and hence prevent you from learning how to form good loops. Think cleverly in order not to use any of these instructions. If you don't know these commands, do not even try to learn them (we will explain "break" in class).

IMPORTANT!

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

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 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 some sample runs of the program that you will develop. The *italic* and **bold** phrases are inputs taken from the user. You have to display the required information in the same order and with the same words and characters as below.

Sample Run 1

```
Please enter a filename for QR database: q.txt

The QR file does not exists
Please enter a filename for QR database: r.txt

The QR file does not exists
Please enter a filename for QR database: qr.txt

Please enter a filename for Price database: p.txt

The Price file does not exists
Please enter a filename for Price database: priec.txt

The Price file does not exists
Please enter a filename for Price database: price.txt

Please enter a filename for your shopping list: shopList1.txt

Kinder Bueno *  2               2.38

VAT(18%):                     0.4284
---------------------------------------
Total:                        2.8084
```

Sample Run 2

```
Please enter a filename for QR database: qr.txt

Please enter a filename for Price database: price.txt

Please enter a filename for your shopping list: shopList2.txt

Kinder Bueno *  2               2.38
Coca Cola (500ml) * 12          21.6
Wi-Fi Router *  1              243.79

VAT(18%):                     48.1986
---------------------------------------
Total:                        315.969
```

Sample Run 3 (Use "shopList3.txt")

```
Please enter a filename for QR database: qr.txt

Please enter a filename for Price database: price.txt

Please enter a filename for your shopping list: shopList3.txt

Coca Cola (330ml) *  5          6.15
Coca Cola (500ml) * 12          21.6
Albeni * 23                    18.17
LG 32" LED TV *  1             799.9

VAT(18%):                     152.248
---------------------------------------
Total:                        998.068
```

General Rules and Guidelines about Homeworks

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!
Berker Demirel & Gülşen Demiröz & Barış Altop