Assignment 5 - Menu Generator

Due Mar 7 by 11:59pm **Points** 20 **Submitting** a file upload **File Types** h and cpp **Available** until Mar 10 at 12:01am

This assignment was locked Mar 10 at 12:01am.

ST. CHARLES COMMUNITY COLLEGE

CPT-182 - Programming in C++

Programming Assignment - Menu Generator (**20** Points)

(Number in Question Bank: Assignment **5.1**)

Program Overview

On February 1, 2022, the Hell's Kitchen series was renewed for a twenty-first and twenty-second season. The twenty-first season, subtitled **Battle of the Ages**, aired September 2022 - February 2023.

Suppose you are working for the boss, Chef Gordon Ramsay, your job is to create an efficient system that generates a menu. Since in Hell's Kitchen, the menu changes every day, so your system should easily add items to the menu. Therefore, **`vector`** would be an excellent solution.

The `Dish` Class

You need to write a class called **`Dish`**, which contains the following data fields and class-member functions.

- A string variable called **`dish_name`**, which represents the name of the dish (e.g., **`"Chicken Parmesan"`**).

  You can assume that the length of **`dish_name`** is **no more than 60** characters. However, please note that **`dish_name`** may contain spaces.
- An unsigned integer variable called **`dish_type`**, which represents whether the **`Dish`** is an appetizer (value **`0`**), entrée (value **`1`**), or dessert (value **`2`**).
- An unsigned integer variable called **`dish_price`**, which represents the price (in USD) of the dish.

  In this assignment, the price is always a positive integer.
- A default constructor (**no** arguments).
- A constructor that takes in **3** arguments, which are the **name**, **price**, and **type** of the **`Dish`**.

  The two constructors can be combined into one constructor using **default parameter values**.
- The **getters** and **setters** for all the data fields.
- A method called **`print_dish`** that takes an output stream as its only argument. The method writes the current **`Dish`** object to the output stream with the following format.

  [dish_name] ($[dish_price])

  For example, if a dish's name is "Chicken Parmesan" and its cost is **`11`** dollars, then the output format of the dish should be **`"Chicken Parmesan ($11)"`**.
- For each method (class member function) in this **`Dish`** class, it is your responsibility to correctly determine whether the method should be **`const`** method, the return type of the method, and whether the arguments should be passed by value, by reference, or by **`const`** reference.

- You **must** define the class in a header file, and implement the class in another `.cpp` file. Lots of **unprofessional** C++ code online writes everything in one file, which is **not** allowed in your assignment. You should do the same thing I did in the lecture. Otherwise, your work will be sharply penalized in grading.

The Input File

- The input file is a **plain text file** (filename: `dishes.txt`).
- Each line of the input file contains exactly **3** data fields, which are the type of the dish (`"Appetizer"`, `"Entree"`, or `"Dessert"`), price of the dish (always a positive integer), and name of the dish (may contain spaces), in that order.
- You **cannot** assume (or guess) the number of dishes stored in the input file. In other words, no matter how many dishes are stored in the input file, your program should correctly process all of them.
- You can assume that there is at least **1** appetizer, **1** entrée, and **1** dessert in the input file.
- Please refer to the **sample input file** to better understand the input file format.

The Output File

- The output file is a **plain text file** (filename: `menu.txt`).
- The output file begins with **`"Menu"`** in the first line, followed by an empty line.
- Then, write **`"Appetizer"`** followed by an empty line to the output file.
- Then, all the appetizers stored in the input file are listed (one dish per line). An empty line should appear after all the appetizers are listed.
- Then, write **`"Entree"`** followed by an empty line to the output file.
- Then, all the entrées stored in the input file are listed (one dish per line). An empty line should appear after all the entrées are listed.
- Then, write **`"Dessert"`** followed by an empty line to the output file.
- Then, all the desserts stored in the input file are listed (one dish per line).
- Please refer to the **sample output file** to better understand the output file format.

Other Development Notes

- In this program, you probably need to use both the stream extraction operator (`">>"`) and the **`getline()`** function.
- A string may contain **unwanted** whitespaces at the beginning and/or at the end. It is your responsibility to remove those whitespaces before writing the string to the output file.
- After reading a line in the input file, you need to create a **`Dish`** object based on the data read in.
- The simplest solution is to create **3** **`vector`**s to store the appetizers, entrées, and desserts. Please think what should be the data type of the **`vector`** (`vector<?>`).
- You only need to separate appetizers, entrées, and desserts in the output file. For the dishes in appetizers, entrées, and desserts, you do **not** need to sort them.

Sample Input and Output File (Click to Download)

Sample Input File (https://drive.google.com/uc?export=download&id=1cn9Pyqnj-TKAIVMu3g7cNOzdKQuUm1IQ)Sample Output File

Assignment Submission and Grading (Please Read)

- Please upload all your `.h` (if any) and `.cpp` files (**not** the ~~entire Microsoft Visual Studio project folder~~) on Canvas.
- Before the assignment deadline, you can submit your work unlimited times. However, only your latest submission will be graded.
- At least **20**% of your code should be **comments**. All variable, function (if any), and class (if any) names should "make good sense". You should let the grader put **least effort** to understand your code. Grader will **take off points**, even if your program passes all test cases, if he/she has to put extra **unnecessary** effort to understand your code.
- Please **save a backup copy** of all your work in your computer hard drive.
- Your program will be graded (tested) using another valid input file (still named `dishes.txt`) to check whether it can generate the expected (correct) output file (with correct format and correct output values in it). As long as the input file is valid, your program should generate a correct output file. In other words, your program should work for **any** valid input file, **not** just the sample input files provided in the assignment instructions.

- In this class, you can assume that the input file (input data) is always **valid** and **has correct format**. You do **not** need to deal with ~~invalid input~~ or ~~error handling~~.
- Your work will be graded after the assignment deadline. All students will receive their assignment grades at (almost) the same time.