C COMPUTER PROGRAMMING (II) (1072) : Midterm Exam II

Cheng-Wen Ko.Ph.D, Associate Professor
E-Mail: cwko@cse.nsysu.edu.tw
Department of Computer Science and Engineering
National Sun Yat-sen University

May 27. 2019, 18:00:~ 21:00

General instructions:

1. Do not open this exam until you are told to begin.
2. Please turn off all cell phones and remove all headphones.
   手機請關機，並禁止使用任何耳機。
3. No USB disk or any portable device can be used.
   禁止使用任何可攜式電腦裝置以及USB 隨身碟。
4. You can only use the text editor for Ubuntu to edit your programs. You will lose 40 points for each time you are caught to use editors/IDEs for windows.
   你只能使用Ubuntu內建之文字編輯器進行程式編輯。每一次被發現使用Windows之 IDE 或文字編輯程式將扣此次考試成績40分。
5. This exam has 3 pages including this cover. There are 4 questions.
6. You don't need to provide the "makefile", yet you are encouraged to do so. Alternatively, you can compile your code with these commands:
   ```
   g++ -c Q1.cpp
   g++ -o Q1.out Q1.o
   ```
7. Compress your programs (*.cpp and *.out) for all of the questions and name the compressed file with your student ID (example: B053040001.zip).   Be sure that this file is transferred successfully to TA before you leave.
   作答完成後請將所有程式檔（*.cpp 以及 *.out ）壓縮成一個檔案，並且以學號命名。學號第一個字母請大寫。離開前請務必跟助教確認檔案已傳輸成功。

1. (25%) Complete the program “Q1.cpp” that manages a list of **up to** 5 players and their high scores. Write a class named `Player` to store a player’s name and score. Be sure to include constructors with this class that sets the name and score. A default construct is also required. In the main function, a vector of the `Player` class is used to store the 10 players. The following features are managed by four functions that you should complete:
   - `addPlayer` – Add a new player and score (up to 5 players).
   - `printPlayer` – Print all players and their score to the screen.
   - `searchPlayer` – Allow the user to enter a player name and output that player’s score or a message if the player’s name has not been entered.
   - `removePlayer` - Allow the user to enter a player name and remove the player from the list.

   Please note that you are not allowed to modify the main function. Only three predefined libraries (iostream, string, vector) can be used in this code.
   You may use the following member functions of vector (but not necessarily):
   ```
   size_type size();
   void push_back(const value_type& val);
   void pop_back();
   ```

2. (25%) Create a class named `Subscriber` that has three member variables:
   - `name` – A string that stores the name of the subscriber.
   - `numChannels` – An integer that tracks how many channels the subscriber subscribes to.
   - `channelList` – A dynamic array of strings used to store the names of the channels that the subscriber subscribes to.

   Write appropriate constructor, mutator, and accessor functions for the class along with the following:
   - A default constructor.
   - `InputData` - A function that inputs all values from the user, including the list of channel names. This function will have to support input for an arbitrary number of channels.
   - `OutputData` - A function that outputs the name and list of all channels.
   - `ResetChannels` - A function that resets the number of channels to 0 and the channelList to an empty list.
   - An overloaded assignment operator that correctly makes a new copy of the list of channels.
   - A destructor that releases all memory that has been allocated.

   A `main` function that tests all these functions are provided in “Q2.cpp”. You are not allowed to change this part of the code. Only 2 predefined libraries (iostream, string) can be used in this code.

   Hint: Recall that `cin >> variable` leaves a newline `'\n'` in the buffer. This can be a problem if you are mixing `cin >> variable` and `getline`. Use `cin.ignore` to discard the newline. Usage:
   ```
   getline(cin, stringVariable);
   cin.ignore(NUMBER_TO_IGNORE, DESIGNATED_CHAR);
   ```

3. (25%) Write a BoxOfProduce class that stored several bundles of fruits or vegetables to ship to a customer. Use a vector instead of an array and add appropriate constructors, mutators, and accessors. The class should have a function to add additional fruits or vegetables to the box so there could be more than three bundles in one box. The output function should output all items in the box. Overload the + operator to return a new BoxOfProduce object that combines the vector contents of both operand BoxOfProduce objects. You don't have to implement or change the main function, only to complete the definition and implementation of BoxOfProduce class.

   Please note that you are not allowed to modify the main function. Only three predefined libraries (iostream, string, vector) can be used in this code.

   You may use the following member functions of vector (but not necessarily):

   ```
   size_type size();
   void push_back(const value_type& val);
   void pop_back();
   void clear();
   ```

4. (25%) One problem with dynamic arrays is that once the array is created using the `new` operator the size cannot be changed. For example, you might want to add or delete entries from the array similar to the behavior of a `vector`. Try to create a class called `DynamicStringArray` that includes member functions to emulate the behavior of a `vector` of strings.

   The class should have:

   - A private member variable called `dynamicArray` that references a dynamic array of type string.
   - A private member variable called `size` that holds the number of entries in the array.
   - A default constructor that sets the dynamic array to `NULL` and sets `size` to 0.
   - A function that returns `size`.
   - A function named `addEntry` that takes a string as input. The function should create a new dynamic array one element larger than `dynamicArray`, copy all elements from `dynamicArray` into the new array, add the new string onto the end of the new array, increment `size`, delete the old `dynamicArray`, and then set `dynamicArray` to the new array.
   - A function named `deleteEntry` that takes a string as input. The function should search `dynamicArray` for the string. If not found, return `false`. If found, create a new dynamic array one element smaller than `dynamicArray`. Copy all elements except the input string into the new array, delete `dynamicArray`, decrement `size`, and return `true`.
   - A function named `getEntry` that takes an integer as input and returns the string at that index in `dynamicArray`. Return `NULL` if the index is out of `dynamicArray`'s bounds.
   - A copy constructor that makes a copy of the input object's dynamic array.
   - Overload the assignment operator so that the dynamic array is properly copied to the target object.
   - A destructor that frees up the memory allocated to the dynamic array.

(End of the Exam)