Lab 8

CSE 165: Object Oriented Programming
Spring 2022
(100 points)

This programming assignment has five tasks, complete each task as instructed. Write a separate file for each of the following tasks. To submit your assignment, please organize your code in the folder ”Lab8” by placing your code in its corresponding sub-folder. For example, store your code for task 1 in the following directory ”Lab8/1/”. Then, submit the compressed version of folder Lab8 to CatCourses. Submissions must arrive by one minute before the lab section of week 11 (3/28 – 4/1). All of the files you need for this programming assignment are available in a ZIP archive file called “Lab8.zip”.

1. **Abstract Data types (5 Points)**

   File `ADT.h` contains an abstract data type called `ADT`. Create a type named `Derived` which inherits from `ADT`. Save your `Derived` class in a file named `Derived.h`. To get an idea about the implementation of the methods in Derived, study the file `ADTs.cpp`, which has an `ADT` pointer pointing to an instance of `Derived`.

   Sample output from ADTs.cpp

   ```
   I did something
   I did something else
   ```

2. **Integer Sorting (10 Points)**

   Your task is to write a class called `Data`, stored in a file named `Data.h`. Your class should be able to store a collection (vector) of integers. In addition to the appropriate constructors, your class should also have the following methods.

   `void add (int number);` // Adds a number to the data set.

   `void print ();` // Prints out the entire data set on a single line, separated by space.

   `void sort ();` // Sorts the data set in ascending order. You may implement any sorting algorithm here, for example, max sort, bubble sort, insertion sort, merge sort, quick sort, etc... There is no need to try to implement the most efficient one, any one will do.

   Make sure your class works as expected with the file `intSort.cpp`.

   Sample output from intSort.cpp

   ```
   Input:    4 5 2 3 1
   Output:   1 2 3 4 5
   ```

3. **Sortable Objects (15 Points)**

   Study the file `Sortable.h`. It contains class template `Sortable` which inherits from class template `vector<T>` and adds sorting to it. Write a driver code for `Sortable.h` in a .cpp file to create Sortable objects to store the input data from sample input files. The first input file contains integer numbers, the second input file contains characters and the third one has strings in every line. Then call method `sort` from `Sortable` and print out the sorted values.

   Output

   Output1: 1 2 3 4 5 6 7 8 9

   Output2: a b c d e f g

   Output3: a big dog is running

4. **Sortable Objects II (15 Points)**

   Create a header file called `Matrix.h` and implement a class template called `Matrix` which inherits from `Sortable`. Each `Matrix` object is in fact a `Sortable` vector that has two variables row and column to store the size of the matrix. The class template `Matrix` has a constructor `Matrix(int row, int col)` to initialize the `Matrix` object by setting its size. It also has methods `print` and `resize`. The method `print()` uses the object's row and column values to print out the vector in matrix format, and `bool resize(int row)` resizes the matrix by setting new values for row and column. As input, `resize` only gets a new value for row and it should find the proper value for column. `resize` should return False if it is not possible to resize the original matrix to fit the new row size, and return True otherwise.

   You need to write a driver code in a .cpp file to create one `Matrix` object for storing numbers and another object for characters. You need to ask the user to enter the row and column size, and generate **random** numbers to fill the first object and and generate **random** characters to fill the second object. Then for each object call `print` once before calling `sort` and once after it. Finally, ask the user for a new row size, then print out the resized matrix.

   To generate random numbers you can check this link.

Sample Output

Numbers:

```
sample row size: 3
sample column size: 2
Before Sort:
          14 10
          12 13
          11 15

After Sort:
          10 11
          12 13
          14 15

New number of rows: 6

Resized matrix:
          10
          11
          12
          13
          14
          15
```

Characters:

```
sample row size: 4

sample column size: 5

Before Sort:
          b v e i h
          k g c j t
          d f y a o
          n u r x q


After Sort:
          a b c d e
          f g h i j
          k n o q r
          t u v x y


New number of rows: 2


Resized matrix:
          a b c d e f g h i j
          k n o q r t u v x y
```

5. **OpenGL (55 Points)**

In this lab, we will learn how to create OpenGL window using Qt framework.

Task 1: 2D Graphics

Download the zip folder "QT_OPENGL_Example1.zip" and extract the content. Now, open Qt Creator and load QT_OPENGL_Example1.pro. Study `mainwindow.h`, `mainwindow.cpp` and `main.cpp`. Try to change code and see the changes by running the project.

Hint: To understand code, watch the tutorial below

a. https://youtu.be/a2CpOQsgB84

b. https://youtu.be/6eu8oMULqaU

c. https://youtu.be/ITalCqf4Ifc

Task 2: 3D Graphics

Download the zip folder `QT_OPENGL_Example2.zip` and extract the content. Now, open Qt Creator and load the QT_OPENGL_Example2.pro. Study `mainwindow.h`, `mainwindow.cpp` and `main.cpp`. Try to change code and see the changes by running the project.

Hint: To understand code, watch the tutorial below:
https://youtu.be/mENP56CmeVw

Task 3: 3D Graphics

Your task is to use `QT_OPENGL_Example2.zip` and draw a Tetrahedron. You need to create an animation such that the tetrahedron rotate around z axis and along x-y plane. Your project should be named as Tetrahedron_Animation. You need to submit the entire project folder as zip file.

As part of your demo, you will be asked questions about Task1, Task2 and Task3 code. So, prepare yourself when you demo.