Programming Assignment #4/5
CS 162: Introduction to Computer Science

Submit your assignment to the D2L Dropbox
Email a backup copy to karlaf@pdx.edu

The purpose of the 4th and 5th programs is to implement the new concepts learned which include (a) pointers, (b) dynamically allocated arrays, and (c) linear linked lists. Our goal is to continue to create programs with classes, functions (w/ arguments) and a small main function that delegates to a series of functions where the real work takes place. **Place your class interface in a .h file and the class implementation in a .cpp file. Your submission should have at least 3 files!**

In this programming assignment, you are **not** allowed to use global variables. You are allowed (as usual) to use the cstring library (e.g., strlen, strcpy, and strcmp). Limit your main (and all functions) to no more than 30 statements of code (for executable statements… *not counting variable definitions, blank lines, lines with just curly brackets, or comments*).

Program Assignment:
Oregon has so many great opportunities in the summer for great activities. Whether it is hiking, fishing, boating, wind surfing, kayaking, or rock climbing, there are many options on how to be active all within a 1-2 hours of Portland. But after living her awhile, it is easy to forget all that is available so close. When have I hiked up to the Pittock Mansion? Not for a few years. Or, fished the trask river? Gee not for about 20+ years.

Your job for this last 2 assignments is to build a **class** called "Activity" that will keep track of a single activity that is available within Oregon (or Washington if you prefer). All arrays of characters used in class Activity need to be dynamically allocated! It will keep track of the following information at a minimum:

1. Type of Activity (e.g., Wind surfing)
2. Location of the Activity (e.g., Columbia River Gorge)
3. The season - When best to do the Activity (e.g., which season is best)
4. Description of your last experiences doing this
5. The last time you did this activity (e.g., a date)

Once this is working, build a class called "Adventure" that will keep track of all of the activities. You will need to use a dynamically allocated array of Activities as the class' data member. You will also need a data member to keep track of the number of activities in this adventure.

The following is a suggested set of classes. You may add more or modify the arguments or functions:

```
class Activity
{
    public:
                Activity ();
                ~Activity();   //A destructor used to deallocate dynamic memory

                //add a description or date to an activity (you may change the args!)
                void Add_Info(char more_info[], int month, int year);
                void Read(); //read in a new activites information from the user

                //Only displays the information if the season matches
                void Display(char season[]);

                //Displays all of the activity's information
                void Display();
   private:
      /*Put a dynamically allocated arrays here */
};

class Adventure
{
    public:
                Adventure ();
                ~Adventure();   //A destructor used to deallocate dynamic memory
                void New_Activity(); //Read in a new activity's information to the list
                void Display_all (char season[]);

                void Display_all();
   private:
      /*Put a dynamically allocated array of activities, an integer count of the
number of activities, and an integer for the size of the array created */
};
```

******You are always welcome to do more! But, really focus on making general purpose functions that can be re-used. Anytime you have code that has already***

existed elsewhere in your program (such as to error check input or give the user another chance), write a function instead!

In Program #5 we will store the activities in a linear linked list!

You are welcome to use external data files, but they are not required.

Things you should know...as part of your program:

1. Make sure to prompt the user for any input requested. Make sure it is clear from your prompts what the user is expected to do.
2. Have main test out all of the functions to make sure they work for all conditions.
3. The program should continue until the user wants to quit. Allow them to continue until they are done.
4. Make sure to put your name in your program
5. Submit an electronic copy of your .cpp file as an attached file to the dropbox on D2L (go to: http://d2l.pdx.edu/ to login). Make sure to hit the submit button after uploading your files (otherwise they will be lost)