CMPE-50 Object-Oriented Concepts and Methodologies, Tarng, Spring 2021
Homework #4

Due: 4/20/2021 Tuesday midnight

The submission of the homework should be the cpp files with the output in the code comment. Each problem needs to have a complete program, meaning, it needs to contain the main() function and some test code and data. Therefore, you need to submit six cpp files in total. Do not zip the files. Name the files in the following way: CMPE50-HW-4-1.cpp, CMPE50-HW-4-2.cpp, etc. If the solution includes some input files, also submit the input files.

Total 40 points

1. [10 pts] (Chapter 9: Pointer)
   Write a program that allows user to enter an array of integer numbers and then reverses this array of integer numbers. This program works for arrays of any size by using a dynamic array to hold the numbers. The user needs to first enter the size of the array then enter the numbers to the array elements. We need a function that takes an integer array and the size as input parameters and reverses the array. The function should use two pointers, `front` and `rear`. The `front` pointer should initially reference the first number in the array, and the `front` pointer should initially reference the last number in the array. Reverse the array by swapping the numbers referenced by `front` and `rear`, then increment `front` to point to the next number and decrement `front` to point to the preceding number, and so on, until the entire array is reversed. Write a main program to test your function on various sizes of arrays, both even and odd length. Deallocate the dynamic array at the end of the main function.

2. [10 pts] (Chapter 9: Pointer) Enhance the following StringVar class by adding:
   - member functions copy_piece, which returns a substring (of type StringVar);
   - member function one_char, which returns a specified single character; and
   - member function set_char, which changes a specified character
   - An overloaded version of the == operator (note that only the string values have to be equal; the values of max_length need not be the same)
   - An overloaded version of + that performs concatenation of strings of type StringVar
   - An overloaded version of the extraction operator >> that reads one word (as opposed to input_line, which reads a whole line)

```
class StringVar
{
public:
        StringVar(int size);
        StringVar( );
        StringVar(const char a[]);
        StringVar(const StringVar& string_object);
        ~StringVar( );

        int length( ) const;
        void input_line(istream& ins);
        StringVar copy_piece(int pos, int npos);
        char one_char(int pos);
        void set_char(int pos, char ch);
        friend ostream& operator <<(ostream& outs, const StringVar& the_string);
        friend istream& operator >>(istream& ins, StringVar& the_string);
        friend bool operator==(const StringVar& sv1, const StringVar& sv2);
        friend StringVar operator+(const StringVar& sv1, const StringVar& sv2);
private:
        char *value; //pointer to dynamic array that holds the string value.
        int max_length; //declared max length of any string value.
};
```

3. [10 pts] (Chapter 11: Class Inheritence and Operator Overloading) Give a definition of a class Manager for managers who have direct reports. The class Manager is a derived class of SalariedEmployee and it needs to add a new member variable reports which is a dynamic array of object type SalariedEmployee to represent the the list of direct report employees. You need to define the following data and function members:
    - A variable called reports, which is a pointer of type SalariedEmployee, and is used to hold an array of direct reports.
    - A vairalbe called noReport, which is of type int, to store the number of direct reports.
    - Default constructor and copy constructor
    - Assignment operator =
    - A destructor
    - A member function addReport to add a new report, i.e., a SalariedEmployee object, to the reports array. In addReport, you need to allocate a new array of SalariedEmployee of size noReport+1, copy the old array to the new array, then add the new member to the end of the new array. Then delete the old array and copy the new array to reports.
    - An overloaded operator << to print the list of direct reports to the output stream.

4. [10 pts] (Chapter 15: Polymorphism) You are running a pet clinic and need a program to manage the pets. Create a base class Pet. We need to store the following information, with the type in parenthesis, in the base class Pet: name (string), age (int), dates of visits (string pointer). Since the dates of visits can grow, we need to use a string pointer to a dyamic array of strings and each date is recorded in a string object in the following format" "mm/dd/yyyy". In addition, we need a class Dog that is a derived class of Pet.

The derived class Dog contains the following member variables: breed (string) and dates of vaccination shots (string pointer). The type of dates of vaccination shots is the same as dates of visits and we should use a string pointer to a dynamic array of strings. You need to define the following member and friend functions for Pet:

- Constructor
- Virtual Destructor
- Accessors and mutators
- Virtual function print to print the name, age, and dates of visits
- Copy constructor
- Overloaded assignment operator =

Also define the following member functions for Dog:

- Constructor
- Virtual destructor
- Accessors and mutators
- Virtual function print to print name, age, dates of visits, breed, and dates of shots
- Copy constructor
- Overloaded assignment operator =

Test your program. Write a function PrintBill that takes an argument of a reference to Pet and print out the pet's information. Decalre a Dog object, fill in the necessary information, and call PrintBill.