Lab 9

Due Date: Nov 1, 2019
Total Points: 15 points

The purpose of this lab is to understand pointers and dynamic memory allocation.

Part 1: Introduction to Pointers

Questions 1-3 are paper and pencil, Question 4 is a programming question

1. Consider the following statements:

```
int *p;
int i, k;
i = 50;
k = i;
p = &i;
```

After these statements, which of the following statements will change the value of i to 20?

a) k = 20;

b) *k = 20;

c) p = 20;

d) *p = 20;

e) Two or more of the answers will change i to 20

2.What is the output of the following code segment?

```
int value1 = 5, value2 = 15;
int *p1, *p2;
p1 = &value1;
p2 = &value2;
*p1 = 10;
*p2 = *p1;
p1 = p2;
*p1 = 20;
cout << "value1==" << value1 << "/ value2==" << value2;
```

3. What is the output of the following code segment?

```
double values[6] = {10, 20, 30, 40, 50, 60};
double *valptr= values;
cout << 3*values[1] + valptr[3] + *(valptr + 2);
```

4. Fill in the code below as indicated in the comments. This program finds the area of a rectangle using **pointer variables**. It also prints the rectangle dimension (length and width) in ascending order.

```
#include <iostream>
using namespace std;

int main()
{
   int length;     // holds length
   int width;      // holds width
   int area;       // holds area

   int *lengthPtr = nullptr;  // int pointer to point to length
   int *widthPtr = nullptr;   // int pointer to point to width


     // prompt the user to enter length and width
     // then make lengthPtr & widthPtr point to length & width
     // respectively


     // find and print the area using only the pointer variables


     // compare length and width using only the pointer variables
     // and print them in ascending order


   return 0;
}
```

Part 2: Dynamic arrays

Write a program that will read monthly sales of a certain company into a **dynamically allocated array of double values**.

Your program should:

- prompt the user to enter the size of the array (that is the number of monthly sales)
- dynamically allocate an array large enough to hold the number of monthly sales given by the user
- find and print the yearly sum of all the monthly sales

Note: don't forget to deallocate memory!

Sample run:

```
Enter the number of monthly sales to be input: 4
Enter the monthly sales for month 1: 1290.89
Enter the monthly sales for month 2: 905.95
Enter the monthly sales for month 3: 1567.98
Enter the monthly sales for month 4: 994.83
The total sales for the year is: $4759.65
```

Part 3: Array of pointers to objects

In this part, you need to reuse the class *Course* from lab 4. Recall that a Course class has the following private attributes: courseNumber, courseName, and number of credits. It also has the following public methods:

- Default constructor that sets courseNumber to 0, courseName to “”, and number ofCredits to 0.
- Overloaded constructor that takes 3 parameters used to initialize the 3 attributes.
- Set function(s) that set the objects data.
- Print function that displays the object data in a neat fashion.

Write a main program that:

a) Declares an array of pointers to Course objects. You may assume the array size is 3.
b) Makes the array point to 3 new Course objects (use the new operator in a for loop).
c) Calls the class method set to set the attributes of these objects to the following:

```
70503 CS211 4
70507 CS231 4
70509 CS331 3
```

d) Calls the class method print to display the data of each Course in a neat table format. For example,

```
Number      Name        Credits
---------------------------------
70503       CS211       4
70507       CS231       4
70509       CS331       3
```