Lab 02: Classes, Objects, and Object Based Programming

Section 1: Guess program outputs.

1.

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

class Rectangle
{
      private:
            double length;
            double width;
      public:
            void setLength(double len)
            {  length = len; }

            void setWidth(double wid)
            {  width = wid; }

            double getLength()
            {  return length; }

            double getWidth()
            {  return width; }

            double getArea()
            {  return length * width; }
};

class Carpet
{
      private:
            double pricePerSqYd;
            Rectangle size;             // size is an instance of
                                        // the Rectangle class
      public:
            void setPricePerYd(double p)
            {  pricePerSqYd = p; }

            void setDimensions(double len, double wid)
            {  size.setLength(len/3);   // Convert feet to yards
               size.setWidth (wid/3);
            }

            double getTotalPrice()
            {  return (size.getArea() * pricePerSqYd); }
};

// ************** Client Program *****************
int main()
{
      Carpet purchase;                // This variable is a Carpet object
      double pricePerYd;
      double length;
      double width;

      cout << "Room length in feet: ";
      cin  >> length;
      cout << "Room width in feet : ";
```

```
      cin  >> width;
      cout << "Carpet price per sq. yard: ";
      cin  >> pricePerYd;

      purchase.setDimensions(length, width);
      purchase.setPricePerYd(pricePerYd);

      cout << "\nThe total price of my new " << length << " x " << width
           << " carpet is $" << purchase.getTotalPrice() << endl;

      return 0;
}
```

2.

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

class SimpleStat
{
  private:
      int largest;            // The largest number received so far
      int sum;                // The sum of the numbers received
      int count;              // How many numbers have been received

      bool isNewLargest(int); // This is a private class function

  public:

      SimpleStat();           // Default constructor
      bool addNumber(int);
      double calcAverage();

      int getLargest()
      {  return largest; }

      int getCount()
      {  return count; }
};

// SimpleStat Class Implementation Code

/************************************
 *  SimpleStat Default Constructor   *
 ************************************/
SimpleStat::SimpleStat()
{
      largest = sum = count = 0;
}

/************************************
 *      SimpleStat::addNumber        *
 ************************************/
bool SimpleStat::addNumber(int num)
{   bool goodNum = true;
      if (num >= 0)              // If num is valid
      {
            sum += num;            // Add it to the sum
            count++;               // Count it
            if(isNewLargest(num))  // Find out if it is
                  largest = num;     // the new largest
```

```
	}
	else                          // num is invalid
		goodNum = false;

	return goodNum;
}

/************************************
 *     SimpleStat::isNewLargest     *
 ************************************/
bool SimpleStat::isNewLargest(int num)
{
	if (num > largest)
		return true;
	else
		return false;
}

/************************************
 *      SimpleStat::calcAverage     *
 ************************************/
double SimpleStat::calcAverage()
{
	if (count > 0)
		return static_cast<double>(sum) / count;
	else
		return 0;
}

// Client Program

/************************************
 *                main              *
 ************************************/
int main()
{
	int num;
	SimpleStat statHelper;

	cout << "Please enter the set of non-negative integer \n";
	cout << "values you want to average. Separate them with \n";
	cout << "spaces and enter -1 after the last value. \n\n";

	cin >> num;
	while (num >= 0)
	{
		statHelper.addNumber(num);
		cin >> num;
	}
	cout << "\nYou entered " << statHelper.getCount() << " values. \n";
	cout << "The largest value was " << statHelper.getLargest()  <<
endl;
	cout << "The average value was " << statHelper.calcAverage() <<
endl;

	return 0;
}
```

Section 2: Review Questions and Exercises

1. What does ADT stand for?

2. Which of the following must a programmer know about an ADT to use it?
a) What values it can hold
b) What operations it can perform
c) How the operations are implemented

3. The two common programming methods in practice today are ____ and ____ .

4. _____ programming is centered around functions, whereas _____ programming is centered around objects.

5. An object is a software entity that combines both ____ and ____ in a single unit.

Section 3: Programming Challenges

1. Date
Design a class called Date that has integer data members to store month, day, and year. The class should have a three-parameter default constructor that allows the date to be set at the time a new Date object is created. If the user creates a Date object without passing any arguments, or if any of the values passed are invalid, the default values of
1, 1, 2001 (i.e., January 1, 2001) should be used.
The class should have member functions to print the date in the following formats:
3/15/16
March 15, 2016
15 March 2016
Demonstrate the class by writing a program that uses it. Be sure your program only accepts reasonable values for month and day. The month should be between 1 and 12.
The day should be between 1 and the number of days in the selected month.

2. Report Heading
Design a class called Heading that has data members to hold the company name and the report name. A two-parameter default constructor should allow these to be specified at the time a new Heading object is created. If the user creates a Heading object without passing any arguments, "ABC Industries" should be used as a default value for the company name and "Report" should be used as a default for the report name. The class should have member functions to print a heading in either one-line format, as shown here:
Pet Pals Payroll Report
or in four-line "boxed" format, as shown here:

```
**************************************************
                    Pet Pals
                 Payroll Report
**************************************************
```

Try to figure out a way to center the headings on the screen, based on their lengths.
Demonstrate the class by writing a simple program that uses it.