2143 OOP - Test 3

Name: _____

#	Possible	Earned	#	Possible	Earned
1	10		7	10	
2	10		8	10	
3	10		9	10	
4	10		10	10	
5	10		11	10	
6	10		12	10	
				120	

Instructions

- Use pencil only
- Write your name at the top of all pages turned in (unless they are stapled).
- When stapling pages together, do it at the extreme top left corner.
- Make sure your pages are in order, with questions also in order.
- Handwriting that is illegible (messy, small, not straight) will lose points.
- Indentation matters. Keep code aligned correctly.
- All answers will be written on the paper provided, and not directly on the test.
- When possible, write your answers vertically and mark clearly:

> A) answer one
> B) answer two
> C) answer three

- NOT

> a.answer one b.answer two c.answer three

- Failure to comply will result in **loss of letter grade**.

Question 1

- A) _____ Is the object definition;
- B) _____ Is the object;
- C) The data members `name` and `id` are:
  - a) public
  - b) private
  - c) protected
  - d) private and protected
  - e) a b & c

```
class person {
        char name[20];
        int id;
        void getdetails(){}
};

int main() {
        person p1;
}
```

Question 2

A) True / False : The structure below provides an adequate example of encapsulation.

```
struct Fraction {
        int numerator;
        int denominator;

        void set(int n,int d)
        {
                numerator = n;
                denominator = d;
        }

        void print()
        {
                cout<<numerator<<"/"<<denominator<<endl;
        }
};
```

B) What minimal changes to the struct above would you have to make to implement proper **data abstraction** or **implementation hiding**.

Question 3

Polymorphism means to have many forms. What this means in OOP is that we override and overload methods in our classes. There are two distinct categories of polymorphism: **compile time** and **run time**. Look at the list below and on your answer sheet indicate which type they are: **C** for compile time, **R** for run time, or **B** for both.

- A) Operator overloading.
- B) Method overloading.
- C) Method overriding.

Question 4

```
class Vehicle {
public:
        Vehicle()
        {
        cout << "This is a Vehicle" << endl;
        }
};
class FourWheeler {
public:
        FourWheeler()
        {
        cout << "This is a 4 wheeler Vehicle" << endl;
        }
};
class Car: public Vehicle, public FourWheeler {

};

// main function
int main()
{
        Car obj;
        return 0;
}
```

What is the output of the inheritance example above?

Question 5

- Explain what abstraction means from an OOP standpoint.
- Does storing data in a linked list vs an array have any relation to abstraction? Explain.

Question 6

To add two fractions you need to:

- Find a common denominator by finding the LCM (Least Common Multiple) of the two denominators.
- Change the fractions to have the same denominator and add both terms.
- Reduce the final fraction obtained into its simpler form by dividing both numerator and denominator by the largest common factor.

Assume you have the following class that has all of those listed methods implemented for you.

```
class fraction{
        int numerator;
        int denominator;
        fraction reduce(fraction f);
        int  lca,int b);
public:
        fraction(int n,int d);
        void setumerator(int n);
        void setDenominator(int d);
};
```

Overload the + sign to add two fractions. You can assume your defining your method inline. Assume all the methods above are implemented. Just write the overloaded method ... nothing else.

Question 7

- A) All class members declared as _____ will be available to everyone.
- B) This access modifier is similar to one of the other access modifiers, the difference is that the class member declared as _____ are inaccessible outside the class but they can be accessed by any subclass (derived class).
- C) Only member functions or _____ _____ of another class are allowed to access the private data members of this class.
- D) The _____ data members of this class can be accessed from anywhere in the program using the dot operator.
- E) The class members declared as _____ can be accessed only by the functions inside the class.

Question 8

```
class base
{
public:
        void fun_1() { cout << "base-1\n"; }
        virtual void fun_2() { cout << "base-2\n"; }
        virtual void fun_3() { cout << "base-3\n"; }
        virtual void fun_4() { cout << "base-4\n"; }
};

class derived : public base
{
public:
        void fun_1() { cout << "derived-1\n"; }
        void fun_2() { cout << "derived-2\n"; }
        void fun_4(int x) { cout << "derived-4\n"; }
};

int main()
{
    base *p;
    derived obj1;
    p = &obj1;

    p->fun_1();  // Question A
    p->fun_2();  // Question B
    p->fun_3();  // Question C
    p->fun_4();  // Question D
    p->fun_4(5); // Question E

    // Question F: are the three lines below valid? State why or why not.
    derived *p1;
    base p2;
    p1 = &p2;
}
```

For questions A-F write on your answer sheet what prints out, or if it errors and why.

Question 9

A **copy constructor** is automatically created by the compiler when needed. However there are certain times a **copy contructor** needs to be created by the user. When we create our own copy constructor, especially of we have pointers (assume a linked list), we need to make sure we do it correctly. Given the class below:

```
class LinkedList{
        nodePtr *head;
        int size;
        LinkedList(){
                head = null;
                size = 0;
        }
};
```

Using the above class definition, write a copy constructor for the ***LinkedList*** class that will initialize an instance of a linked list with a copy another linked list. Just write the constructor as if it were inline.

Question 10

When discussing the concept of friendship between class we must remember that:

- A) The purpose of making class **X** a friend of class **Z** is so you give **X** access to **Z**'s _____.
- B) However the friendship between **X** and **Z** is not _____.
- C) And friendship is also not _____.

Question 11

Write a function called `countMe` that counts the number of times it has been called.

Question 12

Write a class called `NumObjects` that counts the number of objects in existance. You should assume that each object will be created and destroyed before your program ends. So your count should be equal to the number of existing objects.