2143 OOP - Final Exam

Name: ________________________________

#	Possible	Earned
1	10	
2	20	
3	10	
4	10	
5	15	
6	15	
7	15	
	100	

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**.

Multiple Choice 01:

```
class A {
    friend class B;
};
```

1. Looking at the above snippet, which of the following statements is true?

- A) The friendship declaration, above, has the same effect as making B a subclass of A, since B now can access A's protected members.
- B) If B is already a subclass of A, the friendship declaration has no effect.
- C) The friend declaration gives B greater access to A's members than if B were a subclass of A.
- D) None of the above

2. The signature of the standard `<< operator` for printing an integer is:

- A) `ostream &operator<<(ostream &, const int)`
- B) `ostream &ostream::operator<<(ostream &, const int)`
- C) `int &operator<<(ostream &, const int)`
- D) `int int::&operator<<(ostream &, const int)`

3. Which of the following is true about virtual functions in C++.

- A) Virtual functions are functions that can be overridden in derived class with the same signature.
- B) Virtual functions enable run-time polymorphism in a inheritance hierarchy.
- C) If a function is 'virtual' in the base class, the most-derived class's implementation of the function is called according to the actual type of the object referred to, regardless of the declared type of the pointer or reference. In non-virtual functions, the functions are called according to the type of reference or pointer.
- D) All of the above

4. Which of the following is true about pure virtual functions? 1) Their implementation is not provided in a class where they are declared. 2) If a class has a pure virtual function, then the class becomes abstract class and an instance of this class cannot be created.

- A) Both 1 and 2
- B) Only 1
- C) Only 2
- D) Neither 1 nor 2

5. Which one of the following is correct, when a class grants friend status to another class?

- A) The member functions of the class generating friendship can access the members of the friend class.
- B) All member functions of the class granted friendship have unrestricted access to the members of the class granting the friendship.
- C) Class friendship is reciprocal to each other.
- D) There is no such concept

6. If a function is friend of a class, which one of the following is wrong?

- A) A function can only be declared a friend by a class itself.
- B) Friend functions are not members of a class, they are associated with it.
- C) Friend functions are members of a class.
- D) It can have access to all members of the class, even private ones.

7. Which of the following operators cannot be overloaded

- A) `.` (Member Access or Dot operator)
- B) `?:` (Ternary or Conditional Operator)
- C) `::` (Scope Resolution Operator)
- D) `.*` (Pointer-to-member Operator)
- E) All of the above

8. If there is a virtual function in a base class and another class is derived from our base class. Which of the following are true.

- A) You have to implement the virtual function in the derived class.
- B) You are not allowed to implement the virtual function in the derived class.
- C) You must use the virtual keyword in the derived class.
- D) You don't necessarily have to override the virtual method.

9. What is the maximum number of base classes you can inherit from:

- A) 1
- B) 2
- C) No limit you will every worry about.
- D) None of the above

10. ____________ is the packing of data and functions operating on that data into a single component and restricting the access to some of the object's components and ___________ is a mechanism which represent the essential features without including implementation details.

Question 02:

#	Word	Choice	Definition
1	Inheritance	A	The redefinition of base class function in its derived class with same signature.
2	Friend	B	The capability of a class to derive properties and characteristics from another class.
3	Inheritance	C	Supports an important feature of Object Oriented Programming known as Data Hiding.
4	Overloading	D	Preserves their value even after they are out of their scope.
5	Private	E	Overriding an operator for a class is an example of _________.
5	Instance	E	All class members declared as __________ will be available to everyone.
6	Polymorphism	F	It provides multiple definitions of the function by changing signature (parameters).
7	Composition	G	Can access private members of other class in which it is declared.
8	Static	H	A member function which is declared within base class and must be re-defined by derived class.
9	This	I	A member function of which initializes objects of a class.
10	Pure Virtual	J	A constant pointer that holds the memory address of the current object.
11	Protected	K	A call to a member function will cause a different function to be executed depending on the type of object that invokes the function.
12	Encapsulation	L	A member function which is declared within base class and is re-defined by derived class.
13	Public	M	A way to combine simple objects or data types into more complex ones.
14	Override	N	Only member functions or __________s of another class are allowed to access the private data members of this class.
15	Abstraction	O	The __________ data members of this class can be accessed from anywhere in the program using the dot operator.
16	Protected	P	The class members declared as __________ can be accessed only by the functions inside the class.
17	Constructor	Q	Enables new objects to take on the properties of existing objects.

Question 03:

Write a small snippet of code that shows you understand encapsulation.

Question 04:

Given:

```
class Rodent{
private:
    string Order;
    string Family;
    string Subfamily;
public:
    Rodent(){
      Order = "Rodentia";
      Family = "Muridae";
    };
    virtual string getOrder(){ return Order;}
    virtual string getFamily(){ return Family;}
    virtual string getSubfamily() = 0;
};

class Gerbil : private Rodent{

public:
    Gerbil(){
      Subfamily = "Gerbillinae";
    };

    string getSubfamily(){return Subfamily;}
};

int main(){
  Gerbil G;
  cout<<G.getSubfamily()<<endl;
  return 0;
}
```

Does this code error? And if it does, how can you fix it?

Question 05:

```
class NumCollection{
    int *Nums;
    int Size;
public:
    NumCollection(int size=25){
        Nums = new int[size];
        Size = size;
    }
    void addNum(int x){// inserts num at next spot}
    int getNum(){// removes num from last spot}
    int getSize(){return Size;}
}
```

Overload the modulus operator so that one "NumCollection" can be applied to another one.

Question 06:

Create a simple “shape” hierarchy: a base class called `Shape` and derived classes called `Circle`, `Rectangle`, and `Triangle`. In the base class, make a **pure virtual function** called `draw()`. Data members for each shape are needed for each derived class (e.g. Circle needs radius, triangle needs ....). You do not have to implement setters and getters for all the data members.

Question 07:

Write a class that contains two class data members `numBorn` and `numLiving`. The value of `numBorn` should be equal to the number of objects of the class that have been instanced. The value of `numLiving` should be equal to the total number of objects in existance currently (ie, the objects that have been constructed but not yet destructed.)
