CS 485 Exam 1 Review Topics

All the material in the Shalloway book. **Vocabulary** in particular is important.

C++

- Constructors
- Destructors
- Dynamic Memory
- CopyAndSwap
- Inheritance
- Virtual/Pure Virtual
- CopyAssignmentOperator
- Polymorphism

Any topic addressed via source code in CS485_Student_Examples.

UML

Be able to produce and read a UML diagram comparable to the UML diagram for Bank or Chadd's AlarmClock

Design

- Define encapsulation
- Given a UML diagram, explain if that is a good design. In particular, explain what is being encapsulated.
- SOLID
- Composition/Aggregation/Inheritance
- What is an interface?
- Cohesion & Coupling
- Explain "program to the abstract".
- Abstract vs Concrete.

Questions!

0. What is the guiding principle of design?
1. Describe two advantages of using design patterns.
2. Give an instance where a design encapsulates data.
3. Give an instance where a design encapsulates functionality.
4. Explain the Open/Closed principle.
5. Why is tight coupling a bad idea? Explain how tight coupling is in opposition to "isolate what may change."
6. Give an example in Chadd's AlarmClock UML where code depends on the abstract and not the concrete. Explain how this may benefit the developer as the requirements change in the future.
7. State the Rule of Three. Explain why the Rule of Three is a good idea.
8. We said that a better definition of an object is "data and responsibilities." We also said to avoid get and set functions as much as possible.
   - Why is giving an object responsibilities a good idea, design-wise?
   - Why are get and set methods a bad idea, design-wise?

9. Which constructors, destructor, or member functions are called on each line of main() below?

```
class Person
{
  public:
     Person();
     Person(std::string cName);
     Person(const Person & rcOther);
     ~Person();

     Person& operator=(Person cOther);
};

int main()
{
   Person cMe("Chadd");
   Person cInstructor = cMe;
   Person cYou;

   cYou = "Your Name";

   cMe = cInstructor;

   return 0;
}
```

10. Does Person::operator= use the CopyAndSwap idiom? How do you know?

Imagine the following line of code invokes a CopyAndSwap based operator=.

X = Y;
Assume:

1. X contains dynamically allocated memory.
2. Y contains dynamically allocated memory.
3. After operator=, X points to newly allocated dynamic memory which contains a copy of the data in Y.

11. Which function allocates the new dynamic memory pointed to by X (referred to in assumption 3) ?
12. Which function deallocates the original dynamic memory pointed to by X (referred to in assumption 1)?
13. What does `Person(const Person &rcData) = delete;` mean? Why would you do this?
14. Why is composition generally preferred over inheritance?
15. Define polymorphism.
16. What is an interface?
17. What is a function signature?
18. In CS 250 we said inheritance was a way to reuse code. In 485 we said inheritance is a way to specialize a class. Explain what is meant by each statement.

19. When does a virtual function get invoked? Build a very small example to demonstrate when a virtual function is invoked.
20. Explain the virtual friend idiom.
21. Are the following two statements equivalent?
 const int * pData;
 int const * pData;

22. What issues do raw pointers have?
23. How do smart pointers try to address the issues from 22?
24. Why does Valgrind not work with Visual Studio?
25. Define and explain the bad design principles: Rigidity, Fragility, Immobility.