CS 485 Exam 2 Review Topics

All the material in the Shalloway book

Vocabulary in particular is important.

C++

Enums
Exceptions
Namespace
STL
Double Dispatch
Lambda

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.

Design

Strategy Pattern
Factory Method/Abstract Factory Pattern
Singleton
Visitor
Observer
MVP

1. Why would you want to use the Factory Method pattern? What is being encapsulated? Why is this important to encapsulate? Discuss how the Factory method cuts down on repeated code.

2. What are the pros and cons of using a static function as a Factory? What are the pros and cons of using a new class to contain a virtual function that is used as a Factory? Does the Factory Method pattern violate the Open/Close principle? Why?

3. Why is the Singleton pattern often disparaged as just a big global variable? What difficulties can arise from using a Singleton? Why was a Singleton a good solution to the Currency Converter or why was a Singleton a bad solution to the Currency Converter? Justify your answer.
4. How can Templates in C++ increase encapsulation?

5. Why should we prefer a non-member, non-friend function to either a friend function or a member function? Be sure to discuss encapsulation and abstraction in your answer.

6. What is double dispatch? Why does the visitor pattern need to implement double dispatch? How does the Visitor pattern in C++ implement double dispatch?

7. Why might you choose to throw an exception rather than just return false; to indicate an error occurred in a function?

8. How is a lambda like a function pointer? How is a lambda different than a function pointer?

9. What is a functor?

10. What is an iterator? How does the use of an iterator increase encapsulation?

11. What is a type-safe container? Why was our CS 300 linked list (which contained a void*) not type-safe?

12. Templates are resolved at compile time. Why does this make finding bugs easier?

13. Templates don't require the type supplied to be derived from a specific base class. How does a the compiler determine if a particular type is appropriate for use with a particular templated function?

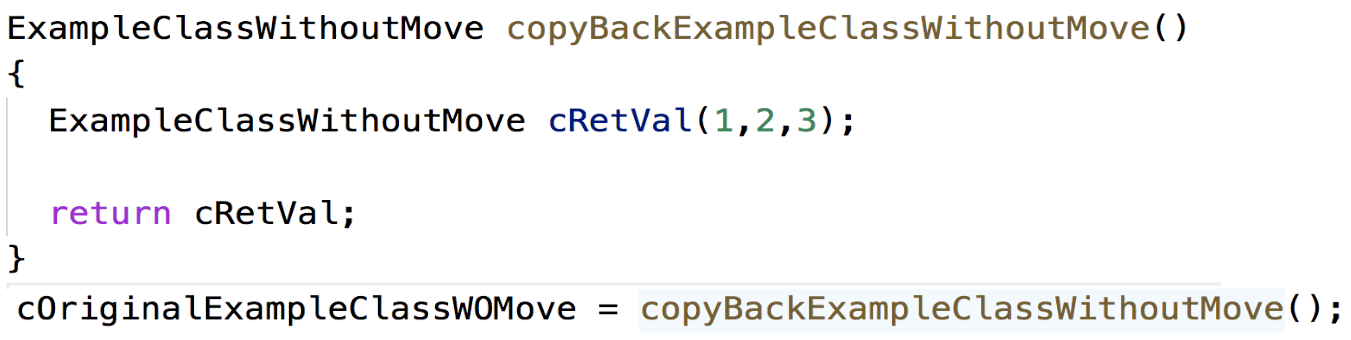
14. Why is lambda often called a closure in other languages?

15. What does it mean that a lambda is a first-class object?

16. What is going on here:

```
int main () {
  Exam question();
}
```

17. Be sure to understand how ctors, cctors, and dtors are called here:



```
ExampleClassWithoutMove copyBackExampleClassWithoutMove()
{
  ExampleClassWithoutMove cRetVal(1,2,3);

  return cRetVal;
}
cOriginalExampleClassWOMove = copyBackExampleClassWithoutMove();
```

18. Explain the role of the M, V, and P in the MVP pattern.

19. What problem is the MVP pattern trying to solve? What is being encapsulated?

20. Why do scoped enums provide stronger type guarantees than non-scoped enums.

21. What problem is the C++ namespace trying to solve?