

CS2704: Object-Oriented Software Design and Construction

Midterm Exam – Spring 2000

Section: #5454

Instructions:

- Print your name in the space provided below
- This examination is closed book and closed notes
- Answer each question on the opscan sheet provided. Answers on the question paper will not be graded.
- The maximum score is 105/100 (each question is worth 3 points)
- When you have completed the test, sign the pledge at the bottom of this page and turn in **both** the test and the opscan.
- Note that failure to return this test, or to discuss its content with a student who has not taken it, is a violation of the Honor Code.

SOLUTION

Name: _____

Pledge: On my honor, I have neither given nor received unauthorized aid on this examination.

For questions 1 through 10, select (1) if the statement is TRUE, and (2) if the statement is FALSE

1. The implementation is the visible, external aspect of the software that must be understood to use it. (FALSE)
2. Abstraction must consider all details about an entity. (FALSE)
3. The two forms of composition are distinguishable primarily by the degree to which their parts are visible. (TRUE)
4. Only one instantiations can be made from the same class. (FALSE)
5. In order to apply a method to an object, the method must be in that object's public interface. (TRUE)
6. A class may have any number of destructors. (FALSE)
7. Methods in a class may have the same name only if they have a different number of arguments. (FALSE)
8. Stream operators in C++ are overloaded to allow the same operation to be performed on different types. (TRUE)
9. Const may be applied to built-in but **not** to user-defined types. (FALSE)
10. A member function that redefines a built-in operator cannot itself be overloaded. (FALSE)

For questions 11 through 16, find the appropriate term for each definition:

- 1) Abstraction
- 2) Separation
- 3) Composition
- 4) Aggregation
- 5) Association
- 6) Class
- 7) Object
- 8) Encapsulation

11. Design technique that focuses on the essential aspects of an entity and ignores or conceals less important or nonessential aspects (1)

12. A composition of independently constructed and externally visible parts (5)

13. A composition that encapsulates the parts of the composition (4)

14. An organized collection of components interacting to achieve a coherent, common behavior (3)

15. A named software representation for an abstraction that separates the implementation of the representation from the interface of the representation (6)

16. A distinct instance of a given class that encapsulates its implementation details and is structurally identical to all other instances of that class (7)

17. An implementation satisfies an interface if (2)

- 1) it allows the different interfaces to be compiled one at a time
- 2) it provides the behavior specified by the interface
- 3) it reflects the goal to be achieved
- 4) it separates the what from the how

18. The attributes of an abstraction are represented in software by (1)

- 1) data
- 2) classes
- 3) methods
- 4) objects

19. The behaviors of an abstraction are represented in software by (3)

- 1) data
- 2) classes
- 3) methods
- 4) objects

20. What type of method is the following (from the **Frame** class)? (2)

- ```
~Frame();
```
- 1) manipulator/mutator
  - 2) destructor
  - 3) accessor/interrogator
  - 4) constructor

For questions 21 - 23, consider the partial definition of the **Testing** class is shown below.

```
class Testing {
public:
 /*method (a) */ void Show (int x, int y, char * z);
 /*method (b) */ void Show (char* x, char* y, int z);
 /*method (c) */ void Show (float x, char* y);
 /*method (d) */ void Show (bool x, int y=0, int z=1);
};
```

21. Assuming an object of the class, “**obj**”, has been properly constructed, indicate which method is invoked: (5)

```
obj.Show(5, "Hello", "there");
```

- 1) **method (a)**
- 2) **method (b)**
- 3) **method (c)**
- 4) **method (d)**
- 5) None of the these

22. Assuming an object of the class, “**obj**”, has been properly constructed, indicate which method is invoked: (1)

```
obj.Show(5.0, 10, "Hello");
```

- 1) **method (a)**
- 2) **method (b)**
- 3) **method (c)**
- 4) **method (d)**
- 5) None of the these

23. Assuming an object of the class, “**obj**”, has been properly constructed, indicate which method is invoked: (4)

```
obj.Show(true, 10);
```

- 1) **method (a)**
- 2) **method (b)**
- 3) **method (c)**
- 4) **method (d)**
- 5) None of the these

24. When declaring an array of objects, as in, for example: (1)

```
Frame f[10];
```

- 1) the class must have a constructor that takes no arguments
- 2) the class may not have any overloaded constructors
- 3) each object may be individually constructed at the time the array is declared
- 4) each object in the array must have the same set of constructor values

25. The variables defined in the private section of a class (3)

- 1) are accessible to the methods only of one instance of that class
- 2) are hidden both conceptually and visually
- 3) are accessible to the methods in all instances of that class
- 4) are visible to aggregate classes that contain the class

26. The C++ compiler must see the definition of a class A (2)

- 1) before compiling another class whose interface is similar to the interface of class A
- 2) before compiling the implementation of class A
- 3) before compiling a class whose methods are invoked by A
- 4) before compiling a class to which A refers

For questions 27 and 28, consider code segment:

```
char myString[] = "Test 1";
char *ptr = 0;

char * p1 = myString;
char * const p2 = myString;
const char * p3 = myString;
```

27. Which of the following pointers when used in the blank **does not** cause an error? (4)

```
____[2] = 'a';
```

- |       |                 |                  |
|-------|-----------------|------------------|
| 1) p1 | 4) 1 and 2 only | 7) 1, 2, and 3   |
| 2) p2 | 5) 1 and 3 only | 8) None of these |
| 3) p3 | 6) 2 and 3 only |                  |

28. Which of the following pointers when used in the blank **does not** cause an error? (5)

```
____ = ptr;
```

- |       |                 |                  |
|-------|-----------------|------------------|
| 1) p1 | 4) 1 and 2 only | 7) 1, 2, and 3   |
| 2) p2 | 5) 1 and 3 only | 8) None of these |
| 3) p3 | 6) 2 and 3 only |                  |

For questions 29 through 32, consider the class declaration:

```
class Location {
private:
 double X;
 double Y;
public:
 Location(double initX = 2.0, double initY = 3.0);
 void setX(double x);
 void setY(double y);
 double getX() const;
 double getY() const;
};
```

29. Which member function is an observer (or reporter) operation? (6)

- |                |                 |                  |
|----------------|-----------------|------------------|
| 1) <b>getX</b> | 4) <b>getY</b>  | 7) 2 and 3 only  |
| 2) <b>setX</b> | 5) 1 and 2 only | 8) None of these |
| 3) <b>setY</b> | 6) 1 and 4 only |                  |

30. Which member function is a mutator operation? (7)

- |                |                 |                  |
|----------------|-----------------|------------------|
| 1) <b>getX</b> | 4) <b>getY</b>  | 7) 2 and 3 only  |
| 2) <b>setX</b> | 5) 1 and 2 only | 8) None of these |
| 3) <b>setY</b> | 6) 1 and 4 only |                  |

31. Suppose we want to add an equality member function to `Location`: (1)

```
bool Location::Equals(const Location& Other) {
 return (_____);
}
```

Which of the following expressions could be correctly used in the blank?

- 1) `( X == Other.getX() ) && ( Y == Other.getY() )`
- 2) `( X == Other.X() ) && ( Y == Other.Y() )`
- 3) All of the above
- 4) None of these

32. How would the member function `Equals()` given above be invoked, assuming the declarations: (2)

```
Location A(0.4, 0.5), B;
```

- |                         |                 |                  |
|-------------------------|-----------------|------------------|
| 1) <b>Equals(A, B);</b> | 4) 1 and 2 only | 7) 1, 2 and 3    |
| 2) <b>A.Equals(B);</b>  | 5) 1 and 3 only | 8) None of these |
| 3) <b>A.Equals.B;</b>   | 6) 2 and 3 only |                  |

For questions 33 through 35, consider the class declaration:

```
class PrimeNumber {
 ...
public:
 PrimeNumber();
 PrimeNumber(string);
 operator int();
 static void add(int);
};
```

```
PrimeNumber prime;
```

33. Which of the following statements will cause a compiler error? (3)

- 1) `int a = (int)prime;`
- 2) `float a = (float)prime;`
- 3) `string a = (string)prime;`
- 4) 2 and 3
- 5) None of these

34. Which of the following statements will cause a compiler error? (4)

- 1) `PrimeNumber *nprime = new PrimeNumber("H");`
- 2) `PrimeNumber *nprime = new PrimeNumber('H');`
- 3) `PrimeNumber *nprime = new PrimeNumber(5);`
- 4) 2 and 3
- 5) None of these

35. Which of the following statements will cause a compiler error? (5)

- 1) `prime.add(5);`
- 2) `PrimeNumber::add(5);`
- 3) `prime.add(PrimeNumber());`
- 4) 2 and 3
- 5) None of these

*Have a nice spring break!*