# **Object Oriented Programming Midterm Exam**

## (2018-05-09 WED)

Name : Student No. :

1. **What are the three principal characteristics in Object Oriented Programming? D**
2. Encapsulation, has a / is a Relationship, abstract class
3. Polymorphism, overloading, overriding
4. Encapsulation, inheritance, abstract class
5. Encapsulation, inheritance, polymorphism
6. **Which one is the proper main method in JAVA language? B**
7. public final void main(String args[])
8. public static void main(String[] args)
9. public static int main(String args[])
10. private static void main(String[] args)
11. private static void main(String args[])
12. **Which of the following cannot be used as an identifier? [choose 2] A, B**
13. final
14. 4you
15. $name
16. \_company
17. per4son
18. **Which of the following is used to find the length of an array? A**
19. array.length;
20. array.length();
21. array.size();
22. array.size;
23. **Which of the following is true? A, C**
24. The constructor of the parent class is not inherited by the subclass.
25. The constructor cannot be overloaded.
26. The constructor cannot be overridden.
27. The constructor of the parent class cannot be called with super operator.
28. Methods cannot contain this operator and super operator calls.
29. **Which statement is not true for the modifier final? D**
30. final variables cannot be changed
31. final methods cannot be overridden.
32. final classes cannot be sub-classed
33. final instance variables are initialized by default
34. **Which statement is not true for the modifier static? E**
35. static methods can access only static variables
36. static methods may not be overridden to be non-static.
37. Non-static methods may not be overridden to be static.
38. Abstract methods may not be static.
39. Local variables be declared as static
40. **Which two overload the setVar method? [choose 2] D, E**

Given:

|  |
| --- |
| public class MethodOver {  private int x, y;  private float z;  public void setVar(int a, int b, float c) {  x = a; y = b; z = c;  }  } |

1. void setVar(int a, int b, float c) { x = a ; y = b ; z = c ; }
2. public void setVar(int ax, int by, float cz) { x = ax ; y = by ; z = cz ; }
3. public void setVar(int a, float c, int b) { this(a,b,c); }
4. public void setVar(int a, float c, int b) { setVar(a, b, c); }
5. public void setVar(int a, float b) { x = a ; y = b ; }
6. **Which statement is true? [choose 3] A, B, C**

Given:

|  |
| --- |
| Integer s = new Integer(9);  Integer t = new Integer(9);  Integer l = t;  Long u = new Long(9); |

1. (s.equals(t))
2. (s.equals(9))
3. (t == l)
4. (s == t)
5. (s == u)
6. **Which four statements are true? [choose 4.] A, B, C, F**

Given:

|  |
| --- |
| class Mammal { }  class Raccoon extends Mammal {  Mammal m;  Raccoon () { m = new Mammal(); }  }  class BabyRaccoon extends Mammal { } |

1. Raccoon is-a Mammal.
2. Raccoon has-a Mammal.
3. BabyRaccoon is-a Mammal.
4. BabyRaccoon is-a Raccoon.
5. BabyRaccoon has-a Mammal.
6. BabyRaccoon is-a BabyRaccoon
7. **Which three statements are true? [choose 3] A, D, E**
8. A protected method in class X can be overridden by any subclass of X.
9. A private static method can be called only within other static methods in class X.
10. A non-static public final method in class X can be overridden in any subclass of X.
11. A public static method in class X can be called by a subclass of X without explicitly referencing the class X.
12. A method with the same signature as a private final method in class X can be implemented in a subclass of X.
13. A protected method in class X can be overridden by a subclass of X only if the subclass is in the same package as X.
14. **Which of the following is true? E**

Given:

|  |
| --- |
| public class Test {  /\* default definition of main() function is missing \*/  main() {  int [] v = new int[1];  v[0] = 5;  Test t = new Test();  t.inc(v);  System.out.print(v[0]);  }  void inc(int[] v) {  System.out.print(v[0]++);  }  } |

1. Runtime error
2. Runtime exception
3. Print ‘65’
4. Print ‘55’
5. Print ‘56’
6. **What Code can be placed on line //A to print the season’s entire season on the screen? A**

Given:

|  |
| --- |
| public class Test {  public enum Season { SPIRNG, SUMMER, FALL, WINTER }  public void printSeason(){  //A  System.out.println(s);  }  } |

1. for (Season s : Season.values())
2. for (Season s = Season.values())
3. for (Season s ; s.lengh ; s++)
4. for (Season s ; s.hasNext() ; s.next() )
5. for (Season s = s.next() )
6. **What is the result? D**

Given:

|  |
| --- |
| abstract class C1 { public C1() { System.out.print(1); } }  class C2 extends C1 { public C2() { System.out.print(2); } }  class C3 extends C2 { public C3() { System.out.println(3); } }  public class Ctest {  /\* default definition of main() function is missing \*/  main() { new C3(); }} |

1. 3
2. 23
3. 32
4. 123
5. 321
6. Compilation fails.
7. An exception is thrown at runtime.
8. **Which of the following code uses no error code? C**

Given:

|  |
| --- |
| abstract class Person {  String name;  public abstract String getName();  public void setName( String name) {  this.name = name;  }  }  class Student extends Person {  public String getName(){  return name;  }  } |

1. Person p = new Person();
2. Student p = new Person();
3. Student p = new Student();
4. Student p = {Person} new Student();
5. Student p = (Student) new Person();
6. **Which statement is true? D**

Given:

|  |
| --- |
| class A {  int number = 5;  int getNumber() {  return number;  }  }  class B extends A {  int number = 10;  int getNumber() {  return number;  }  }  public class Test {  /\* default definition of main() function is missing \*/  main(){  A a = new B();  System.out.println(“Number : ” + a.number);  System.out.println(“Number : ” + a.getNumber());  }  } |

1. The code will not compile.
2. The code will compile but runtime error.
3. The code will compile and run, print out “Number : 5” followed by “Number : 5”.
4. The code will compile and run, print out “Number : 5” followed by “Number : 10”.
5. The code will compile and run, print out “Number : 10” followed by “Number : 5”.
6. The code will compile and run, print out “Number : 10” followed by “Number : 10”.
7. **What is the result? F**

Given:

|  |
| --- |
| class Atom {  Atom(String param1) {  System.out.print(param1);  }  }  class Rock extends Atom {  Rock(String param1) {  this("2 ", "3 ");  System.out.print(param1);  }  Rock(String param1, String param2) {  super("2 ");  System.out.print(param2);  }  }  public class Mountain extends Rock {  Mountain() {  super("1 ");  }  /\* default definition of main() function is missing \*/  main() {  new Mountain();  }  } |

1. Compilation fails.
2. 2 2 3 1
3. 2 3 1 2 3 1
4. 2 2 3 1 2 2 3 1
5. An exception is thrown at runtime.
6. 2 3 1
7. **Which four code fragments will be compiled when they are inserted into the part that is commented as "//A"? [choose 4] A, B, E, F**

Given:

|  |
| --- |
| public class Hi {  void m1() { }  protected void m2() { }  }  class Lois extends Hi {  //A  } |

1. public void m1() { }
2. protected void m1() { }
3. private void m1() { }
4. void m2() { }
5. public void m2() { }
6. protected void m2() { }
7. private void m2() { }
8. **Given that: Gadget has-a Sprocket and Gadget has-a Spring and Gadget is-a Widget and Widget has-a Sprocket Which two code fragments represent these relationships? [choose 2] A, C**
9. class Widget { Sprocket s; }  
   class Gadget extends Widget { Spring s; }
10. class Widget { }  
    class Gadget extends Widget { Spring s1; Sprocket s2; }
11. class Widget { Sprocket s1; Spring s2; }  
    class Gadget extends Widget { }
12. class Gadget { Spring s; }  
    class Widget extends Gadget{ Sprocket s; }
13. class Gadget { }  
    class Widget extends Gadget{ Sprocket s1; Spring s2; }
14. class Gadget { Spring s1; Sprocket s2; }  
    class Widget extends Gadget{ }.
15. **What is the result? D**

Given:

|  |
| --- |
| public class Base {  public static final String FOO = "foo";  /\* default definition of main() function is missing \*/  main() {  Base b = new Base();  Sub s = new Sub();  System.out.print( Base.FOO );  System.out.print( Sub.FOO );  System.out.print( b.FOO );  System.out.print( s.FOO );  System.out.print( ((Base)s).FOO );  }  }  class Sub extends Base {  public static final String FOO="bar";  } |

1. foofoofoofoofoo
2. foobarfoobarbar
3. foobarfoofoofoo
4. foobarfoobarfoo
5. barbarbarbarbar
6. foofoofoobarbar
7. foofoofoobarfoo