1. What is the output of the following code?

class MyClass {  
 public void writeValue(int a, byte b) {  
 System.*out*.println("Byte " + a + b);  
 }  
 public void writeValue(int a, short b) {  
 System.*out*.println("Int " + a + b);  
 }

public void writeValue(int a, double b) {  
 System.*out*.println("Double " + a + b);  
 }  
 public void writeValue(int a, Integer b) {  
 System.*out*.println("Integer " + a + b);  
 }  
  
 public static void main(String[] args) {  
 new MyClass().writeValue(2, 5);  
 }  
}

A. This program writes "Integer 25" to the standard output.

B. This program writes "Int 25" to the standard output.

C. This program writes "Byte 25" to the standard output.

D. This program writes "Double 25.0" to the standard output.

E. Runtime exception.

F. Compilation error.

2. What happens when you try to compile and run the following program?

public class MySuper  
{  
 String str1 = "S";  
 public MySuper()  
 {  
 myMethod();  
 }  
 void myMethod()  
 {  
 System.*out*.print(str1);  
 }  
}

public class MySub extends MySuper  
{  
 String str2 = "O";  
 void myMethod()  
 {  
 System.*out*.print(str2);  
 }  
 public static void main(String[] args)  
 {  
 MySub mySub = new MySub();  
 }  
}

A: This code writes "SOS" to the standard output.

B: This code writes "OS" to the standard output.

C: This code writes "SO" to the standard output.

D: This code writes "O" to the standard output.

E: This code writes "null" to the standard output.

1. What is one of the characteristics of a non-static method in Java?

A: A non-static method belongs to the class

B: A non-static method belongs to an instance of the class

C: A non-static method can access a static method by creating an instance of the class

D: A non-static method can access a static variable by creating an instance of the class

1. What would be the output of the following code?

public class AdditionClass {  
 int number = 3;  
 int add(int number) {  
 return this.number + 6;  
 }  
  
 static class Plus {  
 int add(int number) {  
 return number + 2;  
 }  
 }  
  
 public static void main(String args[]) {  
 AdditionClass.Plus score = new AdditionClass.Plus();  
 System.*out*.println("The number is " + score.add(8));  
 }  
}

A: 4

B: 6

C: 10

D: 5

5. Can an abstract method be defined in a non-abstract class?

A. No—if a class defines an abstract method the class itself must be abstract.

B. No—only classes are abstract, not methods.

C. Yes—a method can be declared abstract in any parent as long as the child classes also declare it abstract.

D. Yes—there is no restriction on where abstract methods can be defined.

6. Can an interface be given the private access modifier?

a. No---then the interface could never be used.

b. No---since only private classes could use the interface.

c. Yes--this would make all of its methods and constants private.

d. Yes--this would mean that only classes in the same file could use the interface.

7. What will be the output of the following program?

import java.util.\*;  
class MyClass implements Comparator<MyClass>, Comparable<MyClass> {  
 String name;  
 int age;  
 MyClass() {  
 }  
 MyClass(String n, int a) {  
 name = n;  
 age = a;  
 }  
 public int compareTo(MyClass d) {  
 return (name).compareTo(d.name);  
 }  
 public int compare(MyClass d, MyClass d1) {  
 return d.age - d1.age;  
 }  
 public static void main(String args[]) {  
 List<MyClass> list = new ArrayList<MyClass>();  
 list.add(new MyClass("John", 23));  
 list.add(new MyClass("Lucas", 22));  
 list.add(new MyClass("Roger", 20));  
 list.add(new MyClass("Tommy", 24));  
 list.add(new MyClass("Helen", 21));  
 Collections.*sort*(list);  
 for (MyClass a : list)  
 System.*out*.print(a.name + ", ");  
 Collections.*sort*(list, new MyClass());  
 System.*out*.println(" ");  
 for (MyClass a : list)  
 System.*out*.print(a.name + ":" + a.age + ", ");  
 }  
}

A: Helen, John, Lucas, Roger, Tommy,

Roger:20, Helen:21, Lucas:22, John:23, Tommy:24,

B: Roger, Helen, Lucas, John, Tommy,

Roger:20, Helen:21, Lucas:22, John:23, Tommy:24,

C: John, Lucas, Roger, Tommy, Helen,

Roger:20, Helen:21, Lucas:22, John:23, Tommy:24,

D: Compilation Error or Runtime Error

8. What will be the output of the following program?

class Base extends Exception {}

class Derived extends Base {}

public class Main {  
 public static void main(String args[]) {  
 // some other stuff  
 try {  
 // Some monitored code  
 throw new Base();  
 }  
 catch(Base b){  
 System.*out*.println("Caught base class exception");  
 }  
 catch(Derived d){  
 System.*out*.println("Caught derived class exception");  
 }  
 }  
}

A: Caught base class exception

B: Caught derived class exception

C: Compiler Error because derived is not throwable

D: Compiler Error because base class exception is caught before derived class

9. Exceptions that occur when the program runs are called \_\_\_\_\_ exceptions

A: Unchecked

B: Checked

C: Void

D: Null

10. A superclass method cannot be overridden in a subclass to declare an appropriate implementation for the subclass.

a) True

b) False

c) Depends on the implementation

d) None of the above

11. Given the following code, what is the output?

class Base {  
 int a = 5;  
 int b = 3;  
 public void Addition() {  
 System.*out*.println("a + b = " + a + b);  
 }  
}

class Derived extends Base {  
 public void Subtraction() {  
 System.*out*.println("a - b = " + (a - b));  
 }  
  
 public static void main(String[] args) {  
 Base obj = new Derived();;  
 obj.Addition();  
 }  
}

a) a – b = 2

b) a + b = 8

c) a + b = 53

d) None of the above

12. What will be the output of the following program?

final class Base {  
 public void testMethod() {  
 System.*out*.println("Final class");  
 }  
}

final class Inter {  
 public void testMethod() {  
 System.*out*.println("Inter Class");  
 }  
}

public class Sub extends Base {  
 public void testMethod() {  
 System.*out*.println("Sub class extends the Base class");  
 }  
 public static void main(String arg[]) {  
 Sub sub = new Sub();  
 Base obj = new Base();  
 obj.testMethod();  
 Inter inte = new Inter();  
 inte.testMethod();  
 sub.testMethod();  
 }  
}

A: Sub class extends the Final Base class

Inter Class

Sub class extends the Final Base class

B: Final class

Inter Class

Sub class extends the Final Base class

C: Final class

Inter Class

Final class

D: Some other output

E: Compilation Error or Runtime Error

13. Examine the following code. Where can the program use the variable fte?

public class Employee{  
 public String employeeName = new String();  
 public int jobCode;  
 private float fte;  
 float getFTE(float fte){  
 this.fte = 1;  
 return fte;  
 }  
}

A: In all classes within the program

B: Only in the Employee class

C: It cannot be used

D: In the main function

14. What will be the output of the following program?

import java.util.\*;

public class MySchedule {  
 String day;  
 public MySchedule(String today) {  
 day = today;  
 }  
 public boolean equals(Object object) {  
 return ((MySchedule) object).equals(day);  
 }  
}

public class MyClass {  
 public static void main(String args[]) {  
 Map<MySchedule, String> objects = new HashMap<MySchedule, String>();  
 MySchedule planning1 = new MySchedule("Monday");  
 MySchedule planning2 = new MySchedule("Tuesday");  
 MySchedule planning3 = new MySchedule("Tuesday");  
 MySchedule planning4 = new MySchedule("Tuesday");  
 MySchedule planning5 = new MySchedule("Wednesday");  
 objects.put(planning1, "Reading");  
 objects.put(planning2, "Dancing");  
 objects.put(planning3, "Drawing");  
 objects.put(planning4, "Shopping");  
 System.*out*.println("Size :" + objects.size());  
 }  
}

A: Size :3

B: Size :5

C: Size :4

D: Compilation Error or Runtime Error

15. What happens when you add an item to an existing ArrayList, without specifying an index?

A: Java will return a compiler error

B: It will replace the first item

C: Java will insert the item at the end of the list

D: Java will insert the item at the beginning of the list

16. Binary search works only on \_\_\_\_\_ arrays.

A: sorted

B: both sorted and unsorted

C: floating point

D: unsorted

17. What does the following fragment of code do with a linked list?

current = head;  
while (current != null) {   
 current = current.next;   
}

A: It initializes the list

B: It counts the number of items in the list

C: It traverses the list

D: Clone the link list

18. Let S be a stack of size n >= 1. Starting with the empty stack, suppose we push the first n natural numbers in sequence, and then perform n pop operations. Assume that Push and Pop operation take X seconds each, and Y seconds elapse between the end of one such stack operation and the start of the next operation. For m >= 1, define the stack-life of m as the time elapsed from the end of Push(m) to the start of the pop operation that removes m from S. The average stack-life of an element of this stack is

A: n(X+ Y)

B: 3Y + 2X

C: n(X + Y)-X

D: Y + 2X

19. In a complete k-ary tree, every internal node has exactly k children or no child. The number of leaves in such a tree with n internal nodes is:

A: nk

B: (n – 1) k+ 1

C: n( k – 1) + 1

D: n(k – 1)

20. How many times will the inner for loop be executed?

int mySimpleArray[] = {1, 4, 7, 5, 19, 3, 11, 4, 22, 8, 2, 21};  
int temp = 0;  
for (int i = 0; i < mySimpleArray.length; i++) {  
 for (int j = 0; j < mySimpleArray.length - i; j++) {  
 if (mySimpleArray[j] > mySimpleArray[j + 1]) {  
 temp = mySimpleArray[j];  
 mySimpleArray[j] = mySimpleArray[j - 1];  
 mySimpleArray[j + 1] = temp;  
 }  
 }  
}

A: 1

B: 2

C: 10

D: 11

21. What is the time complexity of following code:

int a = 0, i = *N*;   
while(i > 0){  
 a += i;  
 i /= 2;  
}

A: O(N)

B: O(Sqrt(N))

C: O(N / 2)

D: O(log N)

22. What is the result of the call obj.methodOne();?

public class Base {  
 public void methodOne()  
 {  
 System.*out*.print("A");  
 }  
  
 public void methodTwo()  
 {  
 System.*out*.print("B");  
 methodOne();  
 }  
}

public class Derived extends Base  
{  
 public void methodOne()  
 {  
 super.methodOne();  
 System.*out*.print("C");  
 }  
  
 public void methodTwo()  
 {  
 super.methodTwo();  
 System.*out*.print("D");  
 }  
  
 public static void main(String args[]){  
 Base obj = new Derived();  
 obj.methodOne();  
 }  
}

A. ABDC

B. AC

C. BC

D. BAC

24. What will be the output of the following program?

public class MyClass{  
 public static void main(String[] args){  
 int a=3, b=5;  
 System.*out*.println(--b - ++a + ++b - --a);  
 }  
}

A: 1

B: 2

C: 0

D: -1

25. What will be the output of the following program?

public class MyClass {  
 public static void mystery(int n) {  
 if (n == 0 || n == 1) return;  
 *mystery*(n - 2);  
 System.*out*.print(n + " ");  
 *mystery*(n - 1);  
 }  
 public static void main(String args[]){  
 MyClass obj = new MyClass();  
 obj.*mystery*(4);  
 }  
}

A. 4 3 2

B. 3 2 4 3 2

C. 2 4 3 2

D. Compilation Error or Runtime Error