1. Which primitive type can not be casted to any other primitive type?

    - int
    - !boolean
    - char
    - long

2. Values of which type can be used in the switch construction (JDK 1.7+):

    - !int
    - Long
    - !All enum types
    - !String
    - !Short
    - double
    - All types, for which the equals method is defined

3. Which of the two pieces of code prevents the NullPointerException occurrence:

    ```
    if (a != null && a.size > 0) { //fragment 1
    if (a != null & a.size > 0) {  //fragment 2
    ```

    - !First
    - Second
    - Both
    - None of them

4. Values of which type can be used in the “advanced for” construction after the “:” symbol?

    - Any types
    - !Collections
    - !Arrays
    - !Any types implementing Iterable

5. Which modifiers can a top-level class have?

    - !public
    - private
    - !abstract
    - !final
    - protected
    - static
    - virtual

6. Which type should be used instead of String to compile the following code?

    ```
    class A {
    ```

```
  public void m(String... args) {  }
}
class Tester {
  public void m() {
     A a = new A();
     a.m(1, 3L, 4.4f);
  }
}
```

- Double
- Float
- !Number
- Integer
- !Object
- Long

7. What will the following program print:

```
public class B extends A {
  private int j = 3;
 @Override
  public void setI() { i = j; }
  public static void main(String[] args) {
     B b = new B();
     System.out.println(b.getI());
  }
}
class A {
  protected int i = 1;
  public A() { setI(); }
  public void setI() { i = 2; }
  public int getI() { return i; }
}
```

- !0
- 1
- 2
- 3
- It won't compile

8. Which modifier can be set for the A class?

```
public class B {
  public static void main(String[] args) {
     class A {}
  }
}
```

- !final
- !abstract
- static
- public

9. Which modifier can be set for the A class?

```
public class B {
  class A {}
}
```

- !final
- !abstract
- !static
- !public

10. What will happen during the following program compilation:

```
1       class VariableInit {
2               static String s;
3               float f;
4               int i;
5
6               public static void main(String[] args) {
7                       int a;
8                       float f = 0;
9                       System.out.println(a);
10                      System.out.println(f);
11                      System.out.println(s);
        }
}
```

- compilation error on line 2
- compilation error on line 7
- compilation error on line 8
- !compilation error on line 9
- compilation error on line 10
- compilation error on line 11
- compiling without errors

11. This method reverses the values of two variables (without using the third one), and outputs the result. What are valid values for the a and b parameters?

```
private void swap(int a, int b) {
   System.out.println("a = " + a + " b = " + b);
   a = a + b;
   b = a - b;
   a = a - b;
```

```
    System.out.println("a = " + a + " b = " + b);
}
```

- The sum of a and b shall not exceed 2147483647 (2^{31}-1)
- !Any values are valid
- The sum of a and b shall not exceed 4294967296 (2^{32})
- a and b must be positive
- a and b shall not equal 0

12. Which value will the i variable contain after the object creation?

```
public class A {
  private int i;
  static {
     i = 1;
  }
  public A() {
     i = 2;
  }
}
```

- 1
- 2
- !Code will not compile

13. How will this code be executed after the exception occurring on line 1:

```
try {
  throw new IOException();  // 1
 } catch (IOException ex) {
    throw new IOException();// 2
 } catch (Exception ex) {
                    // 3
 } finally {
                    // 4
 }
                    // 5
```

- 2, 3, 4, 5
- 3, 4, 5
- 2 and throwing IOException from line 2
- !2, 4 and throwing IOException from line 2

14. Which exceptions will occur if we run this code without command line parameters?

```
public static void main(String[] args) {
 String s = (String) args[0];
 if (s.length()>0) {
```

```
        int x = Integer.parseInt(s);
        x++;
        System.out.println(x);
}
}
```

- ClassCastException
- IllegalStateException
- NumberFormatException
- IllegalArgumentException
- !ArrayIndexOutOfBoundsException
- ArithmeticException
- None

15. There is a code:

```
public static String test() {
String s = null;
try {
        s.length();
        s +=1;
        return s;
} catch(Exception e) {
        s +=2;
        return s;
} finally {
        s +=3;
        return s;
}
}

public static void main(String[] args) {
System.out.print(test());
}
```

What will be displayed during its execution?

- null1
- null2
- null3
- !null23
- null123
- null2, then NullPointerException stack trace
- NullPointerException stack trace
- Code will not compile

16. Which of the &= and &&= operators is valid?

- !&=
- &&=
- Both
- None

17. Which values will be assigned to the i, j, k variables?

```
int k = 1;
int i = 2;
int j = ++ i + k ++;
```

- k = 1, i = 2, j = 3
- k = 2, i = 3, j = 5
- !k = 2, i = 3, j = 4
- k = 2, i = 2, j = 4

18. Which of the following statements is correct?

- The enum enumeration should declare that it extends java.lang.Enum
- It is allowed to create enumeration subclasses
- !Enumerations can contain public methods
- !Enumerations can contain data with private modifier

19. The x and y variables are of an enumerated type (enum). What is the best way to test them for equality?

- !if (x == y)
- if (x.equals(y))
- if (x.toString().equals(y.toString()))
- if (x.hashCode() == y.hashCode())

20. Let's consider the following enumeration:

```
enum Spice { NUTMEG, CINNAMON, CORIANDER, ROSEMARY; }
```

Which options will compile without errors:

- !Spice sp = Spice.NUTMEG; Object ob = sp;
- !Spice sp = Spice.NUTMEG; Object ob = (Object)sp;
- Object ob = new Object(); Spice sp = object;
- !Object ob = new Object(); Spice sp = (Spice)object;

21. Which method can be used to get a list of all the values from the States enumeration?

- new ArrayList(States)
- States.names()
- !States.values()

- States.list()
- States.getValues()

22. Will this code compile? Will it be possible to use the INT variable from the B class?

```
class Tester {
  public static void main(String[] args) {
     B b = new B();
  }
}
class B implements I1, I2 {}
interface I1 {
  public static final int INT = 1;
}
interface I2 {
  public static final int INT = 10;
}
```

- !The code compiles, but you can not use B.INT
- The code compiles, and you can use B.INT. The value will depend on the order of the interfaces enumerated in the class description
- Code will not compile

23. How to change the size of the array inside the m method of the B class?

```
class B {
  public void m(int[] arg) {
  }
}
class Tester {
  public static void main(String[] args) {
     B b = new B();
     int a[] = new int[10];
     b.m(a);
  }
}
```

- arg = new int[100]
- arg.resize(100)
- arg.length = 100
- !It is impossible to change the size of the arg array

24. Will the following code compile?

```
class X extends Exception{}
class Y extends X{}
class C{ void doSomething() throws Y{} }
```

class D extends C{ void doSomething() throws Y{} }

- !yes
- no

25. What will be the output of the program?

```
class A {}


class B extends A {}


public class Main {
  public static void main(String[] args) {
     A a = new B();
     B b = (B) a;
     System.out.print(a == b);
  }
}
```

- false
- !true
- Compilation error
- Run-time error
- None of the above

26. Which operations with reference variables in the body of the main () method do not lead to a compilation error?

```
class A{}
class B extends A {}
class C extends B {}


public class MyClass {
  public static  void main(String args[]) {
     A x1 = new A();
     B x2 = new B();
     C x3 = new C();
     //<- code here
  }
}
```

- !x2 = x3;

- !x1 = x3;
- !x1 = x2;
- x3 = x1;
- all operations above are wrong

27. Which statements about the following code are correct:

```
class A{
        A(int i){ //1
        }
}
class B extends A{ //2
}
```

- the compiler is trying to create a default constructor for the A class
- !the compiler is trying to create a default constructor for the B class
- compilation error on line 1
- !compilation error on line 2

28. What will be the result of this program execution?

```
class Top{}
class Sub extends Top{}
public class Test{
        public static void main(String[] args){
                Top t = new Top();
                Sub s = (Sub) t; } }
```

- !Run-time exception
- Compiling and running without errors
- Compilation error

29. What will be the output of the program?

```
class Parent{
        static int x = 1;
        public Parent(){ x += 2; }
        }
        class Child extends Parent{
                public Child(){ x += 1; }
        public static void main(String[] args) {
                Child c = new Child();
                System.out.println(x);
        }
}
```

- 1
- 2
- 3
- !4

30. If we compile and call the m method, what will happen with exception on line 1:

```
public void m() throws IOException {
 try {
   throw new IOException();    // 1
 } catch (IOException ex) {
   throw new IOException();    // 2
 } catch (Exception ex) {

 } finally {

 }
}
```

- It will be placed inside the exception on line 2
- It will be saved to a global list of exceptions
- !It will be lost

31. Let's consider the following code:

```
class CheckNameException extends Exception {};

public void checkName(String name) {
 if (name.equals("John")) {
            System.out.println("Correct name!");
 } else {
            throw new CheckNameException();
 }
}
```

What changes should we make to compile it without errors?

- !To add throws Exception to the header of the checkName() method
- !To add throws CheckNameException to the header of the checkName() method
- !Place the body of the checkName() method to the try {…} catch(Exception e) {} block
- !Place the body of the checkName() method to the try {…} catch(CheckNameException e) {} block
- !Inherit CheckNameException from the RuntimeException class
- Inherit CheckNameException from the Throwable class

32. Which of the following is not true about JVM:

- !It is compiler

- It interprets the bytecode
- It is specification
- It makes Java cross-platform language

33. To execute a java class file named Test.class we write at command line:

- javac Test.class
- java Test.class
- !java Test
- Test

34. A final keyword causes:

- !a variable to be made a constant
- call garbage collector
- !makes it non-inheritable

35. x=x+10, can be written as:

- !x+=10;
- x=+10;
- x++10
- cannot be done

36. What will be the output of the program?

```
class Test{
 public static void main(String [] args){
            byte b = 127;
            b++;
            System.out.println(b);
            }
}
```

- 127
- 128
- !-128
- 0
- Compile time error
- Run-time exception

37. What will be the output of the program?

```
class Test{
 public static void main(String [] args){
            byte b = 100;
            byte b2 = 11;
```

```
        byte b3 = b + b2;
        System.out.print(b3);
        }
}
```

- 111
- !Compile time error
- Run-time exception

38. Java always pass class objects to a method argument as:

- pointer
- alias
- value
- !reference value

39. Let's consider the Mouse class:

```
public class Mouse {
    public String name;

    public void run() {
        try {
            name.toString();
            System.out.print("1");
        } catch (NullPointerException e) {
            System.out.print("2");
            throw e;
        }
        System.out.print("3");
    }

    public static void main(String[] args) {
        new Mouse().run();
        System.out.print("4");
    }
}
```

What will be the output of the program?

- 1
- !2
- 3
- 4
- !NullPointerException stack trace

40. What will be the output of the program?

```
public class Test{
  static int x;
  boolean catch() {
     x++;
     return true;
  }
  public static void main(String[] args) {
     x=0;
     if ((catch() | catch()) || catch())
        x++;
     System.out.println(x);
  }
}
```

- 1
- 2
- 3
- !Compilation fails

41. What will be the output of the program?

```
int i = 1, j = 10;
do {
  if(i++ > --j) {
     continue;
  }
} while (i < 5);
System.out.println("i = " + i + "and j = " + j);
```

- i = 6 and j = 5
- i = 5 and j = 5
- i = 6 and j = 6
- !i = 5 and j = 6

42. What will be the output of the program?

```
class Equals {
  public static void main(String [] args)    {
     int x = 100;
     double y = 100.1;
     boolean b = (x = y);
     System.out.println(b);
  }
}
```

- true
- false

- !Compilation fails
- An exception is thrown at runtime

43. What will be the output of the program?

```
class Test {
  public static void main(String[] args) {
    int x = 0;
    int y = 0;
    for (int z = 0; z < 5; z++) {
      if ((++x > 2) && (++y > 2)) {
        x++;
      }
    }
    System.out.println(x + " " + y);
  }
}
```

- 5 2
- 5 3
- !6 3
- 6 4

44. Which four can be thrown using the throw statement?

1. Error
2. Event
3. Object
4. Throwable
5. Exception
6. RuntimeException

- 1, 2, 3 and 4
- 2, 3, 4 and 5
- !1, 4, 5 and 6
- 2, 4, 5 and 6

45. What will be the output of the program?

```
try {
    int x = 0;
    int y = 5 / x;
  } catch (Exception e) {
    System.out.println("Exception");
  } catch (ArithmeticException ae) {
    System.out.println(" Arithmetic Exception");
  }
  System.out.println("finished");
```

- finished
- Exception
- !Compilation fails.
- Arithmetic Exception

46. Which one of these lists contains only Java programming language keywords (JAVA SE7)?

- !goto, instanceof, native, finally, default, throws
- try, virtual, throw, final, volatile, transient
- strictfp, constant, super, implements, do
- byte, break, assert, switch, include

47. What will be the output of the program?

```
class A {
  final public int GetResult(int a, int b) {
     return 0;
  }
}
class B extends A {
  public int GetResult(int a, int b) {
     return 1;
  }
}
public class Test {
  public static void main(String args[]) {
     B b = new B();
     System.out.println("x = " + b.GetResult(0, 1));
  }
}
```

- x = 0
- x = 1
- !Compilation fails.
- An exception is thrown at runtime

48. What will be the output of the program?

```
class SC2 {
  public static void main(String[] args) {
     SC2 s = new SC2();
     s.start();
  }

  void start() {
     int a = 3;
     int b = 4;
```

```
    System.out.print(" " + 7 + 2 + " ");
    System.out.print(a + b);
    System.out.print(" " + a + b + " ");
    System.out.print(foo() + a + b + " ");
    System.out.println(a + b + foo());
  }

  String foo() {
    return "foo";
  }
}
```

- 9 7 7 foo 7 7foo
- 72 34 34 foo34 34foo
- 9 7 7 foo34 34foo
- !72 7 34 foo34 7foo

49. What will be the output of the program?

```
public void foo(boolean a, boolean b) {
    if (a) {
      System.out.println("A");
    } else if (a && b) {
      System.out.println("A && B");
    } else {
      if (!b) {
        System.out.println("notB");
      } else {
        System.out.println("ELSE");
      }
    }
  }
```

- If a is true and b is true then the output is "A && B"
- If a is true and b is false then the output is "notB"
- !If a is false and b is true then the output is "ELSE"
- If a is false and b is false then the output is "ELSE"

50. What will be the output of the program?

```
public class Test {
  public static void stringReplace(String text) {
    text = text.replace('j', 'c');
  }

  public static void bufferReplace(StringBuffer text) {
    text = text.append("c");
```

```
  }

  public static void main(String args[]) {
    String textString = new String("java");
    StringBuffer textBuffer = new StringBuffer("java");
    stringReplace(textString);
    bufferReplace(textBuffer);
    System.out.println(textString + textBuffer);
  }
}
```

- java
- javac
- !javajavac
- Compile error

51. What will be the output of the program?

```
class ArrayTest {
  public static void main(String[] args) {
    float f1[], f2[];
    f1 = new float[10];
    f2 = f1;
    System.out.println("f2[0] = " + f2[0]);
  }
}
```

- !It prints f2[0] = 0.0
- It prints f2[0] = NaN
- An error at f2 = f1; causes compile to fail.
- It prints the garbage value

52. What will be the output of the program?

```
class Test {
  static boolean b;

  public static void main(String[] args) {
    short hand = 42;
    if (hand < 50 && !b) /* Line 7 */
      hand++;
    if (hand > 50) ;     /* Line 9 */
    else if (hand > 40) {
      hand += 7;
      hand++;
    } else
      --hand;
    System.out.println(hand);
```

```
  }
}
```

- 41
- 42
- 50
- !51

53. What will be the output of the program?

```
int i = 0;
    while (1) {
      if (i == 4) {
         break;
      }
      ++i;
    }
    System.out.println("i = " + i);
```

- i = 0
- i = 3
- i = 4
- !Compilation fails

54. What will be the output of the program?

```
String a = "newspaper";
a = a.substring(5,7);
char b = a.charAt(1);
a = a + b;
System.out.println(a);
```

- apa
- !app
- apea
- apep

55. Which two statements are equivalent?

1. 3/2
2. 3<2
3. 3*4
4. 3<<2

- 1 and 2
- 2 and 3
- !3 and 4
- 1 and 4

56. Which statement is true?

```
class A {
  A() { }
}
class B extends A {
}
```

- Class B'S constructor is public
- !Class B'S constructor has no arguments
- Class B'S constructor includes a call to this( )
- None of these

57. Which statement is true?

- A try statement must have at least one corresponding catch block
- Multiple catch statements can catch the same class of exception more than once
- An Error that might be thrown in a method must be declared as thrown by that method, or be handled within that method
- !Except in case of VM shutdown, if a try block starts to execute, a corresponding finally block will always start to execute

58. After line 8 runs. how many objects are eligible for garbage collection?

```
class X {
  public static void main(String[] args) {
     X x = new X();
     X x2 = m1(x);
     X x4 = new X();
     x2 = x4;
     doComplexStuff(); /* Line 8 */
  }

  static X m1(X mx) {
     mx = new X();
     return mx;
  }
}
```

- 0
- !1
- 2
- 3

59. What is the widest valid returnType for methodA in line 3?

```
public class ReturnIt {
  returnType methodA(byte x, double y) /* Line 3 */
  {
     return (long)x / y * 2;
  }
}
```

- int
- long
- boolean
- !double

60. Which two cause a compiler error?

```
1. float[ ] f = new float(3);
2. float f2[ ] = new float[ ];
3. float[ ]f1 = new float[3];
4. float f3[ ] = new float[3];
5. float f5[ ] = {1.0f, 2.0f, 2.0f};
```

- 2, 4
- 3, 5
- 4, 5
- !1, 2