

6. What is the output of the following program?

```
public class Test {  
    public static void main (String [] args) {  
        try {  
            method("hi");  
            System.out.println("after method");  
        }  
        catch (NumberFormatException ex) {  
            System.out.println("main");  
        }  
    }  
    public static void method(String val) {  
        try {  
            int i = Integer.parseInt(val);  
        }  
        finally {  
            System.out.println("finally");  
        }  
        System.out.println("bye");  
    }  
}
```

- (a) finally  
main
- (b) finally  
bye  
main
- (c) finally  
main  
after method
- (d) finally  
bye  
main  
after method

## Question 1

1. What is the output of the following code (if any)?

```
public class Test {  
    public static void main (String [] args) {  
        try {  
            return;  
        }  
        finally {  
            System.out.print( "Finally" );  
        } } }  
}
```

- (a) Finally
- (b) It compiles successfully but no output
- (c) It causes a compilation error

3. Primitive data types cannot be written to an object file.

(a) Yes

(b) No

4. No conversion for a specific data type is needed when the Scanner class is used to read from a text file.

(a) Yes

(b) No

2. The following code will ...

```
public class Test {  
    public static void main (String [] args)  
    {  
        myMethod(200);  
    }  
    public static void myMethod (int n)  
    throws  
        BigIntegerException  
    {  
        if (n > 100)  
            throw new BigIntegerException();  
        System.out.println(n*2);  
    }  
}  
class BigIntegerException extends Exception  
{  
    /*Constructor*/  
}
```

- (a) throw BigIntegerException
- (b) print 400
- (c) print 400 and throw BigIntegerException
- (d) cause a compilation error

11. What is the output of the following program (if any)?

```
import java.util.*;
public class Test {
    public static void main (String [] args) {
        try {
            m1();
        }
        catch (InputMismatchException e) {
            System.out.println("catch in main");
        }
    }
    public static void m1() {
        try {
            m2();
        }
        catch (NumberFormatException e) {
            System.out.println("catch in m1");
        }
    }
    public static void m2() {
        String [] list =
        {"Love", "Java", "(2)"};
        int x = Integer.parseInt(list[4]);
    }
}
```

- (a) catch in m1
- (b) catch in main
- (c) No output, it causes a compilation error
- (d) No output, exception handled by system

```
b) public void rewardClass(String c)
//Total: 2.25
for (int i=0; i<nop;i++)
if(plist[i].getClass().getName().equals(c))
plist[i].IncrementRewared();
```





```
c) public Passenger[] longestMile(double m)
// Total: 2pt
int pCnt=0;
for (int i=0;i<nop;i++)
    if(plist[i] instanceof Business && ((Business)plist[i]).getmiles()>=m)
        pCnt++;
Passenger[] longestp = new Passenger[pCnt];
int j=0;
for (int i=0;i<nop;i++)
    if(plist[i] instanceof Business && ((Business)plist[i]).getmiles()>=m)
        longestp[j++]=new Business(plist[i].getPNO(),plist[i].getP(),
        ((Business)plist[i]).getmiles());
return longestp;
```



```

a) public boolean addPassenger(Passenger p)
//Total: 2.75
if (nop==plist.length) {
    System.out.println("Sorry, the list of passengers is full");
    return false;
}
if (p instanceof Saver)
    plist[nop++] = new Saver((Saver)p);
else if (p instanceof Plus)
    plist[nop++] = new Plus((Plus)p);
else if (p instanceof Business) {
    // counts number of business passengers in plist
    int Bcnt=0;
    for(int i=0; i<nop; i++)
        if (plist[i] instanceof Business)
            Bcnt++;
    if(Bcnt>=50) { // or Bcnt==50
        System.out.println("There is a maximum of only 50 business passenger");
        return false;
    }
    else
        plist[nop++] = new Business(p.getPNO(),p.getP(),((Business)p).getmiles());
}
// business case
return true;

```





the output of the following program?


```
public class Test {  
    static int i = 5;  
    public static void main (String [] args) {  
        try {  
            i = myMethod();  
        }  
        catch (Exception e) {  
            i--;  
        }  
        System.out.println(i);  
    }  
    public static int myMethod() throws  
    Exception {  
        try {  
            if (i >= 5)  
                throw new Exception();  
            i++;  
            return i;  
        }  
        catch (RuntimeException e) {  
            i = 0;  
        }  
        finally {  
            i = i * 2;  
        }  
        return i; } }  
}
```

- (a) 0
- (b) 9
- (c) 10
- (d) 11



7. The following code will ...

```
import java.util.*;  
public class Test {  
    public static void main (String [] args) {  
        int i = 0;  
        method(i);  
    }  
    public static void method(int n) {  
        if (n == 0)  
            throw new ArithmeticException();  
        else if (n < 0)  
            throw new IllegalArgumentException();  
    }  
}
```

- 
- (a) compile successfully
  - (b) cause a compilation error (missing throws clause in method's header)

What is the output of the following program (if any)?

```
public class Test {  
    public static void main (String [] args) {  
        try {  
            method();  
            System.out.println("end");  
        }  
        catch (StringIndexOutOfBoundsException e)  
        {  
            System.out.println("catch in main");  
        }  
    }  
    public static void method() {  
        try {  
            String str = "Hello";  
            System.out.println(str.charAt(6));  
        }  
        catch (IndexOutOfBoundsException e)  
        {  
            System.out.println("catch in method");  
        }  
    }  
}
```

- (a) catch in method
- (b) catch in main
- (c) catch in method  
end
- (d) catch in main  
end

5. What is the output of the following program?

```
public class Test {  
    public static void main (String [] args)  
    {  
        try {  
            int i, sum;  
            sum = 10;  
            for (i = -2; i < 1 ; i++) {  
                sum = (sum / i);  
                System.out.print(sum + ", ");  
            }  
        }  
        catch (ArithmeticException e) {  
            System.out.print("0");  
        }  
    }  
}
```

- (a) -5, 5,
- (b) -5, 5, 0 ✓
- (c) 0
- (d) No output

10. Java allows a class to implement more than one interface.

- (a) Yes  
(b) No