Q1. Given the code fragment:

```
class X
{
    public void printFileContent()
    {
        //Line-1
        throw new IOException();//Line-2
    }
}
public class Test
{
    public static void main(String[] args)//Line-3
    {
        X x= new X();
        x.printFileContent();//Line-4
        //Line-5
    }
}
```

Which two modifications required to compile code successfully?

A. Replace Line-3 with public static void main(String[] args) throws Exception
B. Replace Line-4 with:

```
try
{
      x.printFileContent();
}
catch (Exception e){}
catch (IOException e){}
```

C. Replace Line-3 with public static void main(String[] args) throws IOException
D. Replace Line-2 with throw IOException("Exception Raised");
E. At Line-5 insert throw new IOException();

Ans => options aren't proper here the corrected code is

```
import java.io.*;
class X
{
    public void printFileContent() throws IOException
    {
        //Line-1
        throw new IOException();//Line-2
    }
}
public class Test
{
    public static void main(String[] args)throws IOException
    {
        X x= new X();
        x.printFileContent();//Line-4
        //Line-5
    }
}
```

Q.

Q2. Given the code Fragment:

```
public class Test
{
        void readCard(int cno) throws Exception
        {
                System.out.println("Rearding Card");
        }
        void checkCard(int cno) throws RuntimeException//Line-1
        {
                System.out.println("Checking Card");
        }
        public static void main(String[] args)
        {
                Test t = new Test();
                int cardNo=1234;
                t.checkCard(cardNo);//Line-2
                t.readCard(cardNo);//Line-3
        }
}
```

What is the result?

A. Checking Card
   Reading Card

B. Compilation Fails at Line-1
C. Compilation Fails at Line-2
D. Compilation Fails at Line-3
E. Compilation Fails at Line-2 and Line-3

ans=> option D because throws exception isn't handling by main methode

Q.

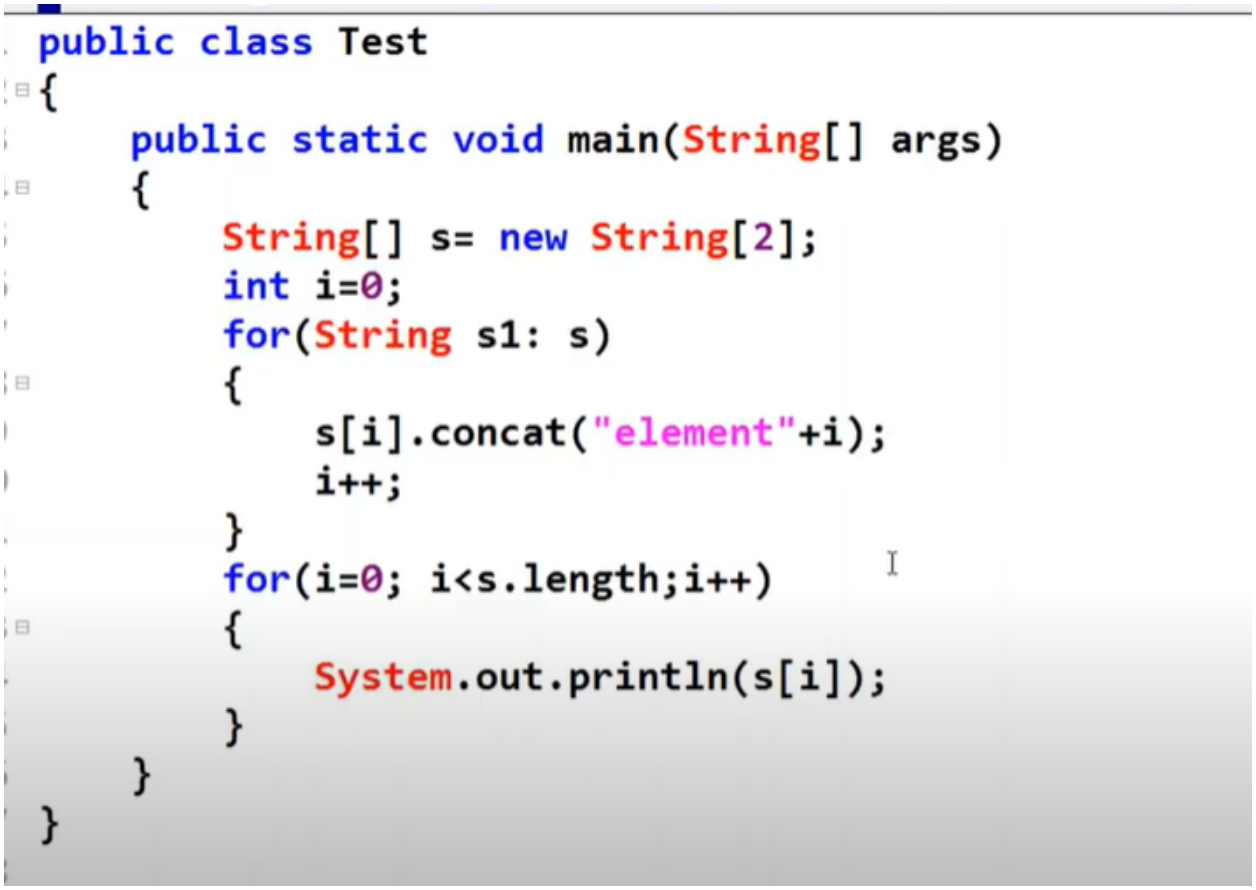
```java
public class MyException extends RuntimeException
{
}

public class Test
{
    public static void main(String[] args)
    {
        try
        {
            m1();
        }
        catch (MyException e)
        {
            System.out.print("A");
        }
    }
    public static void m1()
    {
        try
        {
            throw Math.random() > 0.5 ? new Exception():new MyException();
        }
        catch (RuntimeException e)
        {
            System.out.println("B");
        }
    }
}
```

What is the result?

A. A
B. B
C. Either A or B
D. AB
E. Compilation Fails

res=>E

Q.



```java
public class Test
{
    public static void main(String[] args)
    {
        String[] s= new String[2];
        int i=0;
        for(String s1: s)
        {
            s[i].concat("element"+i);
            i++;
        }
        for(i=0; i<s.length;i++)
        {
            System.out.println(s[i]);
        }
    }
}
```

What is the result?

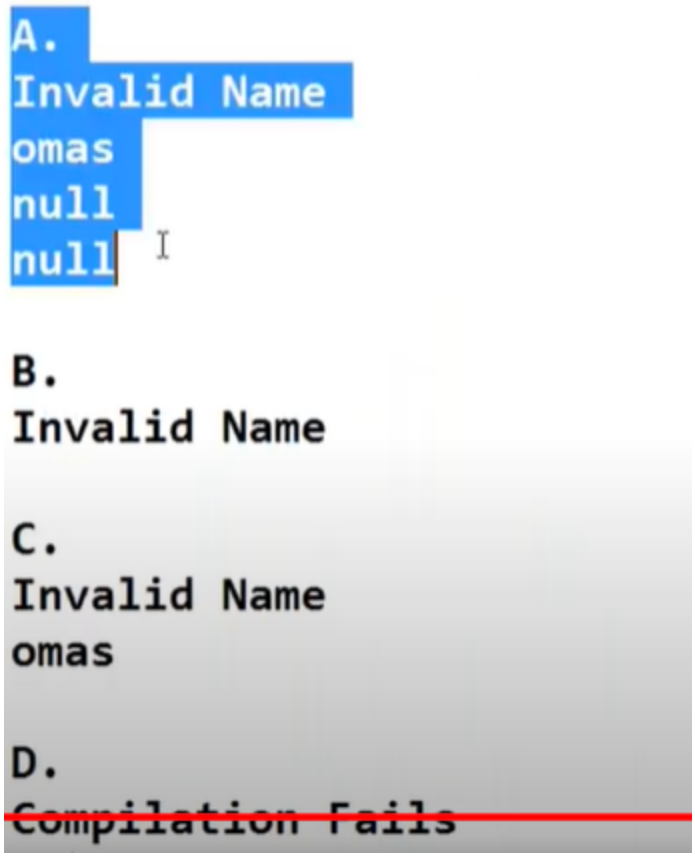
A. element 0
   element 1
B. null element 0
   null element 1
C. null
   null
D. A NullPointerException is thrown at runtime

res=>D

Q.

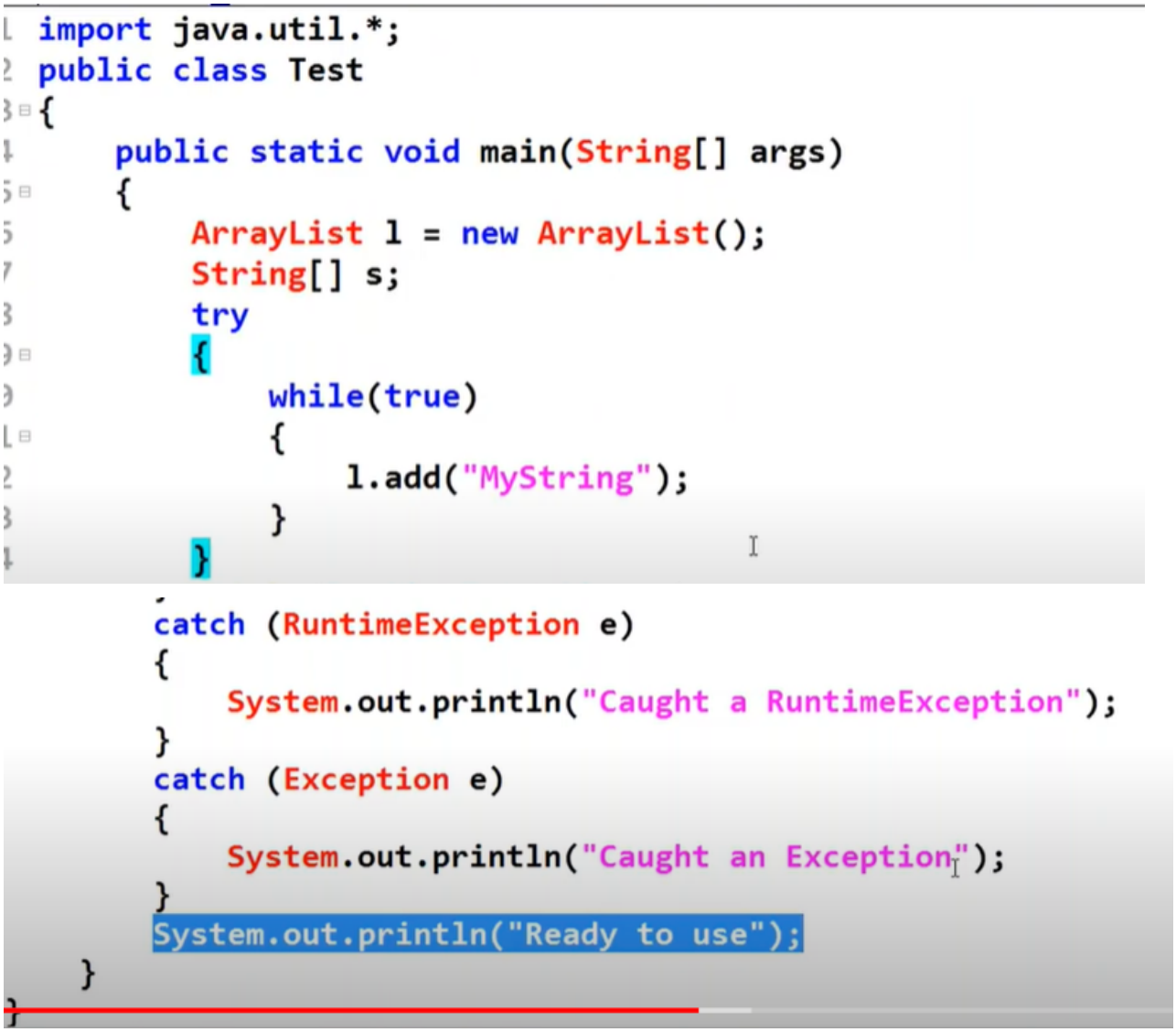
```java
public class Test
{
    public static void main(String[] args)
    {
        String[] names={"Thomas","Bunny","Chinny"};
        String[] pwds=new String[3];
        int i =0;
        try
        {
            for (String n: names)
            {
                pwds[i]=n.substring(2,6);
                i++;
            }
        }
        catch (Exception e)
        {
            System.out.println("Invalid Name");
        }
        for(String p: pwds)
        {
            System.out.println(p);
        }
    }
}
```

What is the result?



res=>A

Q.



```
import java.util.*;
public class Test
{
    public static void main(String[] args)
    {
        ArrayList l = new ArrayList();
        String[] s;
        try
        {
            while(true)
            {
                l.add("MyString");
            }
        }
        catch (RuntimeException e)
        {
            System.out.println("Caught a RuntimeException");
        }
        catch (Exception e)
        {
            System.out.println("Caught an Exception");
        }
        System.out.println("Ready to use");
    }
}
```

What is the result?

A. Caught a RuntimeException printed to the console
B. Caught an Exception printed to the console
C. A runtime error is thrown at runtime
D. Ready to use printed to the console
E. The code fails to compile because a throws keyword required

ans=>C

description=> it will not raise any exception so no catch block here will execute but after some time as while loop keep on adding element it will give out of memory error and as Error class not under Exception class so jvm will not found any handling case for Error class then it will call

default Exception handler and handle going to terminate the program abnormally with print the error info to console.

```
D:\durgaclasses>java Test
Exception in thread "main" java.lang.OutOfMemoryError: Java heap space
        at java.util.Arrays.copyOf(Unknown Source)
        at java.util.Arrays.copyOf(Unknown Source)
        at java.util.ArrayList.grow(Unknown Source)
        at java.util.ArrayList.ensureExplicitCapacity(Unknown Source)
        at java.util.ArrayList.ensureCapacityInternal(Unknown Source)
        at java.util.ArrayList.add(Unknown Source)
        at Test.main(Test.java:11)
```

Q.

```
Q7. Which three are advantages of the Java Exception Mechanism?

A. Improves the program structure because the error handing code is separated
from the normal program function.

B. Provides a set of standard exceptions that covers all possible errors

C. Improves the program structure beacuase the programmer can choose where to
handle exceptions
D. Imporves the program structure because exceptions must be handled in the
method in which they occurred.
E. Allows the creation of new exceptions that are tailored to the particular
program being created.
```

ans=> A C E

Q.

```
Q8. Which 3 statements are true about exception handling?

A. Only unchecked exceptions can be rethrown
B. All Subclasses of the RuntimeException are recoverable
C. The parameter in catch block is of throwable type
D. All subclasses of RuntimeException must be caught or declared to be thrown
E. All Subclasses of the Exception except RuntimeException class are checked
exceptions
F. All subclasses of the Error class are checked exceptions and are recoverable
```

ans=> B C E

Q.

Q9. Which two statements are true?

A. Error class is unextendable
B. Error class is extendable
C. Error is a RuntimeException
D. Error is an Exception
E. Error is a Throwable

ans=>B E