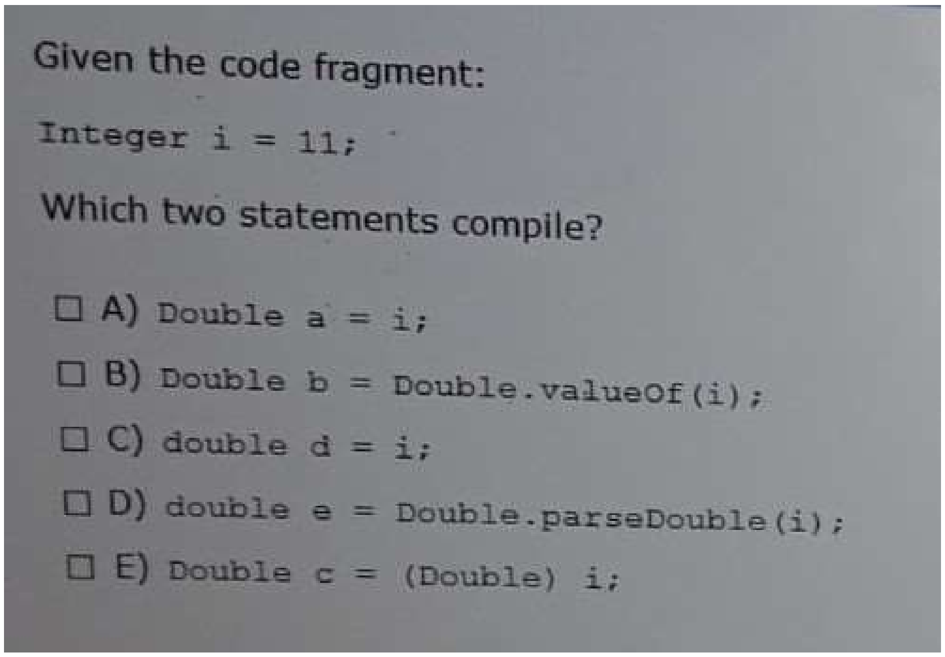
QUS no. 1:-



Given the code fragment:

```
Integer i = 11;
```

Which two statements compile?

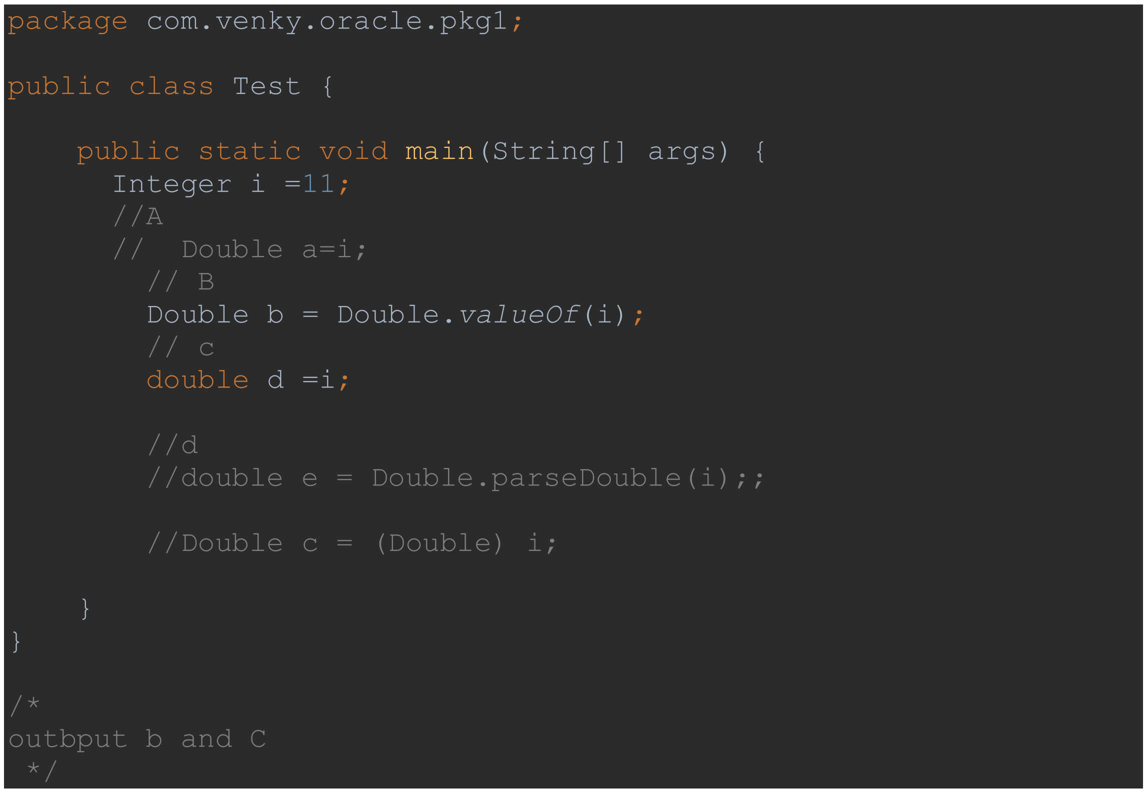
☐ A) `Double a = i;`

☐ B) `Double b = Double.valueOf(i);`

☐ C) `double d = i;`

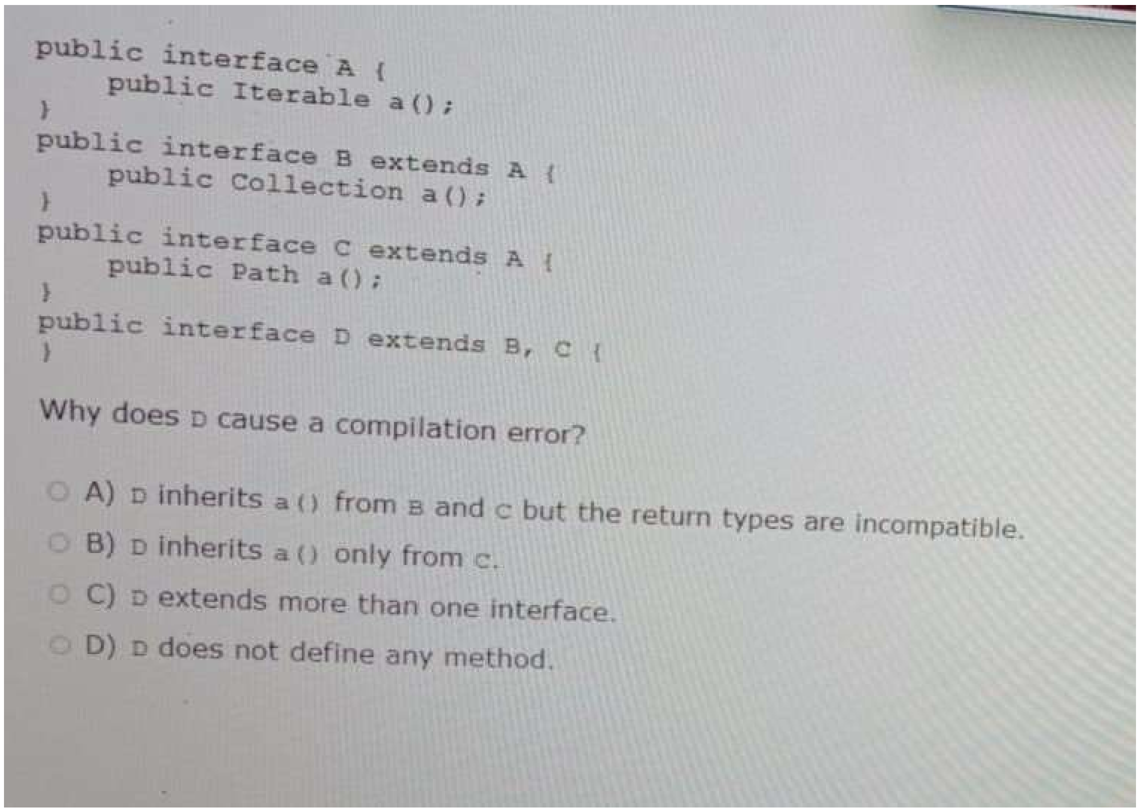
☐ D) `double e = Double.parseDouble(i);`

☐ E) `Double c = (Double) i;`



```
package com.venky.oracle.pkg1;

public class Test {

    public static void main(String[] args) {
      Integer i =11;
      //A
      //  Double a=i;
        // B
        Double b = Double.valueOf(i);
        // c
        double d =i;

        //d
        //double e = Double.parseDouble(i);;

        //Double c = (Double) i;

    }
}

/*
outbput b and C
 */
```

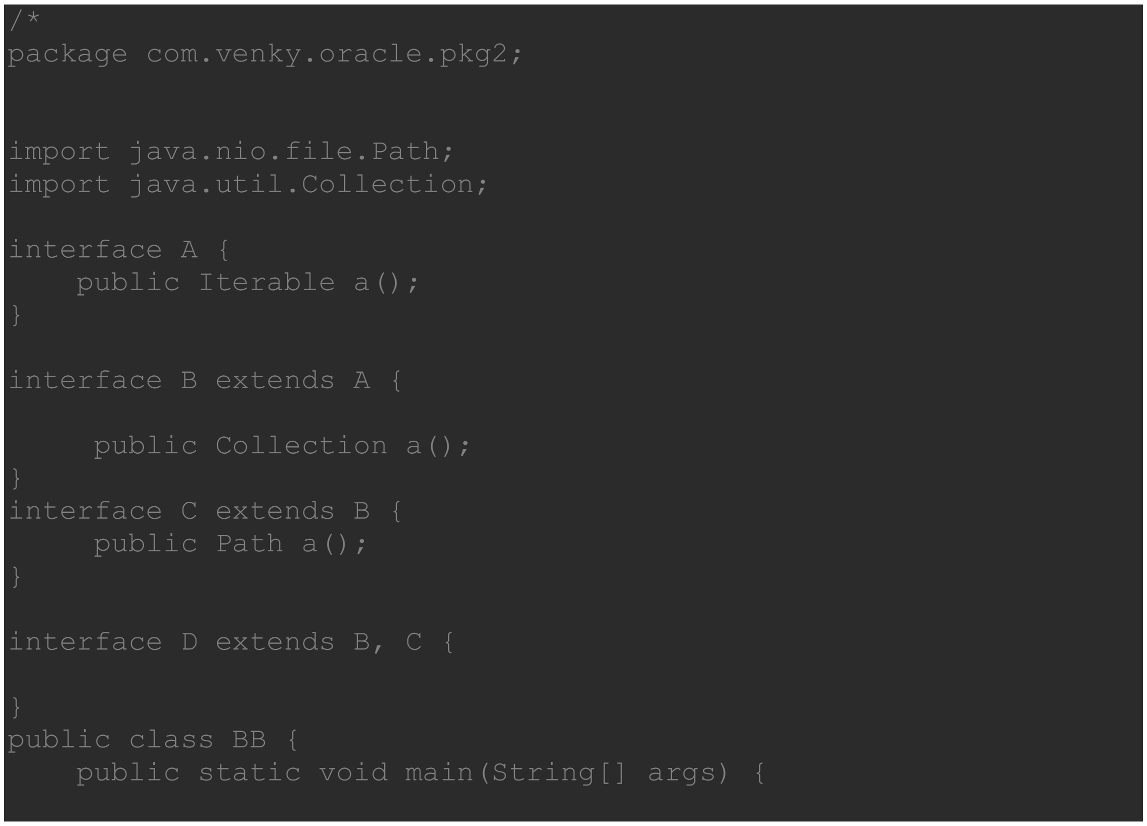
Qus no :- 2



```
public interface A {
    public Iterable a();
}
public interface B extends A {
    public Collection a();
}
public interface C extends A {
    public Path a();
}
public interface D extends B, C {
}
```

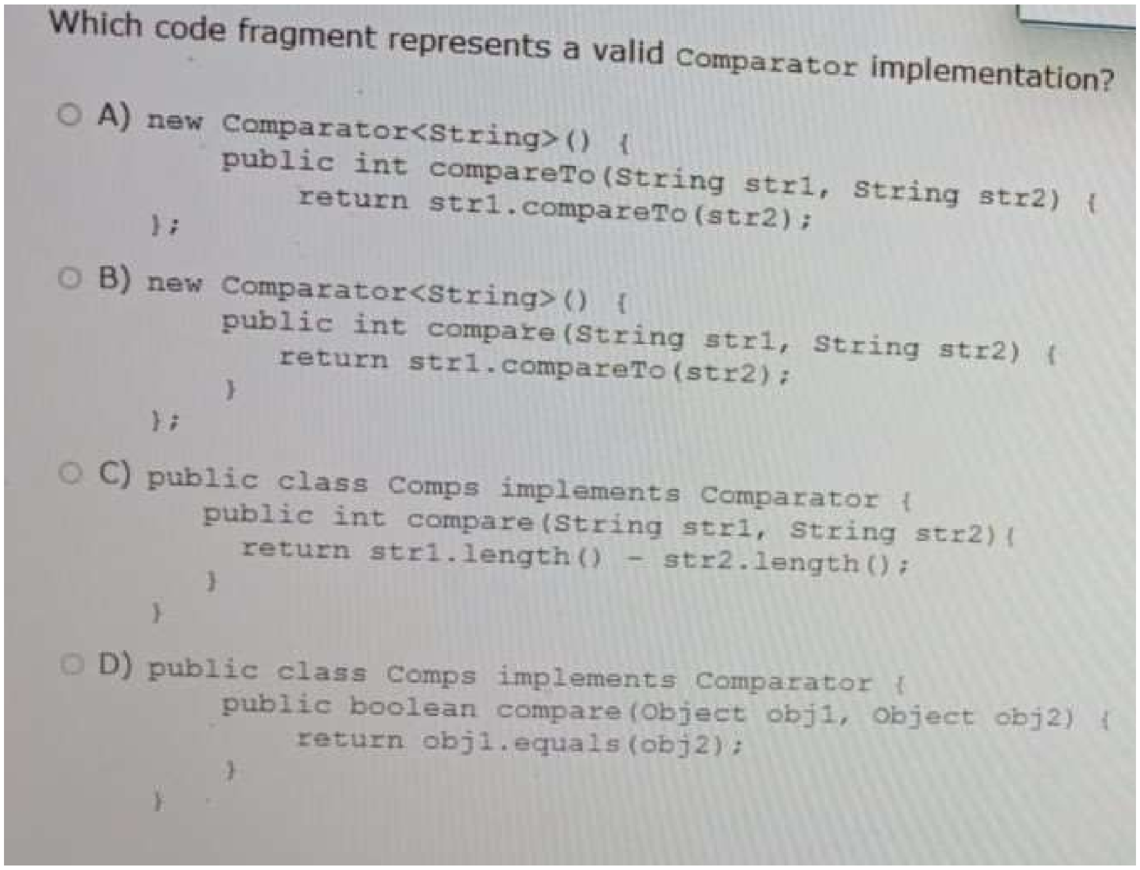
Why does D cause a compilation error?

- A) D inherits a() from B and C but the return types are incompatible.
- B) D inherits a() only from C.
- C) D extends more than one interface.
- D) D does not define any method.



```
/*
package com.venky.oracle.pkg2;


import java.nio.file.Path;
import java.util.Collection;

interface A {
    public Iterable a();
}

interface B extends A {

     public Collection a();
}
interface C extends B {
     public Path a();
}

interface D extends B, C {

}
public class BB {
    public static void main(String[] args) {
    }
}
*/

/*
Outbput opiton A
B and C  have diffrent Rturn type
 */
```

Qus no :-3

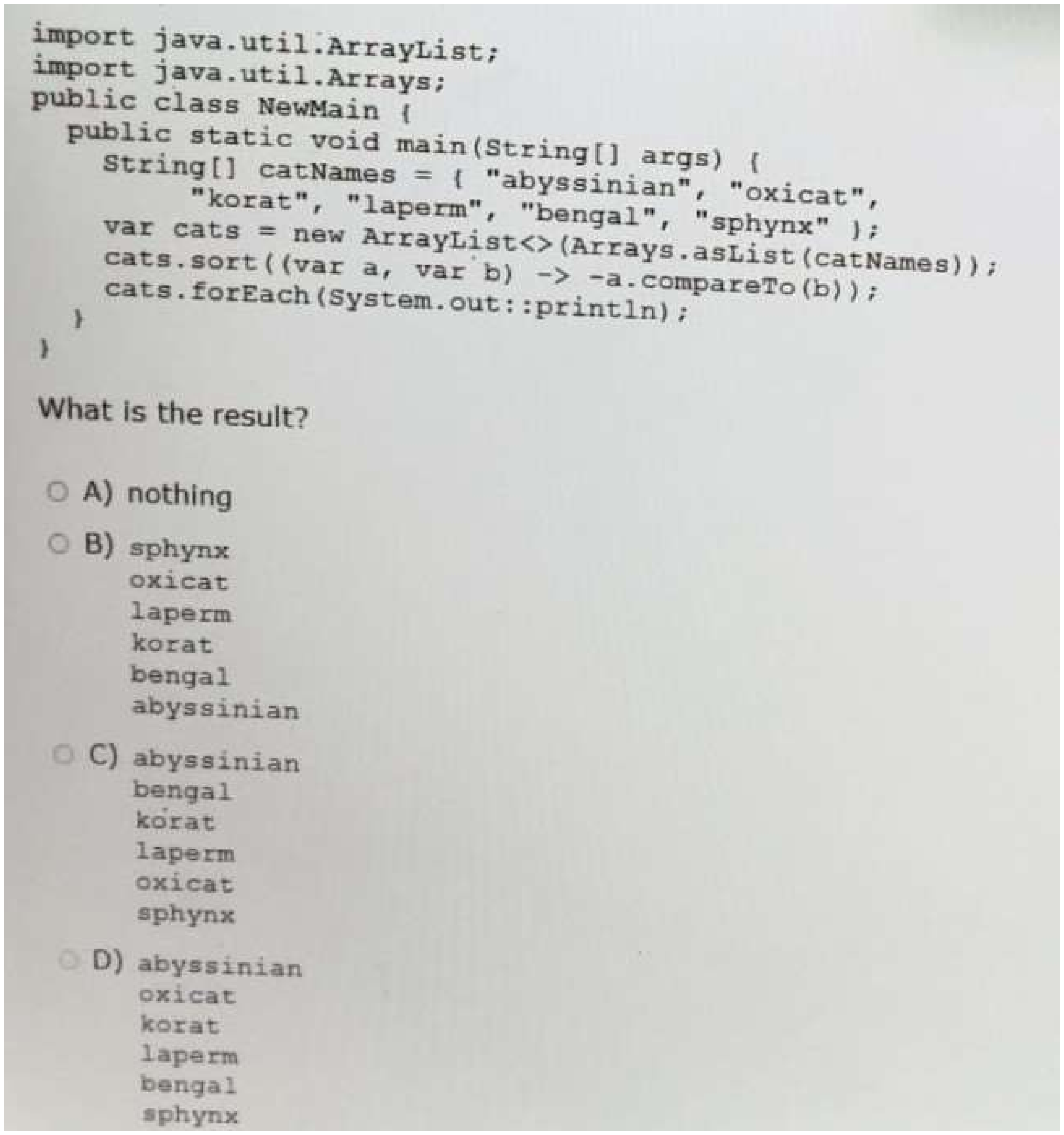


Which code fragment represents a valid Comparator implementation?

- A)
```
new Comparator<String>() {
    public int compareTo(String str1, String str2) {
        return str1.compareTo(str2);
};
```
- B)
```
new Comparator<String>() {
    public int compare(String str1, String str2) {
        return str1.compareTo(str2);
    }
};
```
- C)
```
public class Comps implements Comparator {
    public int compare(String str1, String str2){
        return str1.length() - str2.length();
    }
}
```
- D)
```
public class Comps implements Comparator {
    public boolean compare(Object obj1, Object obj2) {
        return obj1.equals(obj2);
    }
}
```

```
package com.venky.oracle.pkg3;

import java.util.Comparator;

public class Test {
    public static void main(String[] args) {

        Comparator c = new Comparator <String>() {
            public int compare(String o1, String o2) {
                return o1.compareTo(o2);
            }
        };
    }
}
/*
OPTION   B
 */
```

Qus no :-4



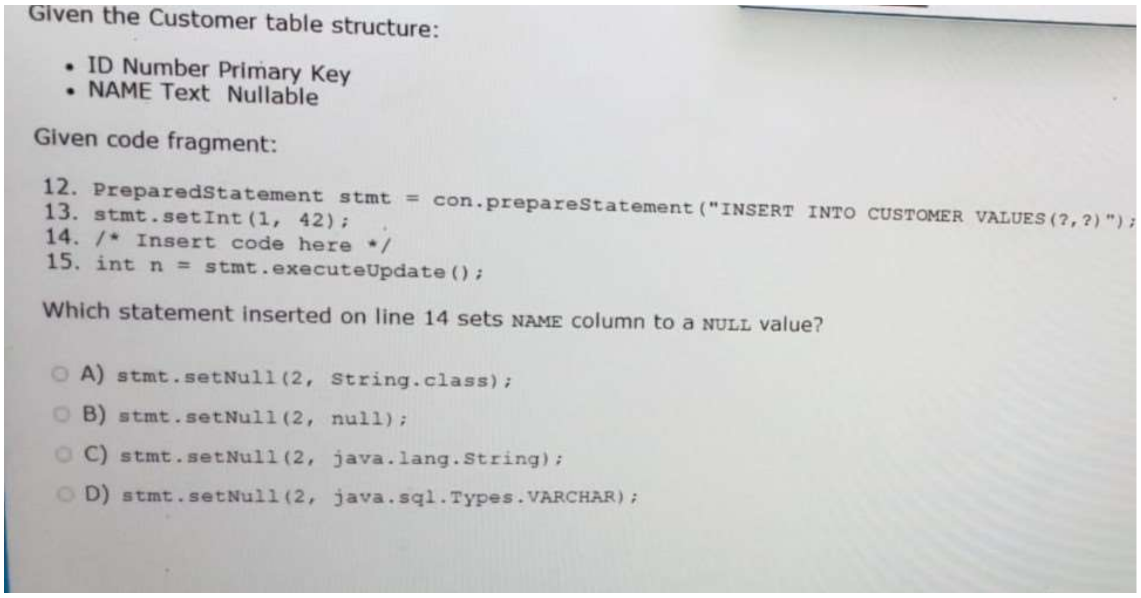
```
import java.util.ArrayList;
import java.util.Arrays;
public class NewMain {
  public static void main(String[] args) {
    String[] catNames = { "abyssinian", "oxicat",
         "korat", "laperm", "bengal", "sphynx" };
    var cats = new ArrayList<>(Arrays.asList(catNames));
    cats.sort((var a, var b) -> -a.compareTo(b));
    cats.forEach(System.out::println);
  }
}
```

What is the result?

- A) nothing
- B) sphynx
  oxicat
  laperm
  korat
  bengal
  abyssinian
- C) abyssinian
  bengal
  korat
  laperm
  oxicat
  sphynx
- D) abyssinian
  oxicat
  korat
  laperm
  bengal
  sphynx

```
package com.venky.oracle.pkg4;


import java.util.ArrayList;
import java.util.Arrays;

public class NewMain {
    public static void main(String[] args) {

        String [] castsName = {"abyssinan", "oxicat", "korant",
"laperm", "bengal", "shynz"};
        var cats = new ArrayList<>(Arrays.asList(castsName));
        cats.sort((var a,var b) -> a.compareTo(b));
        cats.forEach(System.out:: println);



    }
}
/*

OUT PUT
abyssinan
bengal
korant
laperm
oxicat
shynz
 */
```

Qus no:-5



Given the Customer table structure:

- ID Number Primary Key
- NAME Text Nullable

Given code fragment:

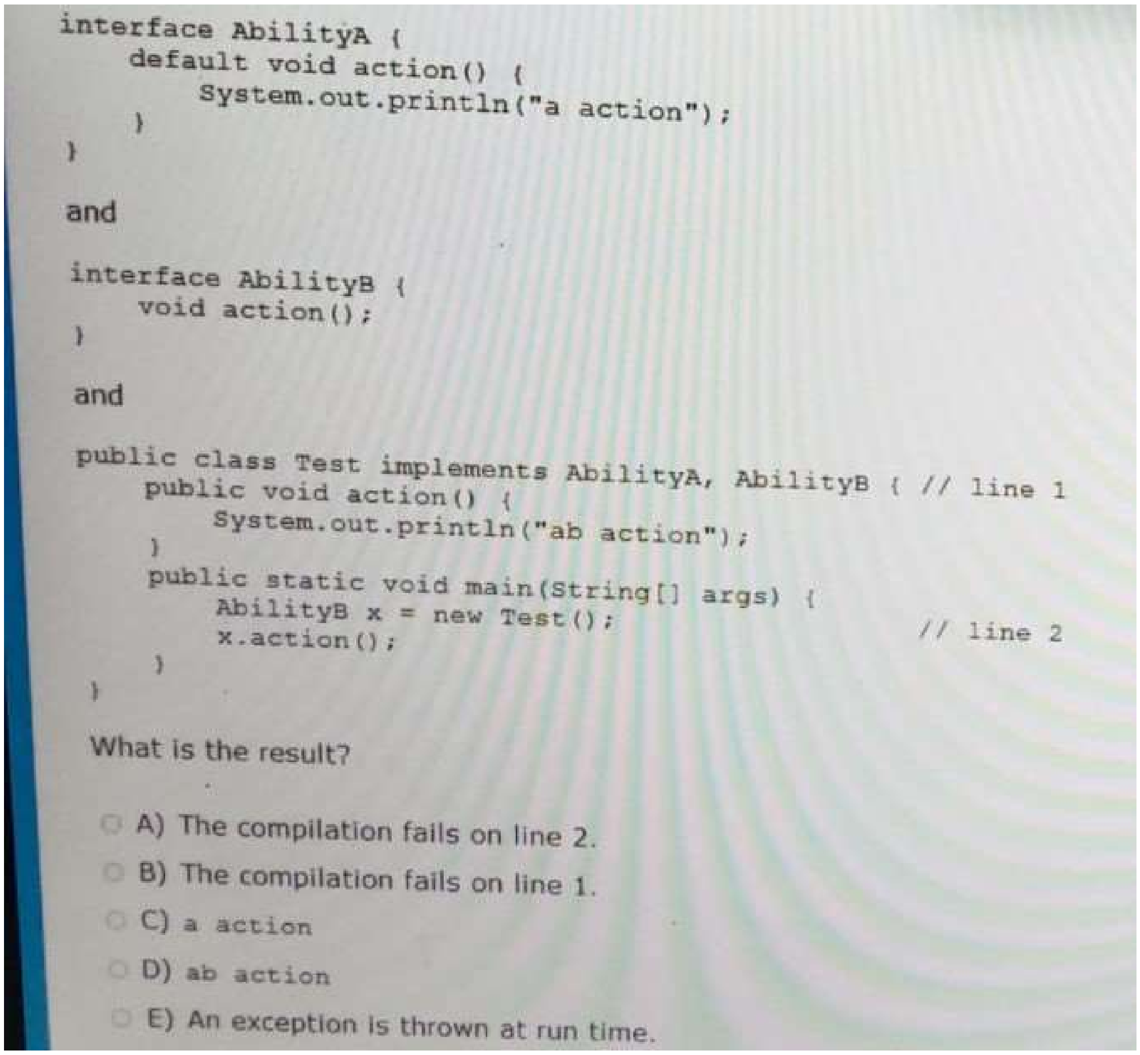
```
12. PreparedStatement stmt = con.prepareStatement("INSERT INTO CUSTOMER VALUES(?,?)");
13. stmt.setInt(1, 42);
14. /* Insert code here */
15. int n = stmt.executeUpdate();
```

Which statement inserted on line 14 sets NAME column to a NULL value?

- A) `stmt.setNull(2, String.class);`
- B) `stmt.setNull(2, null);`
- C) `stmt.setNull(2, java.lang.String);`
- D) `stmt.setNull(2, java.sql.Types.VARCHAR);`

```
package com.venky.oracle.pkg5;

import java.sql.PreparedStatement;
import java.sql.Types;

public class Test {
    public static void main(String[] args) {
       // PreparedStatement ps = con.prepareStatement("insert
into table1 values(?,?,?)");
    //    ps.setNull(1, Types.VARCHAR);
    }
}

//Option  D for the nullable column.
//
```

Qus no :- 6



```
interface AbilityA {
    default void action() {
        System.out.println("a action");
    }
}
```

and

```
interface AbilityB {
    void action();
}
```

and

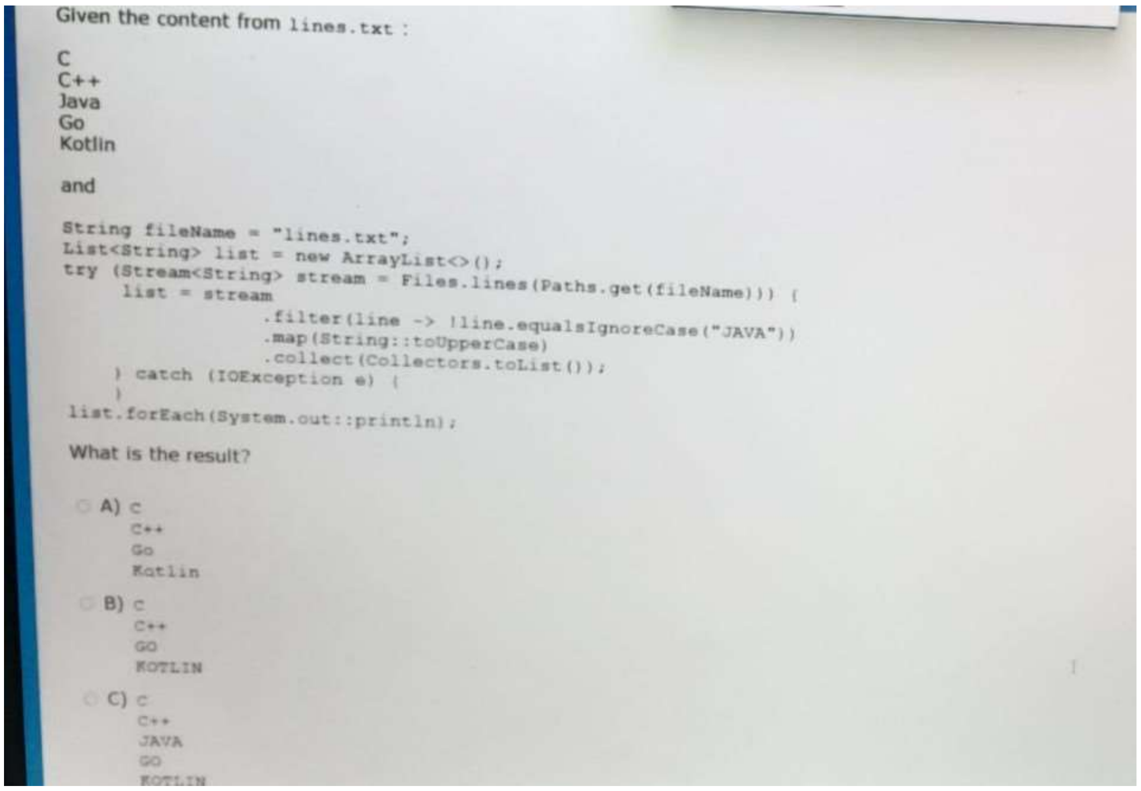
```
public class Test implements AbilityA, AbilityB { // line 1
    public void action() {
        System.out.println("ab action");
    }
    public static void main(String[] args) {
        AbilityB x = new Test();                    // line 2
        x.action();
    }
}
```

What is the result?

- A) The compilation fails on line 2.
- B) The compilation fails on line 1.
- C) a action
- D) ab action
- E) An exception is thrown at run time.

```java
package com.venky.oracle.pkg6;

interface AbilityA {
    default void action() {
        System.out.println("a action");
    }
}
interface AbilityB {
     void action();
}
public class Test implements AbilityA, AbilityB {
    public static void main (String[] args) {
          AbilityA x = new Test();
          x.action();
    }
    public void action() {
        System.out.println("ab action");
    }
}

/*
Output:
ab action
 */
```

Qus no :-7



Given the content from `lines.txt` :

C
C++
Java
Go
Kotlin

and

```java
String fileName = "lines.txt";
List<String> list = new ArrayList<>();
try (Stream<String> stream = Files.lines(Paths.get(fileName))) {
      list = stream
                  .filter(line -> !line.equalsIgnoreCase("JAVA"))
                  .map(String::toUpperCase)
                  .collect(Collectors.toList());
     } catch (IOException e) {
     }
list.forEach(System.out::println);
```

What is the result?

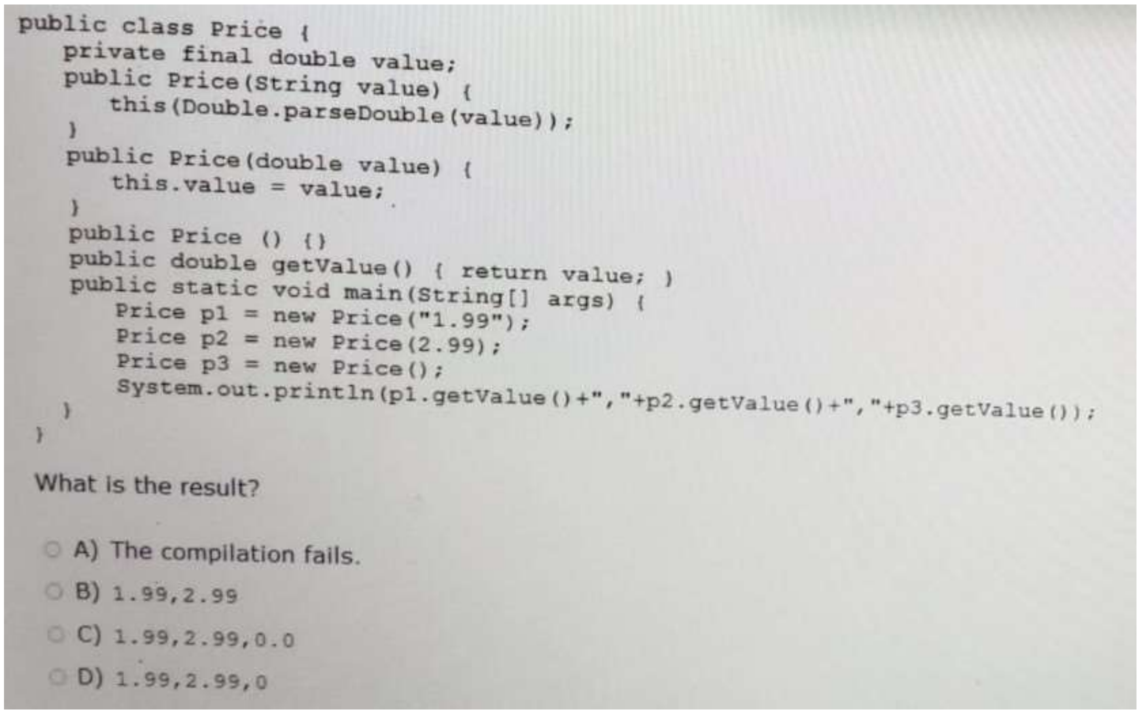
A) c
C++
Go
Kotlin

B) c
C++
GO
KOTLIN

C) c
C++
JAVA
GO
KOTLIN

```
package com.venky.oracle.pkg7;

import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.List;
import java.util.stream.Collectors;
import java.util.stream.Stream;

public class Test {
    public static void main(String[] args) throws IOException {
        String fileName =
"F:\\MyCourse\\repo\\algorithm\\src\\main\\java\\com\\venky\\orac
le\\pkg7\\lines.txt";
        List< String > list = new ArrayList< >();
        try (Stream< String > stream =
Files.lines(Paths.get(fileName))) {
            list = stream
                    .filter(line ->
!line.equalsIgnoreCase("JAVA")).map(String::toUpperCase)
                    .collect(Collectors.toList());
        }
catch (IOException e)
            {

            }
        list.forEach(System.out :: println);
    }
}/**
 C
 C++
 GO
 KOTLIN
 */

*******************Ans ********************************
//c
//c++
//Java
//go
//kotlin
```

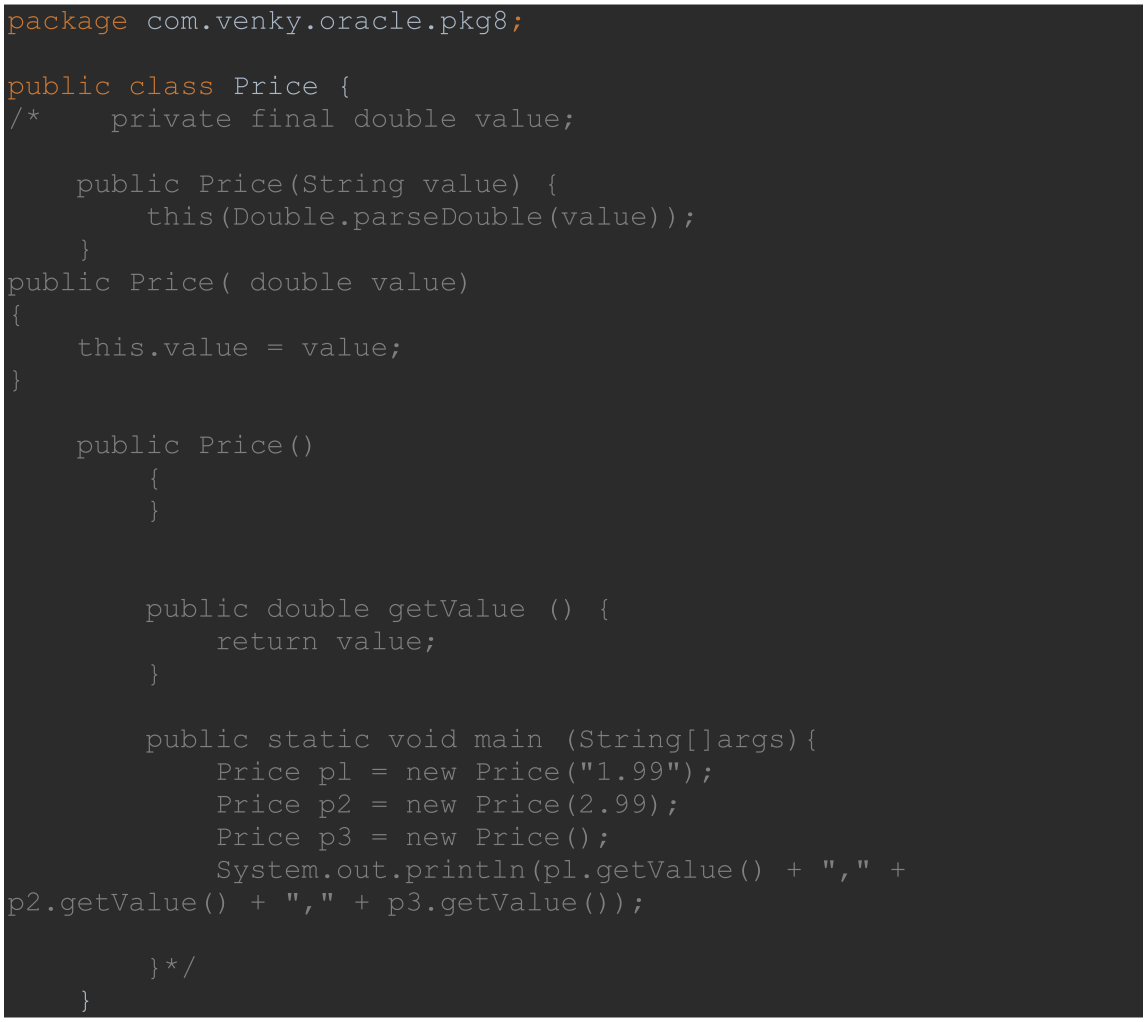
Qus no :- 8



```
public class Price {
    private final double value;
    public Price(String value) {
        this(Double.parseDouble(value));
    }
    public Price(double value) {
        this.value = value;
    }
    public Price () {}
    public double getValue() { return value; }
    public static void main(String[] args) {
        Price p1 = new Price("1.99");
        Price p2 = new Price(2.99);
        Price p3 = new Price();
        System.out.println(p1.getValue()+","+p2.getValue()+","+p3.getValue());
    }
}
```

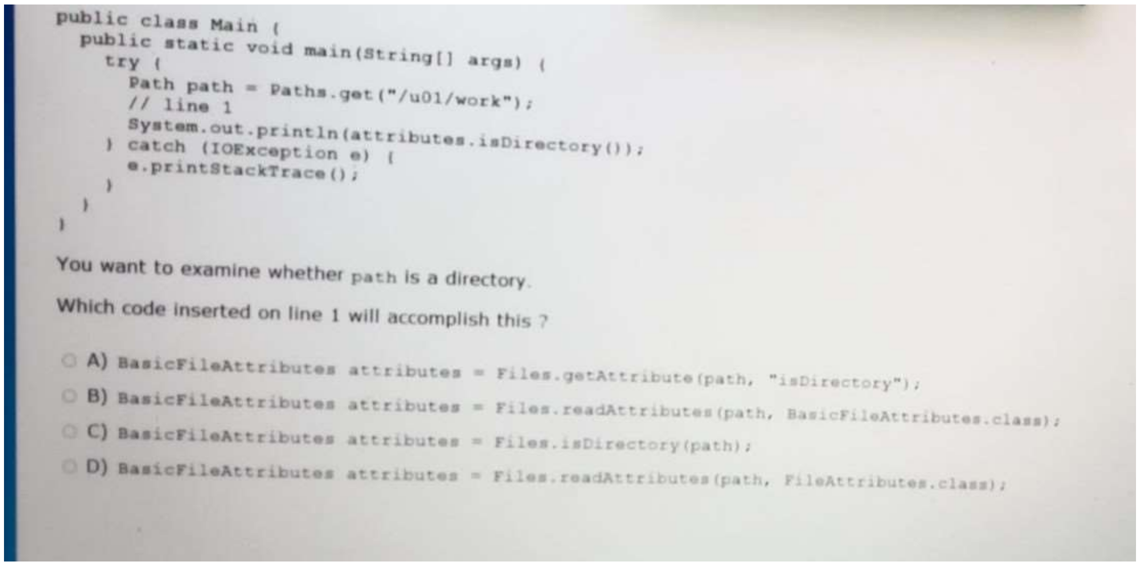
What is the result?

- A) The compilation fails.
- B) `1.99,2.99`
- C) `1.99,2.99,0.0`
- D) `1.99,2.99,0`



```
package com.venky.oracle.pkg8;

public class Price {
/*    private final double value;

    public Price(String value) {
        this(Double.parseDouble(value));
    }
public Price( double value)
{
    this.value = value;
}

    public Price()
        {
        }


        public double getValue () {
            return value;
        }

        public static void main (String[]args){
            Price pl = new Price("1.99");
            Price p2 = new Price(2.99);
            Price p3 = new Price();
            System.out.println(pl.getValue() + "," +
p2.getValue() + "," + p3.getValue());

        }*/
    }
    /*
    Output: Compilation Fails
    java: variable value might not have been initialized
     */
```

Qus no :- 9



```
public class Main {
  public static void main(String[] args) {
    try {
      Path path = Paths.get("/u01/work");
      // line 1
      System.out.println(attributes.isDirectory());
    } catch (IOException e) {
      e.printStackTrace();
    }
  }
}
```

You want to examine whether path is a directory.

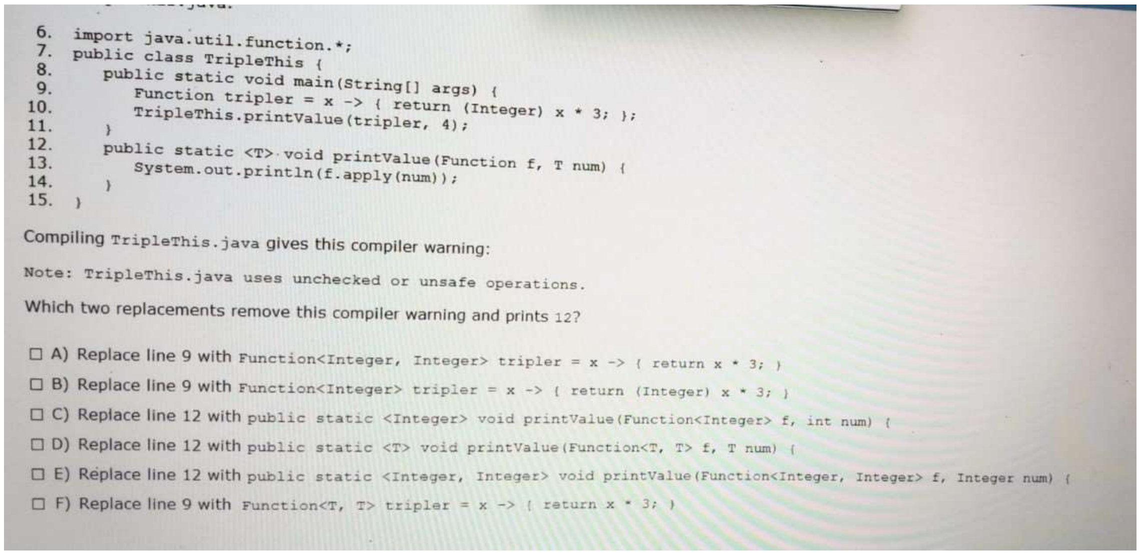
Which code inserted on line 1 will accomplish this ?

- A) BasicFileAttributes attributes = Files.getAttribute(path, "isDirectory");
- B) BasicFileAttributes attributes = Files.readAttributes(path, BasicFileAttributes.class);
- C) BasicFileAttributes attributes = Files.isDirectory(path);
- D) BasicFileAttributes attributes = Files.readAttributes(path, FileAttributes.class);

```
package com.venky.oracle.pkg9;

import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.nio.file.attribute.BasicFileAttributes;

public class Main {
    public static void main(String[] args) {
        try {
            Path path =
Paths.get("F:\\MyCourse\\repo\\algorithm\\src\\main\\java\\com\\v
enky\\oracle\\pkg9");
            // line 1

            //Option A
            //    BasicFileAttributes attributes =
Files.getAttribute(path, "isDirectory");
            // Option     B
            BasicFileAttributes attributes =
Files.readAttributes(path, BasicFileAttributes.class);
            // Option     C
            //   BasicFileAttributes attributes =
Files.readAttributes(path);
```

```
            //Option  D
            //    BasicFileAttributes attributes =
Files.readAttributes(path, FileAttributes.class);
            System.out.println(attributes.isDirectory());
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Qus no :- 10



```
6.  import java.util.function.*;
7.  public class TripleThis {
8.      public static void main(String[] args) {
9.          Function tripler = x -> { return (Integer) x * 3; };
10.         TripleThis.printValue(tripler, 4);
11.     }
12.     public static <T> void printValue(Function f, T num) {
13.         System.out.println(f.apply(num));
14.     }
15. }
```

Compiling `TripleThis.java` gives this compiler warning:

`Note: TripleThis.java uses unchecked or unsafe operations.`

Which two replacements remove this compiler warning and prints `12`?

- ☐ A) Replace line 9 with `Function<Integer, Integer> tripler = x -> { return x * 3; }`
- ☐ B) Replace line 9 with `Function<Integer> tripler = x -> { return (Integer) x * 3; }`
- ☐ C) Replace line 12 with `public static <Integer> void printValue(Function<Integer> f, int num) {`
- ☐ D) Replace line 12 with `public static <T> void printValue(Function<T, T> f, T num) {`
- ☐ E) Replace line 12 with `public static <Integer, Integer> void printValue(Function<Integer, Integer> f, Integer num) {`
- ☐ F) Replace line 9 with `Function<T, T> tripler = x -> { return x * 3; }`

```
package com.venky.oracle.pkg10;

import java.util.function.Function;
/* Baase problem
*/
public class TripleThis {
    public static void main(String[] args) {

        Function  tripler = x->{return  (Integer)x*3;};
        TripleThis.printValue(tripler,4);
    }

    private static <T>  void printValue(Function f, T num) {
        System.out.println(f.apply(num));
    }
}

/*
Option D an A

 */
```

```java
package com.venky.oracle.pkg10;

import java.util.function.Function;

public class TripleThisA {
    public static void main(String[] args) {

        Function<Integer,Integer > tripler = x->{return x*3;};
        TripleThisA.printValue(tripler,4);
    }

    private static <T>  void printValue(Function f, T num) {
        System.out.println(f.apply(num));
    }
}
```

```java
package com.venky.oracle.pkg10;

import java.util.function.Function;
/*

public class TripleThisB {
    public static void main(String[] args) {

        Function<Integer > tripler = x->{return(Integer) x*3;};
        TripleThisB.printValue(tripler,4);
    }

    private static <T>  void printValue(Function f, T num) {
        System.out.println(f.apply(num));
    }
}
*/

//THis Code is not working.
```

```java
package com.venky.oracle.pkg10;

import java.util.function.Function;

/*
public class TripleThisC {
    public static void main(String[] args) {

        Function  tripler = x->{return (Integer)x*3;};
        TripleThisC.printValue(tripler,4);
    }
    private static <Integer>  void printValue(Function <Integer>
f, int num) {
        System.out.println(f.apply(num));
    }
}
*/

//THis code is not goign to compile Function required two type
parameters and one is Integer
```

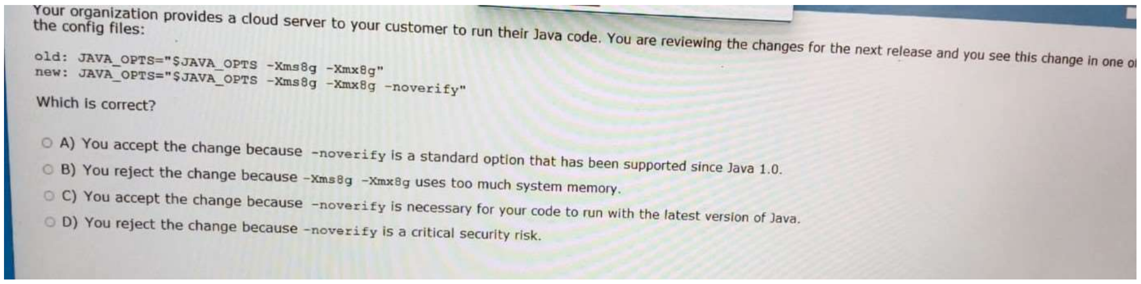
```
package com.venky.oracle.pkg10;

import java.util.function.Function;

public class TripleThisD {
    public static void main(String[] args) {

        Function tripler = x->{return (Integer)x*3;};
        TripleThisD.printValue(tripler,4);
    }

    private static <T>  void printValue(Function <T,T> f, T num)
{
        System.out.println(f.apply(num));
    }
}
// Remove one  This could be potential correct answer.
//Unchecked call to 'apply(T)' as a member of raw type
'java.util.function.Function'
```

```
package com.venky.oracle.pkg10;

import java.util.function.Function;

/*
public class TripleThisE {
    public static void main(String[] args) {

        Function tripler = x->{return (Integer)x*3;};
        TripleThisE.printValue(tripler,4);
    }

    private static <Integer,Integer>  void printValue(Function
<Integer,Integer> f, T num) {
        System.out.println(f.apply(num));
    }
}
*/
```

```
// Compilation failed Two PAramenter are not allowed return type
is Integer
```

```
package com.venky.oracle.pkg10;

import java.util.function.Function;

/*
public class TripleThisF {
    public static void main(String[] args) {

        Function <T,T> tripler = x->{return x*3;};
        TripleThisF.printValue(tripler,4);
    }

    private static <T>  void printValue(Function f, T num) {
        System.out.println(f.apply(num));
    }
}
*/
//Option F is also not going to compile.
```

Qus no:- 11



Your organization provides a cloud server to your customer to run their Java code. You are reviewing the changes for the next release and you see this change in one o
the config files:

```
old: JAVA_OPTS="$JAVA_OPTS -Xms8g -Xmx8g"
new: JAVA_OPTS="$JAVA_OPTS -Xms8g -Xmx8g -noverify"
```

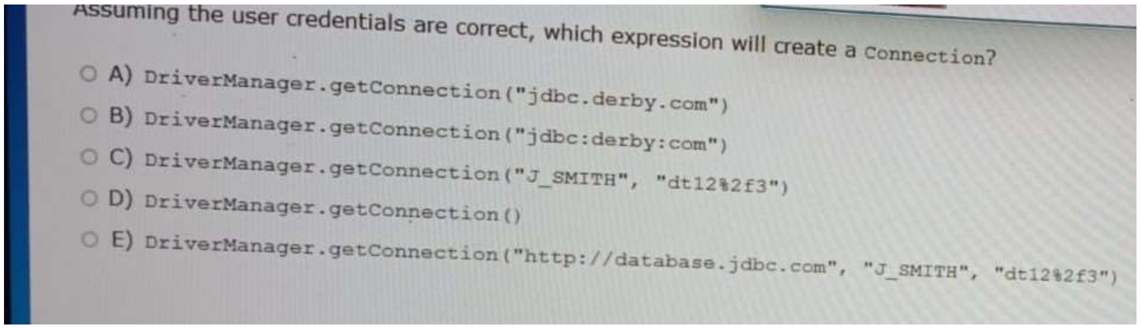
Which is correct?

- A) You accept the change because `-noverify` is a standard option that has been supported since Java 1.0.
- B) You reject the change because `-Xms8g -Xmx8g` uses too much system memory.
- C) You accept the change because `-noverify` is necessary for your code to run with the latest version of Java.
- D) You reject the change because `-noverify` is a critical security risk.

```
==>>>>OPTION -D

Start-up time, I'd say. Verification that classes are correct
takes some time when the class is loaded.
 Since classes might be loaded in a lazy fashion (not on app
start, but when being used for the first time), this might cause
unexpected and undesired runtime delays.

Actually the class does not need to be checked in general. The
compiler will not emit any invalid bytecode or class construct.
The reason for verification is that the class may be build on one
system, get hosted online and is transmitted to you through the
unprotected internet.
```

```
On this path, a malicious attacker might modify the bytecode and
create something the compiler might never create; something that
can crash the JVM or possibly
circumvents security restrictions.
Thus the class is verified before it is used.
If this is a local application, there is usually no need to check
the bytecode again.
```

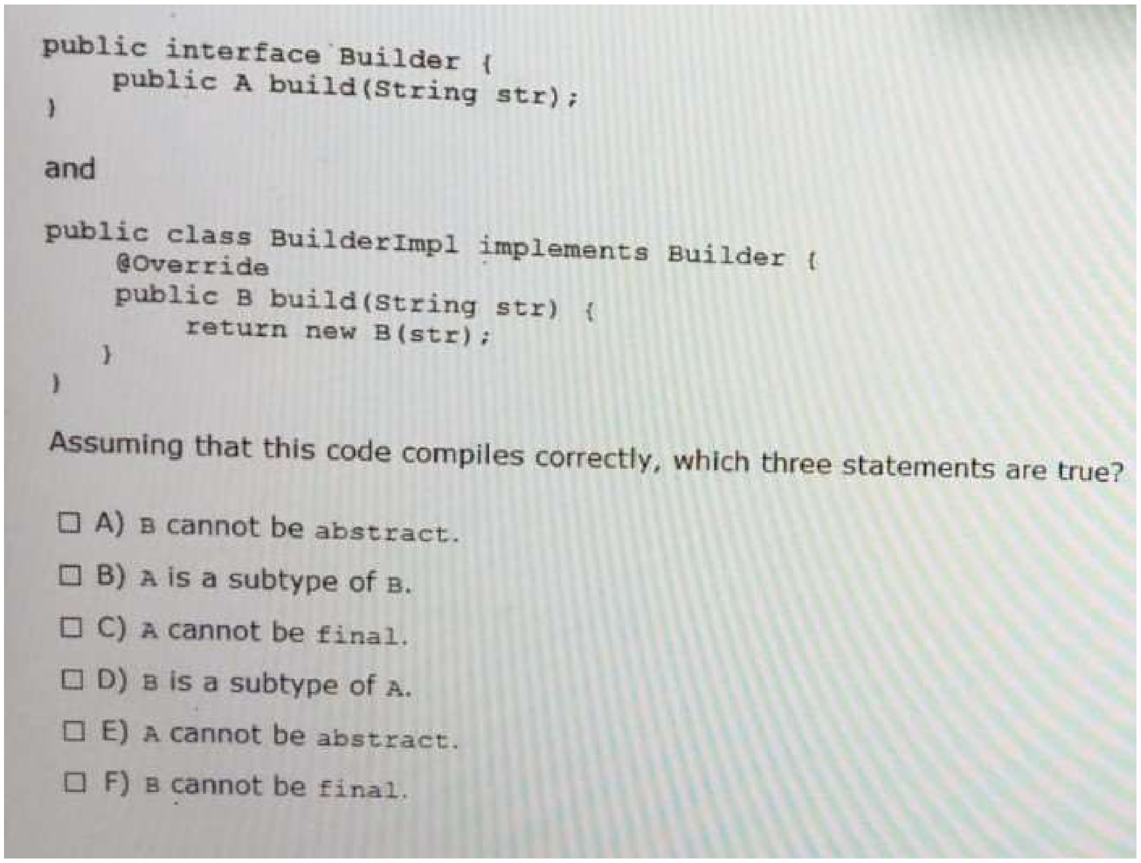
Qus no:- 12


Assuming the user credentials are correct, which expression will create a Connection?

- A) DriverManager.getConnection("jdbc.derby.com")
- B) DriverManager.getConnection("jdbc:derby:com")
- C) DriverManager.getConnection("J_SMITH", "dt12%2f3")
- D) DriverManager.getConnection()
- E) DriverManager.getConnection("http://database.jdbc.com", "J_SMITH", "dt12%2f3")

```
#################################################################
#############################
String dbURL = "jdbc:derby://localhost/webdb;create=true";
String user = "tom";
String password = "secret";
Connection conn = DriverManager.getConnection(dbURL, user,
password);
#################################################################
#############################
String dbURL = "jdbc:derby:codejava/webdb;create=true";
Connection conn = DriverManager.getConnection(dbURL);
#################################################################
#############################
```

Qus no:-13



```
public interface Builder {
    public A build(String str);
}
```

and

```
public class BuilderImpl implements Builder {
    @Override
    public B build(String str) {
        return new B(str);
    }
}
```

Assuming that this code compiles correctly, which three statements are true?

- ☐ A) `B` cannot be `abstract`.
- ☐ B) `A` is a subtype of `B`.
- ☐ C) `A` cannot be `final`.
- ☐ D) `B` is a subtype of `A`.
- ☐ E) `A` cannot be `abstract`.
- ☐ F) `B` cannot be `final`.

```
package com.venky.oracle.pkg13;

public  class  A {

}
```

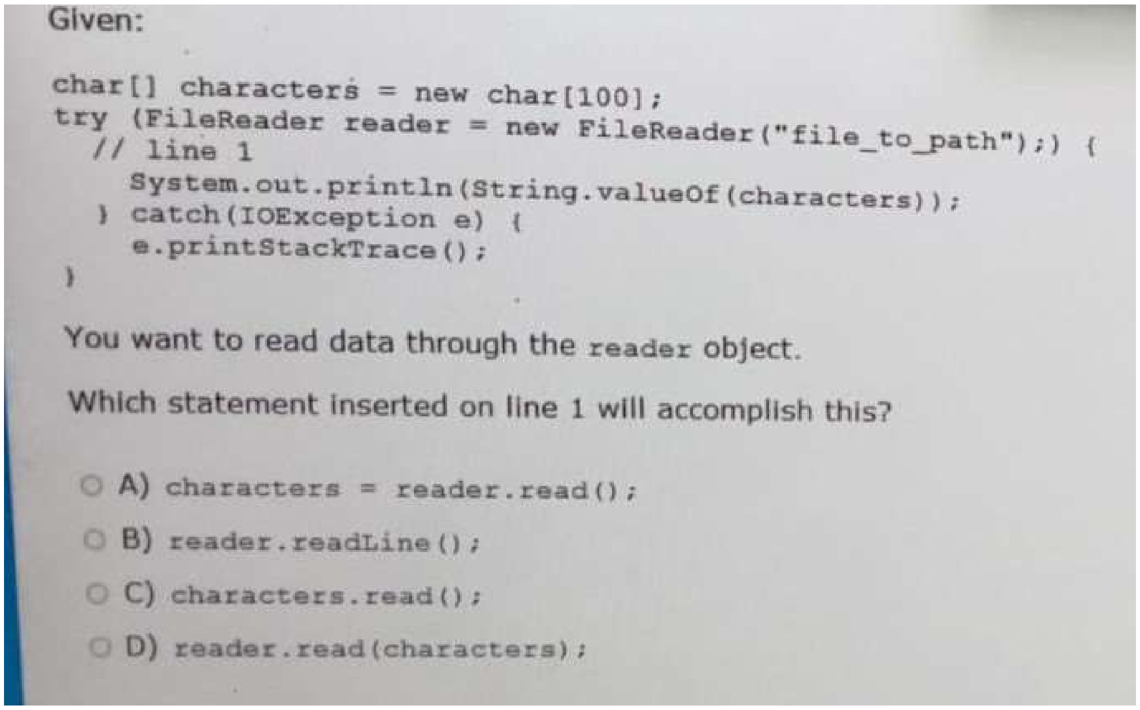
```
package com.venky.oracle.pkg13;

public   class B extends A {

    public B(String str) {
    }
}
```

```
package com.venky.oracle.pkg13;

public interface Builder {
    public A build(String str);

}
```

```
package com.venky.oracle.pkg13;


public class BuilderImpl implements Builder {

    @Override
    public A build(String str) {
        return new B(str);
    }
}
```

```
A --> B cannot be abstract
C --> A cannot be final
D--> B is subtype of A
```

Qus no:- 14


Given:

```
char[] characters = new char[100];
try (FileReader reader = new FileReader("file_to_path");) {
   // line 1
     System.out.println(String.valueOf(characters));
   } catch(IOException e) {
     e.printStackTrace();
}
```

You want to read data through the reader object.

Which statement inserted on line 1 will accomplish this?

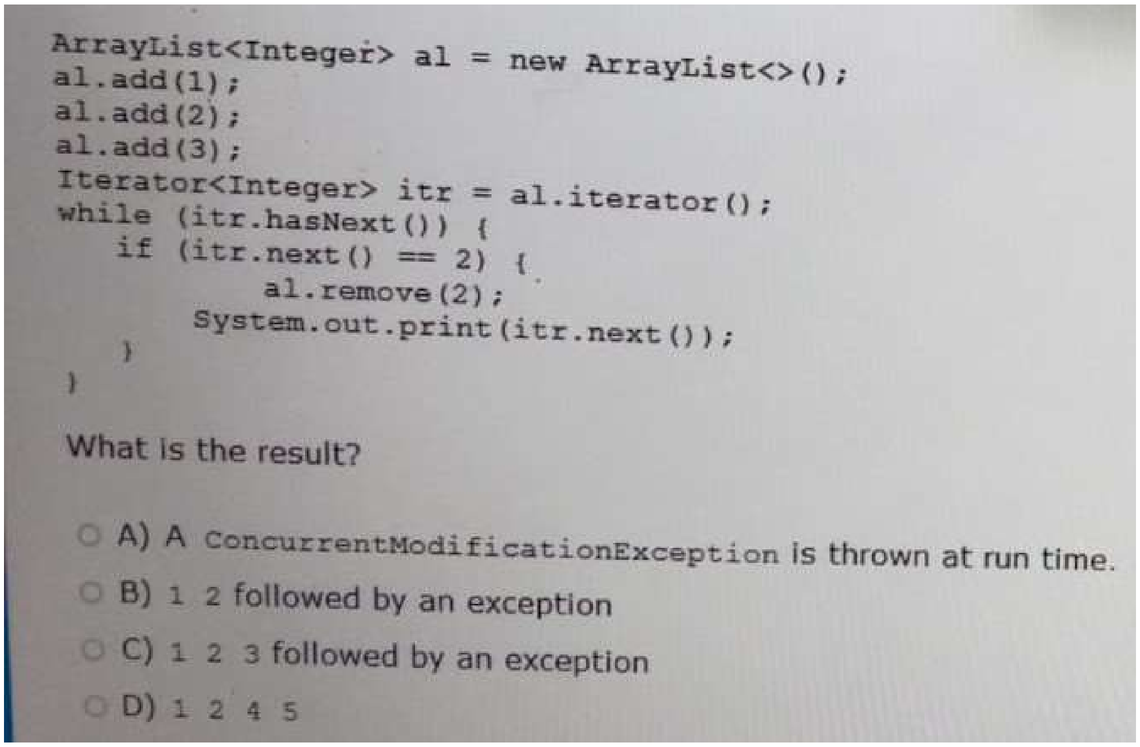
- A) `characters = reader.read();`
- B) `reader.readLine();`
- C) `characters.read();`
- D) `reader.read(characters);`

```
package com.venky.oracle.pkg14;

import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
```

```
public class Test14 {
    public static void main(String[] args) throws IOException {
  char[] characters =  new char[100];

  try(FileReader reader = new
FileReader("F:\\MyCourse\\repo\\algorithm\\src\\main\\java\\com\\
venky\\oracle\\pkg14\\a.txt"))
  {

     // characters= reader.read();
    //  reader.readLine();
     // characters.read();
      reader.read(characters); // Option D is correct
      System.out.println(String.valueOf(characters));

  }
    }
}
/*
OPTION D  reaqer.read(characters);
 */
```

```
a
b
c
d
e
```

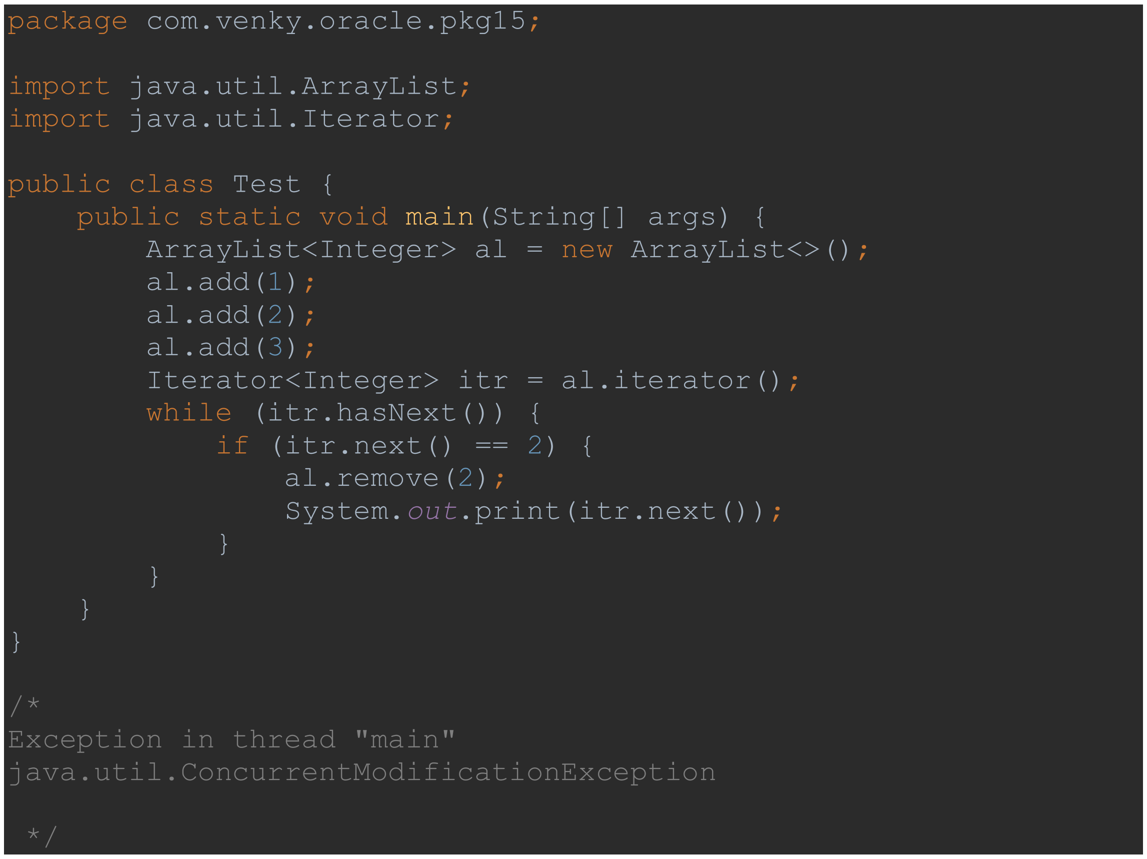
Qus no:- 15



```
ArrayList<Integer> al = new ArrayList<>();
al.add(1);
al.add(2);
al.add(3);
Iterator<Integer> itr = al.iterator();
while (itr.hasNext()) {
    if (itr.next() == 2) {
            al.remove(2);
        System.out.print(itr.next());
    }
}
```

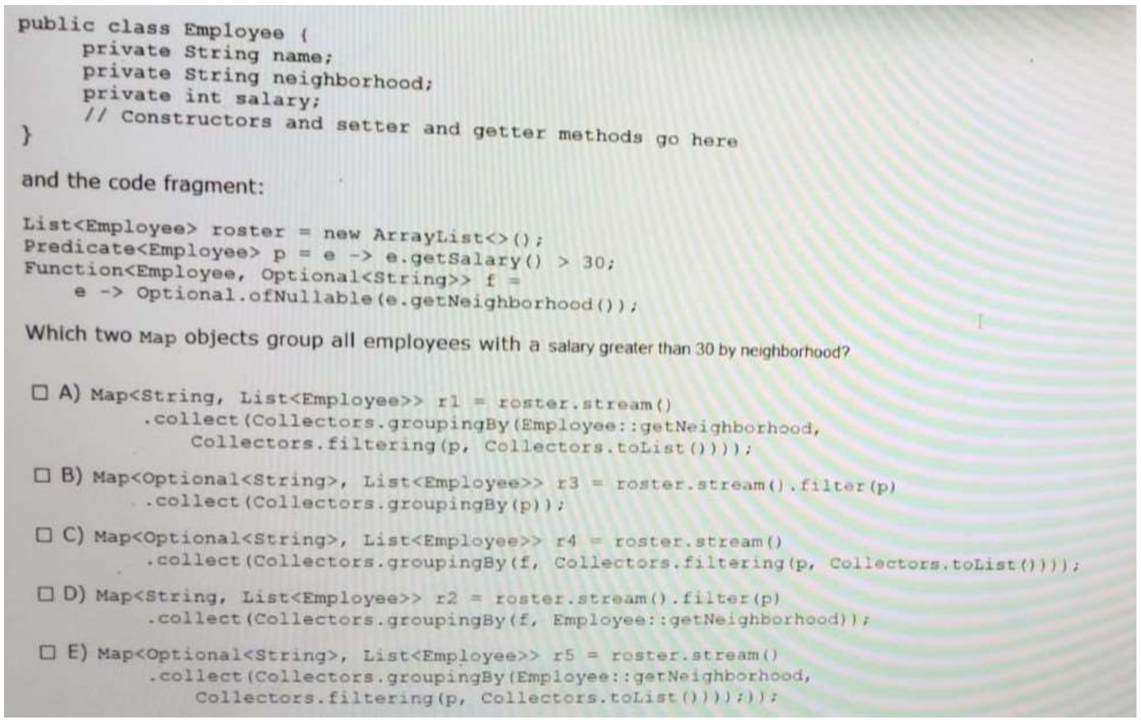
What is the result?

- A) A `ConcurrentModificationException` is thrown at run time.
- B) 1 2 followed by an exception
- C) 1 2 3 followed by an exception
- D) 1 2 4 5



```
package com.venky.oracle.pkg15;

import java.util.ArrayList;
import java.util.Iterator;

public class Test {
    public static void main(String[] args) {
        ArrayList<Integer> al = new ArrayList<>();
        al.add(1);
        al.add(2);
        al.add(3);
        Iterator<Integer> itr = al.iterator();
        while (itr.hasNext()) {
            if (itr.next() == 2) {
                al.remove(2);
                System.out.print(itr.next());
            }
        }
    }
}

/*
Exception in thread "main"
java.util.ConcurrentModificationException

 */
```

Qus no:- 16



```
public class Employee {
    private String name;
    private String neighborhood;
    private int salary;
    // Constructors and setter and getter methods go here
}
```

and the code fragment:

```
List<Employee> roster = new ArrayList<>();
Predicate<Employee> p = e -> e.getSalary() > 30;
Function<Employee, Optional<String>> f =
    e -> Optional.ofNullable(e.getNeighborhood());
```

Which two Map objects group all employees with a salary greater than 30 by neighborhood?

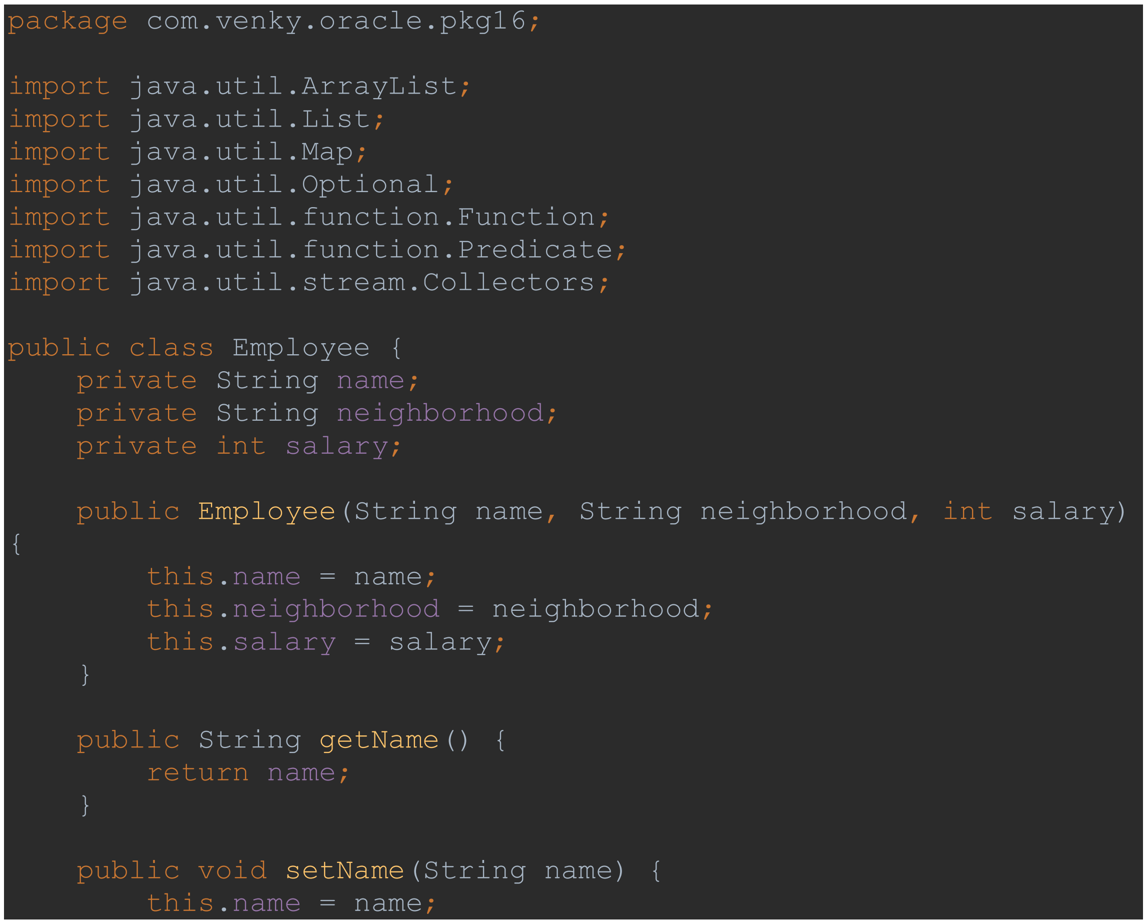
☐ A)
```
Map<String, List<Employee>> r1 = roster.stream()
        .collect(Collectors.groupingBy(Employee::getNeighborhood,
            Collectors.filtering(p, Collectors.toList())));
```

☐ B)
```
Map<Optional<String>, List<Employee>> r3 = roster.stream().filter(p)
        .collect(Collectors.groupingBy(p));
```

☐ C)
```
Map<Optional<String>, List<Employee>> r4 = roster.stream()
        .collect(Collectors.groupingBy(f, Collectors.filtering(p, Collectors.toList())));
```

☐ D)
```
Map<String, List<Employee>> r2 = roster.stream().filter(p)
        .collect(Collectors.groupingBy(f, Employee::getNeighborhood));
```

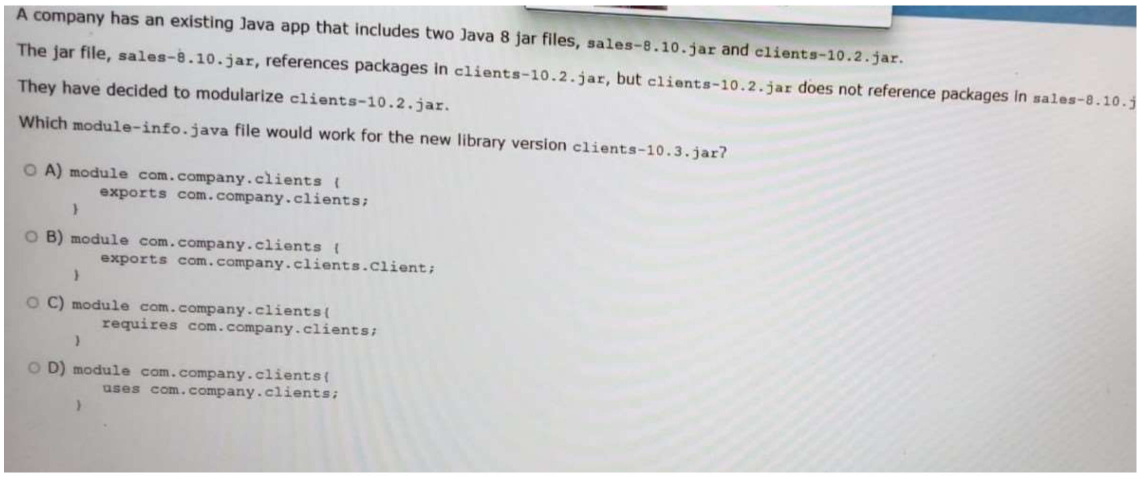
☐ E)
```
Map<Optional<String>, List<Employee>> r5 = roster.stream()
        .collect(Collectors.groupingBy(Employee::getNeighborhood,
            Collectors.filtering(p, Collectors.toList()))););
```



```
package com.venky.oracle.pkg16;

import java.util.ArrayList;
import java.util.List;
import java.util.Map;
import java.util.Optional;
import java.util.function.Function;
import java.util.function.Predicate;
import java.util.stream.Collectors;

public class Employee {
    private String name;
    private String neighborhood;
    private int salary;

    public Employee(String name, String neighborhood, int salary)
{
        this.name = name;
        this.neighborhood = neighborhood;
        this.salary = salary;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getNeighborhood() {
        return neighborhood;
    }

    public void setNeighborhood(String neighborhood) {
        this.neighborhood = neighborhood;
    }

    public int getSalary() {
        return salary;
    }

    public void setSalary(int salary) {
        this.salary = salary;
    }

    @Override
    public String toString() {
        return "Employee{" +
                "name='" + name + '\'' +
                ", neighborhood='" + neighborhood + '\'' +
                ", salary=" + salary +
                '}';
    }

    public static void main(String[] args) {
        List< Employee > roster = new ArrayList< >();
        roster.add(new Employee("John", "Park", 10000));
        roster.add(new Employee("Jane", "Park", 20000));
        roster.add(new Employee("Joe", "Park", 30000));
        roster.add(new Employee("Jack", "Park", 40000));
        roster.add(new Employee("Jill", "Test", 50000));
        roster.add(new Employee("Jana", "Test", 60000));
        roster.add(new Employee("Juan", "tree", 70000));
        Predicate< Employee > p = e -> e.getSalary() > 30000;
        Function <Employee, Optional < String >> f =e ->
Optional.ofNullable(e.getNeighborhood());

        //OPTION A its ture
        Map< String, List < Employee >> r1 = roster.stream()
        .collect(Collectors.groupingBy(Employee::getNeighborhood,
                Collectors.filtering(p, Collectors.toList())));
       // System.out.println(r1);
        //Option B is the compilation error
      //  Map < Optional < String > , List < Employee >> r3 =
roster.stream().filter(p).collect(Collectors.groupingBy(p));
        //Option  C This option is alos true
        Map < Optional < String > , List < Employee >> r4 =
roster.stream().collect(Collectors.groupingBy(f,
Collectors.filtering(p, Collectors.toList())));
       // System.out.println(r4);
```

```
        //Optoin D   Compilation error
/*        Map < String, List < Employee >> r2 =
roster.stream().filter(p)
                .collect(Collectors.groupingBy(f,
Employee::getNeighborhood));*/


        //Option E
  /*      Map < Optional < String > , List < Employee >> r5 =
roster.stream()

.collect(Collectors.groupingBy(Employee::getNeighborhood,
                        Collectors.filtering(p,
Collectors.tolist())))));*/

    }

}


//Solution is A or C is true
```

Qus no:-17



A company has an existing Java app that includes two Java 8 jar files, `sales-8.10.jar` and `clients-10.2.jar`.

The jar file, `sales-8.10.jar`, references packages in `clients-10.2.jar`, but `clients-10.2.jar` does not reference packages in `sales-8.10.j`

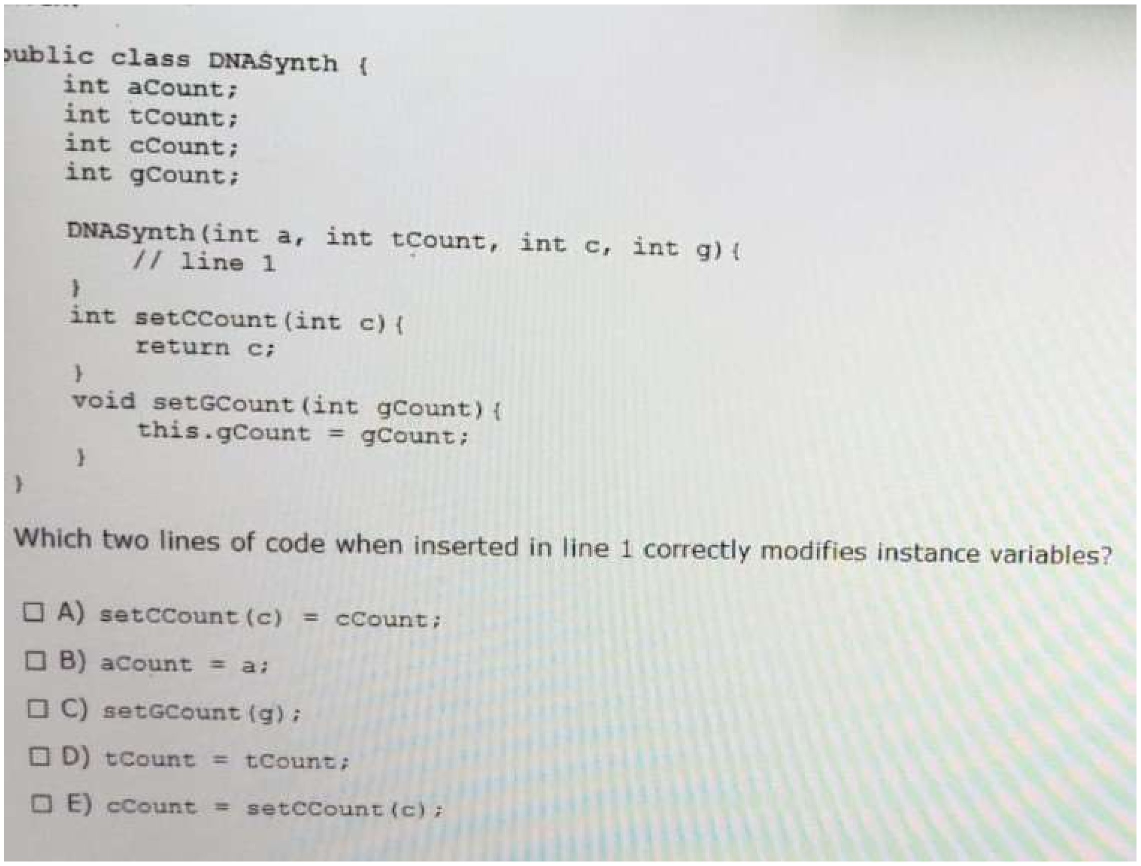
They have decided to modularize `clients-10.2.jar`.

Which `module-info.java` file would work for the new library version `clients-10.3.jar`?

A)
```
module com.company.clients {
    exports com.company.clients;
}
```
B)
```
module com.company.clients {
    exports com.company.clients.Client;
}
```
C)
```
module com.company.clients{
    requires com.company.clients;
}
```
D)
```
module com.company.clients{
    uses com.company.clients;
}
```

```
package com.venky.oracle.pkg17;

public class Main {
    //Learn about java9 will come to this at the end
}
```

Qus no:-18



```
ublic class DNASynth {
    int aCount;
    int tCount;
    int cCount;
    int gCount;

    DNASynth(int a, int tCount, int c, int g){
        // line 1
    }
    int setCCount(int c){
        return c;
    }
    void setGCount(int gCount){
        this.gCount = gCount;
    }
}
```

Which two lines of code when inserted in line 1 correctly modifies instance variables?

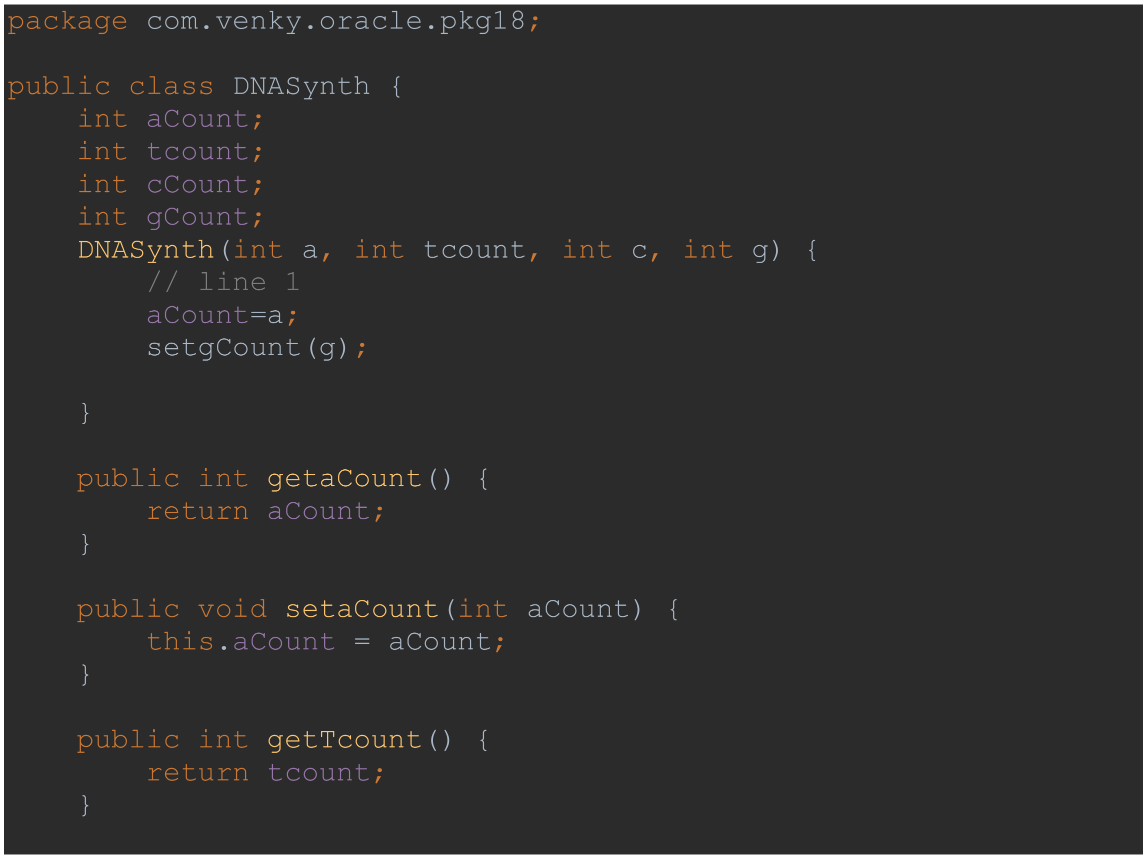
☐ A) `setCCount(c) = cCount;`

☐ B) `aCount = a;`

☐ C) `setGCount(g);`

☐ D) `tCount = tCount;`

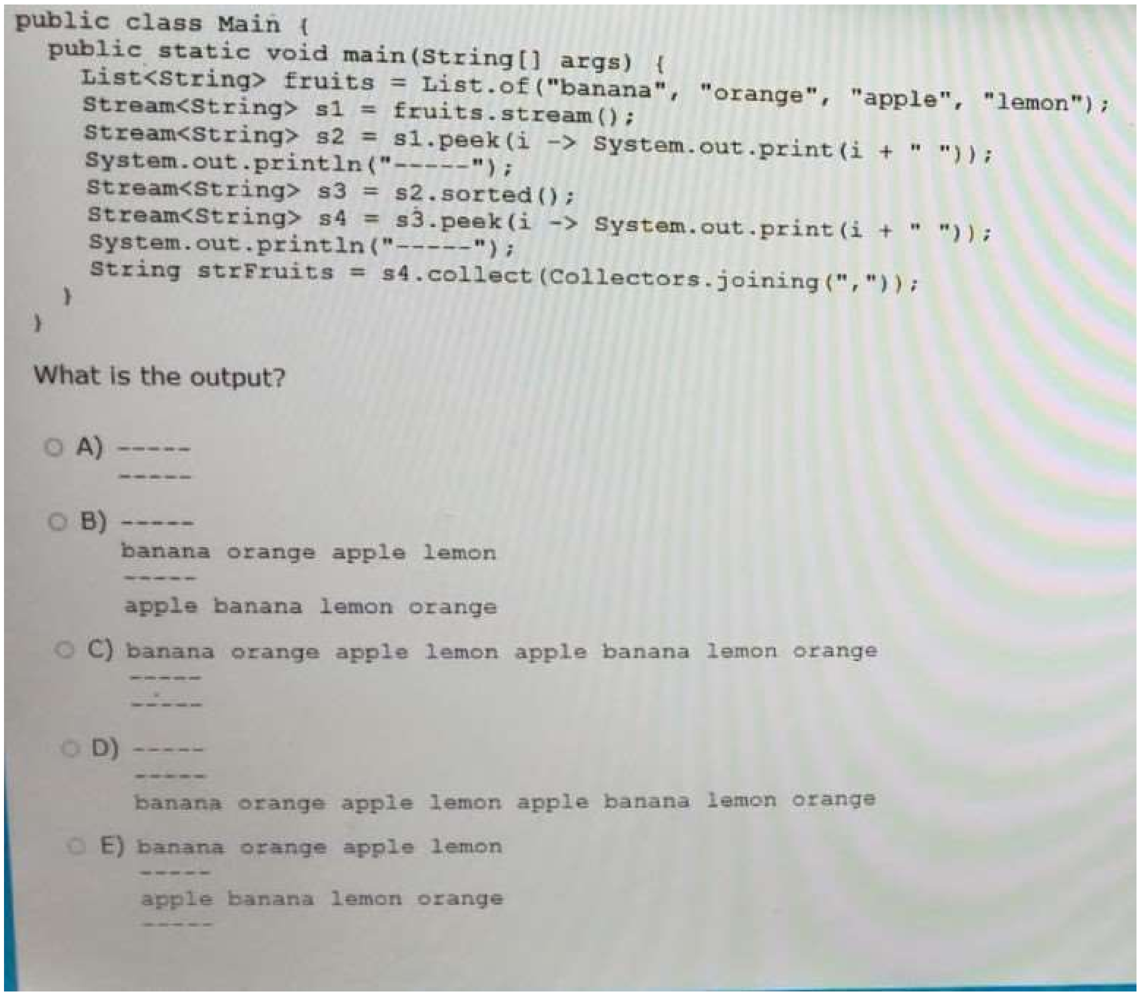
☐ E) `cCount = setCCount(c);`



```
package com.venky.oracle.pkg18;

public class DNASynth {
    int aCount;
    int tcount;
    int cCount;
    int gCount;
    DNASynth(int a, int tcount, int c, int g) {
        // line 1
        aCount=a;
        setgCount(g);

    }

    public int getaCount() {
        return aCount;
    }

    public void setaCount(int aCount) {
        this.aCount = aCount;
    }

    public int getTcount() {
        return tcount;
    }
```

```java
    public void setTcount(int tcount) {
        this.tcount = tcount;
    }

    public int getcCount() {
        return cCount;
    }

    public void setcCount(int cCount) {
        this.cCount = cCount;
    }

    public int getgCount() {
        return gCount;
    }

    public void setgCount(int gCount) {
        this.gCount = gCount;
    }

    public static void main(String[] args) {

        DNASynth dnaSynth = new DNASynth(1,2,3,4);
        System.out.println(dnaSynth.getaCount());
        System.out.println(dnaSynth.getTcount());
        System.out.println(dnaSynth.getcCount());
        System.out.println(dnaSynth.getgCount());
    }
}

/*
Solution is B & C
 */
```

Qus no:- 19



```
public class Main {
  public static void main(String[] args) {
    List<String> fruits = List.of("banana", "orange", "apple", "lemon");
    Stream<String> s1 = fruits.stream();
    Stream<String> s2 = s1.peek(i -> System.out.print(i + " "));
    System.out.println("-----");
    Stream<String> s3 = s2.sorted();
    Stream<String> s4 = s3.peek(i -> System.out.print(i + " "));
    System.out.println("-----");
    String strFruits = s4.collect(Collectors.joining(","));
  }
}
```

What is the output?

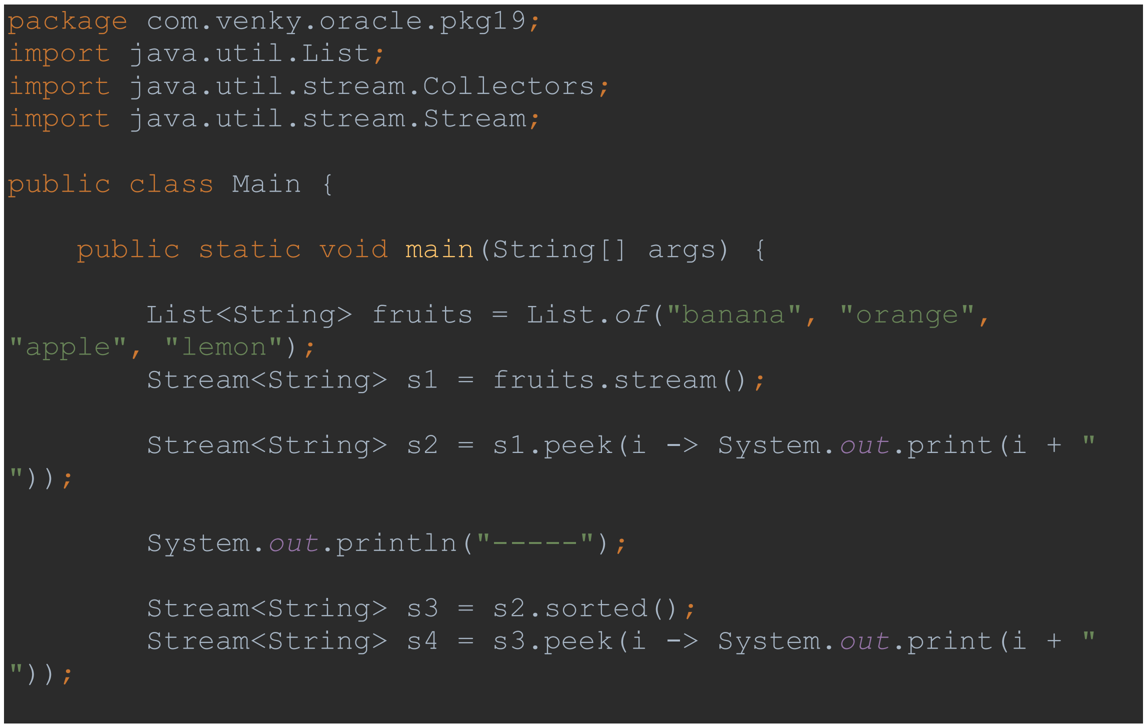
○ A)
```
-----
-----
```

○ B)
```
-----
banana orange apple lemon
-----
apple banana lemon orange
```

○ C)
```
banana orange apple lemon apple banana lemon orange
-----
-----
```

○ D)
```
-----
-----
banana orange apple lemon apple banana lemon orange
```

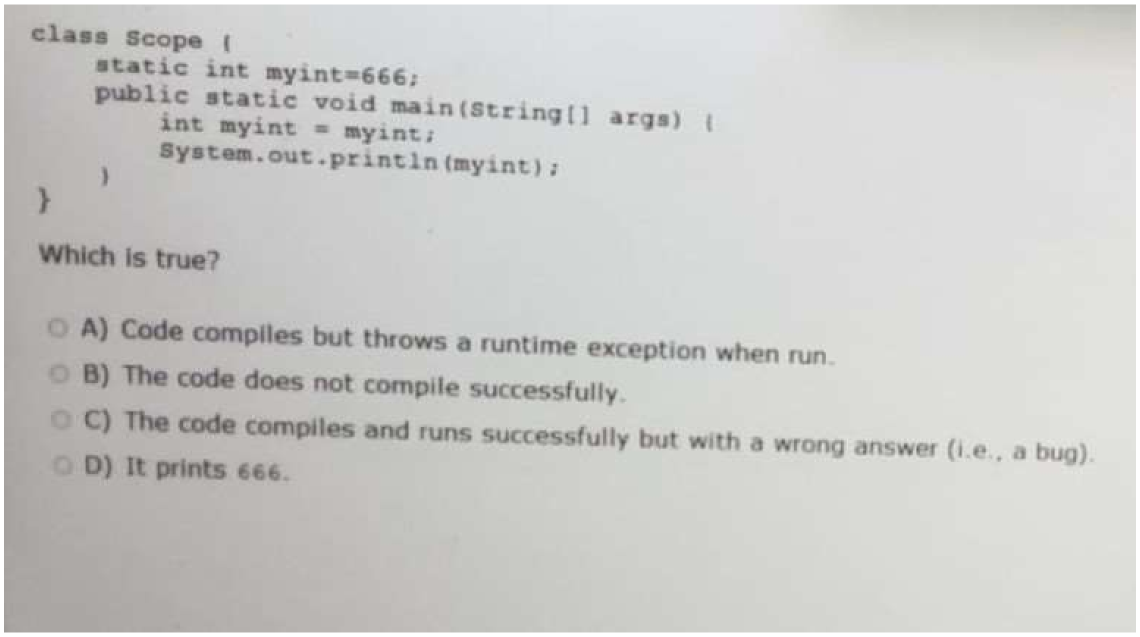
○ E)
```
banana orange apple lemon
-----
apple banana lemon orange
-----
```



```
package com.venky.oracle.pkg19;
import java.util.List;
import java.util.stream.Collectors;
import java.util.stream.Stream;

public class Main {

    public static void main(String[] args) {

        List<String> fruits = List.of("banana", "orange",
"apple", "lemon");
        Stream<String> s1 = fruits.stream();

        Stream<String> s2 = s1.peek(i -> System.out.print(i + "
"));

        System.out.println("-----");

        Stream<String> s3 = s2.sorted();
        Stream<String> s4 = s3.peek(i -> System.out.print(i + "
"));
```

```
        System.out.println("-----");

        String strFruits = s4.collect(Collectors.joining(","));


    }
}
/*
OutPut -- E

-----
-----
banana orange apple lemon apple banana lemon orange
 */
```

Qus no:-20



```
class Scope {
    static int myint=666;
    public static void main(String[] args) {
        int myint = myint;
        System.out.println(myint);
    }
}
```

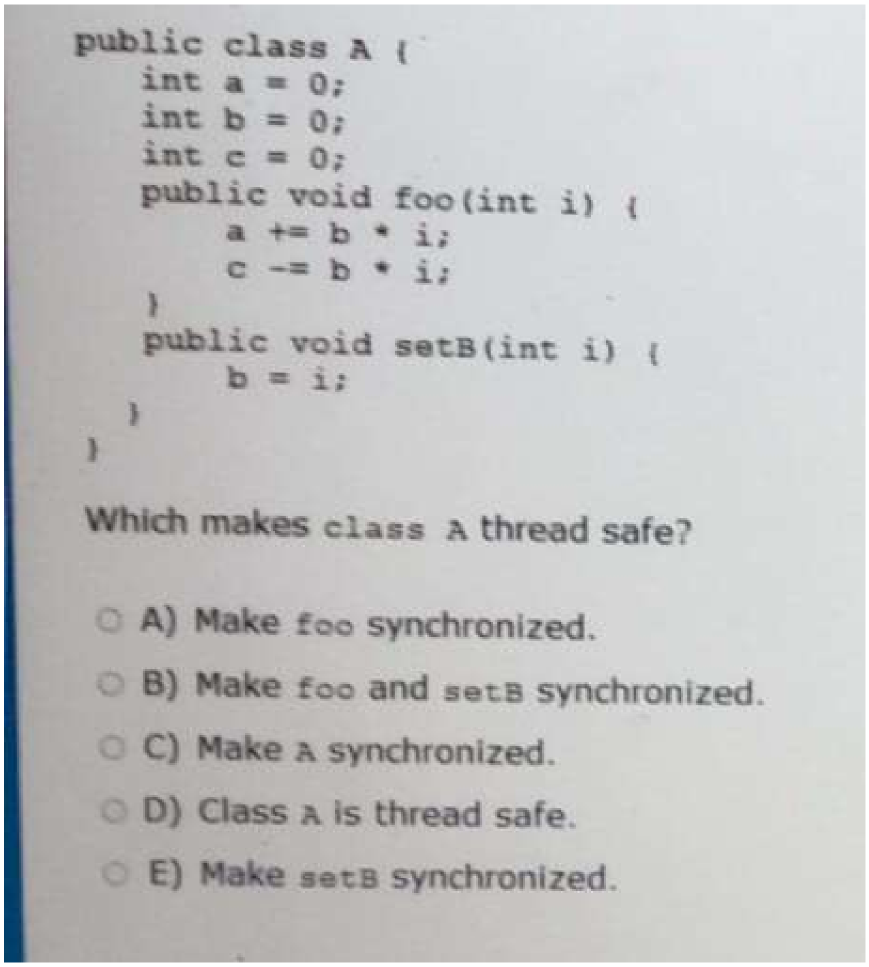
Which is true?

- A) Code compiles but throws a runtime exception when run.
- B) The code does not compile successfully.
- C) The code compiles and runs successfully but with a wrong answer (i.e., a bug).
- D) It prints 666.

```
package com.venky.oracle.pkg20;
/*
class Scope I static int myint 666; public static void
main(String[] args) { System.out.println(myint);

 */
public class Scope {
    static int myint =666;
    public static void main(String[] args) {
//int myint=myint;
        System.out.println(myint);

    }
}
```

```
/*
 Output
Compilation failse at line no 9

 */
```

Qus no:-21



```
public class A {
    int a = 0;
    int b = 0;
    int c = 0;
    public void foo(int i) {
        a += b * i;
        c -= b * i;
    }
    public void setB(int i) {
        b = i;
    }
}
```

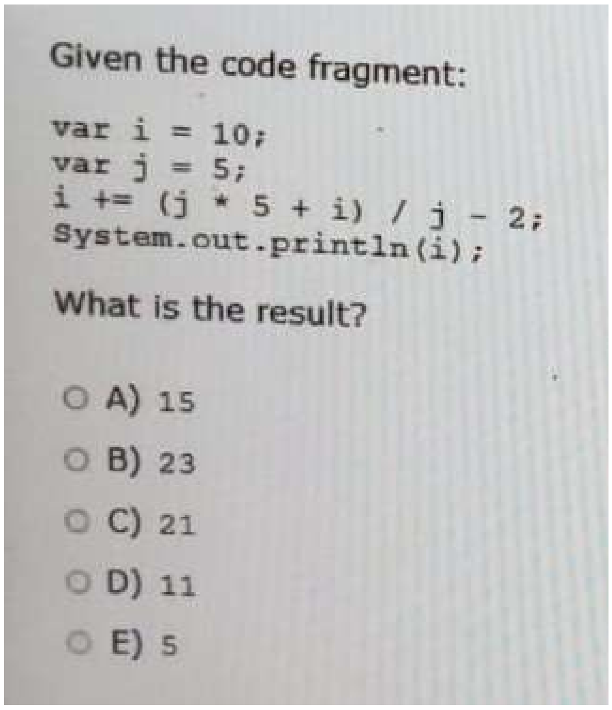
Which makes class A thread safe?

- A) Make foo synchronized.
- B) Make foo and setB synchronized.
- C) Make A synchronized.
- D) Class A is thread safe.
- E) Make setB synchronized.

```
package com.venky.oracle.pkg21;

public   class A {

    int a = 0;
    int b = 0;
    int c = 0;

    public    void foo(int i)  {
        a += b * i;
        c -= b * i;
    }
```

```
  public void setB ( int i){
               b = i;
          }
     }

     /*
     option B make Foo and setB synchronized
      */
```

Qus no:-22



Given the code fragment:

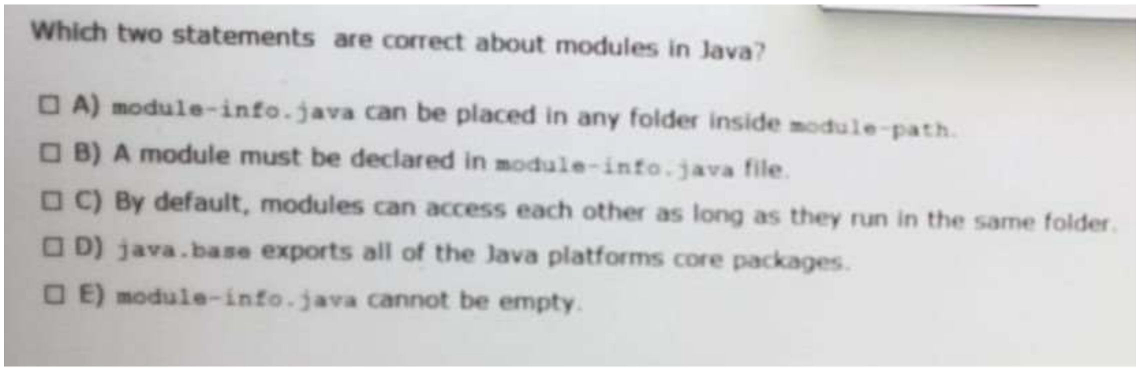
```
var i = 10;
var j = 5;
i += (j * 5 + i) / j - 2;
System.out.println(i);
```

What is the result?

- A) 15
- B) 23
- C) 21
- D) 11
- E) 5

```
package com.venky.oracle.pkg22;

/*
Given the code fragment:
var i = 10; var j = 5; i += (j + 5 + i) Ij - 2;
System.out.println (i);
What is the result?
oooo
O A) 15 o B) 23 o C) 21 o D) 11 o E) 5
 */
public class Test {
    public static void main(String[] args) {
         int i = 10;
         int j = 5;
        i += (j * 5 + i)/j-2; // ITS SIMLY MEANS  I =I+
(EXPRESSION)

    System.out.println (i);
    }
}


/*
Option A is correct

 */
```

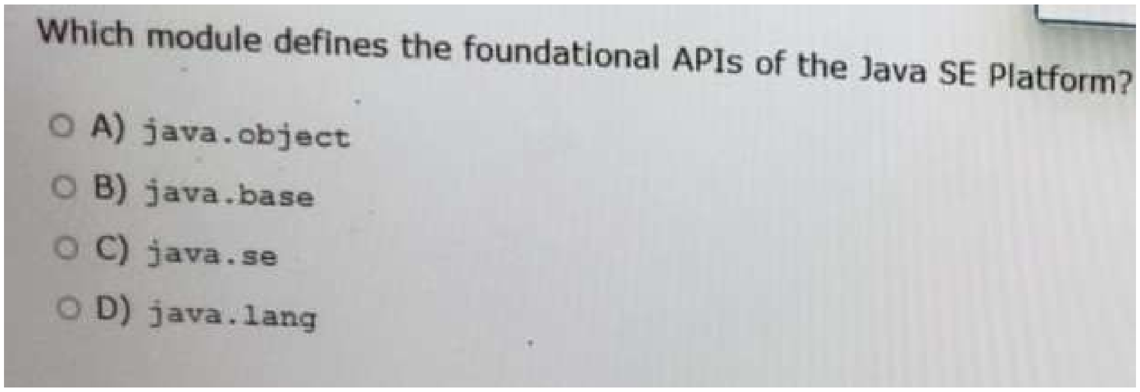
Qus :-23



Which two statements are correct about modules in Java?

- ☐ A) `module-info.java` can be placed in any folder inside `module-path`.
- ☐ B) A module must be declared in `module-info.java` file.
- ☐ C) By default, modules can access each other as long as they run in the same folder.
- ☐ D) `java.base` exports all of the Java platforms core packages.
- ☐ E) `module-info.java` cannot be empty.

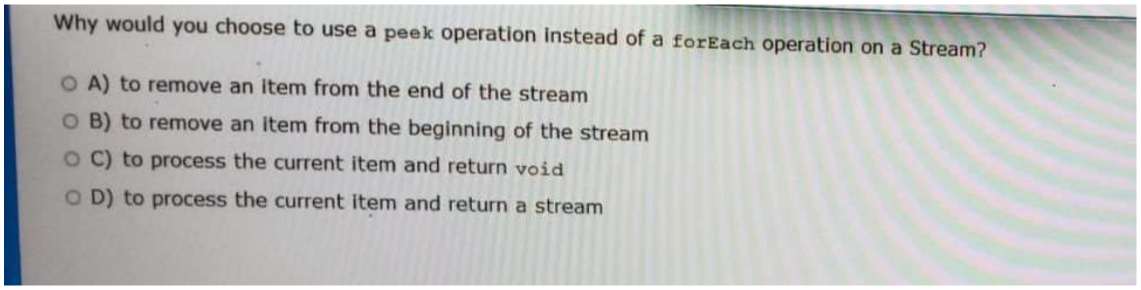
Qus :-24



Which module defines the foundational APIs of the Java SE Platform?

- ○ A) `java.object`
- ○ B) `java.base`
- ○ C) `java.se`
- ○ D) `java.lang`

Qus:- 25

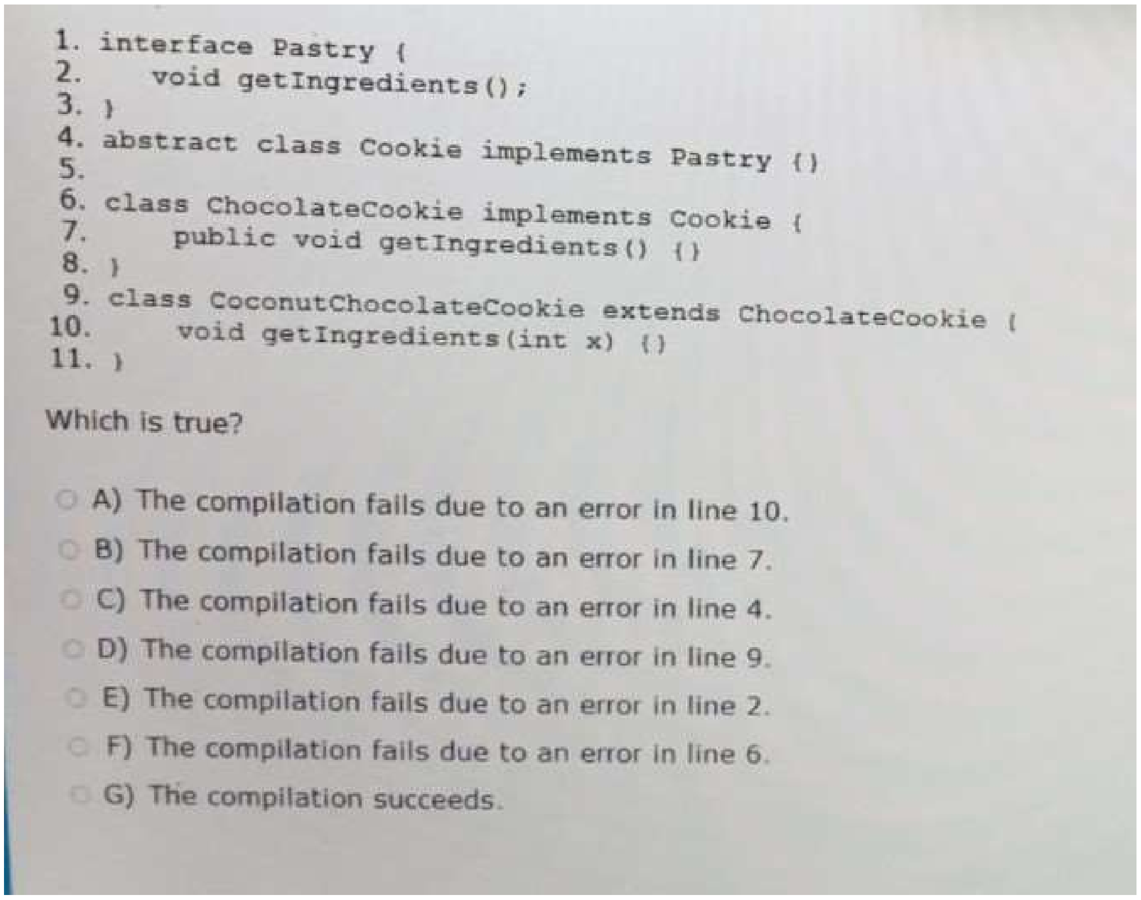


Why would you choose to use a `peek` operation instead of a `forEach` operation on a Stream?

- A) to remove an item from the end of the stream
- B) to remove an item from the beginning of the stream
- C) to process the current item and return `void`
- D) to process the current item and return a stream

```
Option D is correct
```

Qus :-26



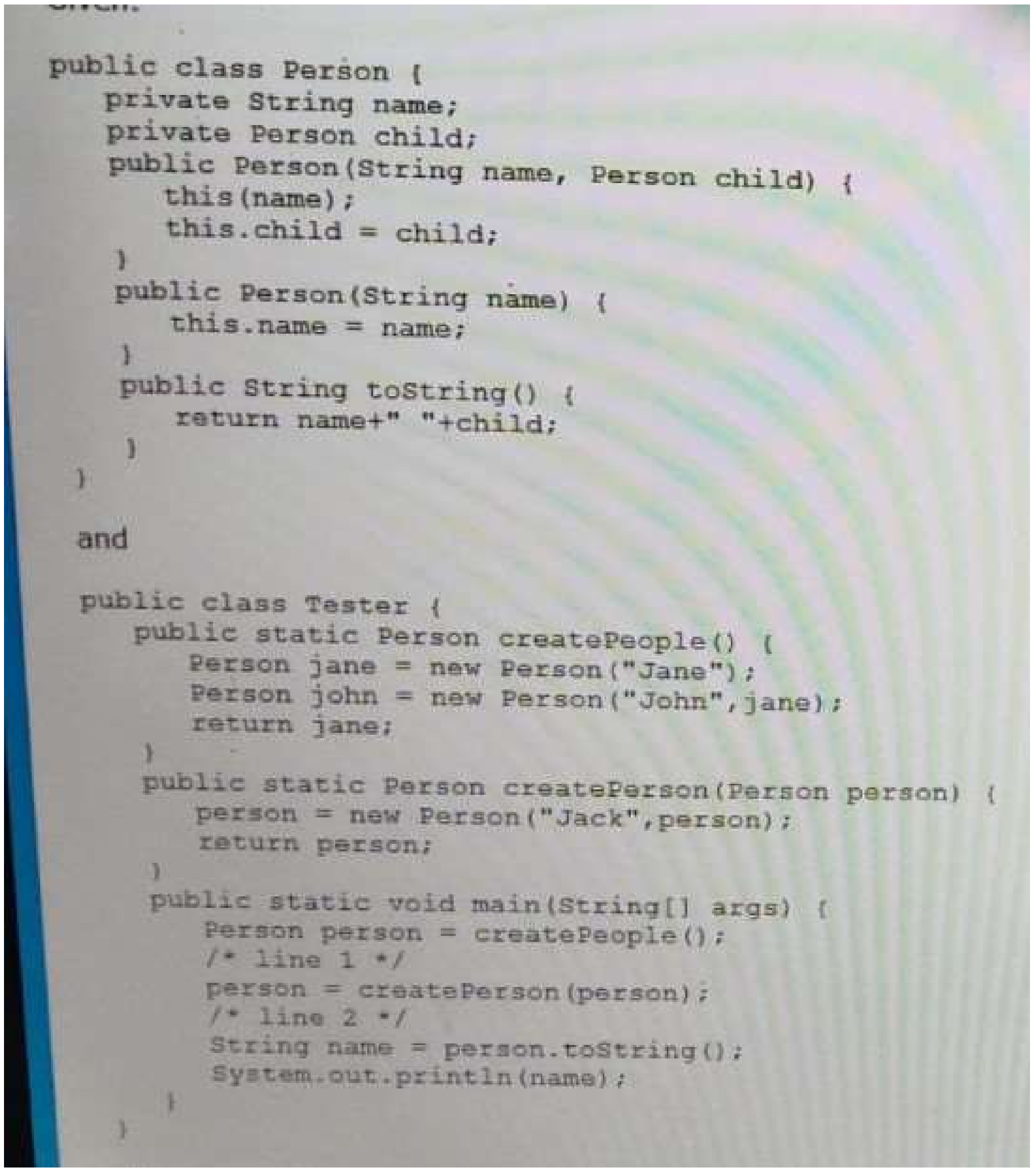
```
1. interface Pastry {
2.     void getIngredients();
3. }
4. abstract class Cookie implements Pastry {}
5.
6. class ChocolateCookie implements Cookie {
7.     public void getIngredients() {}
8. }
9. class CoconutChocolateCookie extends ChocolateCookie {
10.     void getIngredients(int x) {}
11. }
```

Which is true?

- A) The compilation fails due to an error in line 10.
- B) The compilation fails due to an error in line 7.
- C) The compilation fails due to an error in line 4.
- D) The compilation fails due to an error in line 9.
- E) The compilation fails due to an error in line 2.
- F) The compilation fails due to an error in line 6.
- G) The compilation succeeds.

```
package com.venky.oracle.pkg26;
/*
interface Pastry {
    void getIngredients();
}
    abstract class Cookie implements Pastry {
    }
    class ChocolateCookie implements Cookie {
        public void getIngredients(){
        }
    }
 class Coconut ChocolateCookie extends ChocolateCookie
 {
     void get Ingredients(int x) {}
 }

        public class Test {
}
*/

/*
Option  F compilation failes at line no 6
 class ChocolateCookie implements Cookie
 */
```

Qus no :-27

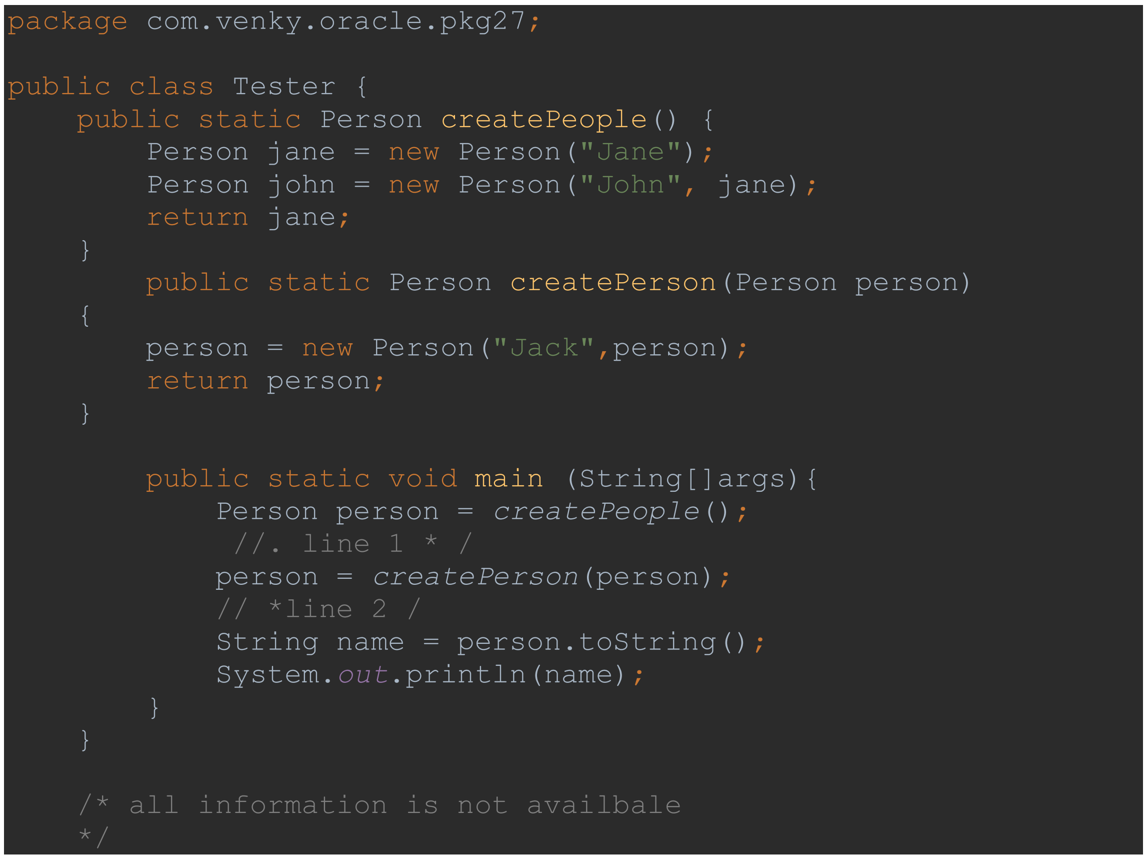


```
public class Person {
    private String name;
    private Person child;
    public Person(String name, Person child) {
        this(name);
        this.child = child;
    }
    public Person(String name) {
        this.name = name;
    }
    public String toString() {
        return name+" "+child;
    }
}
```

and

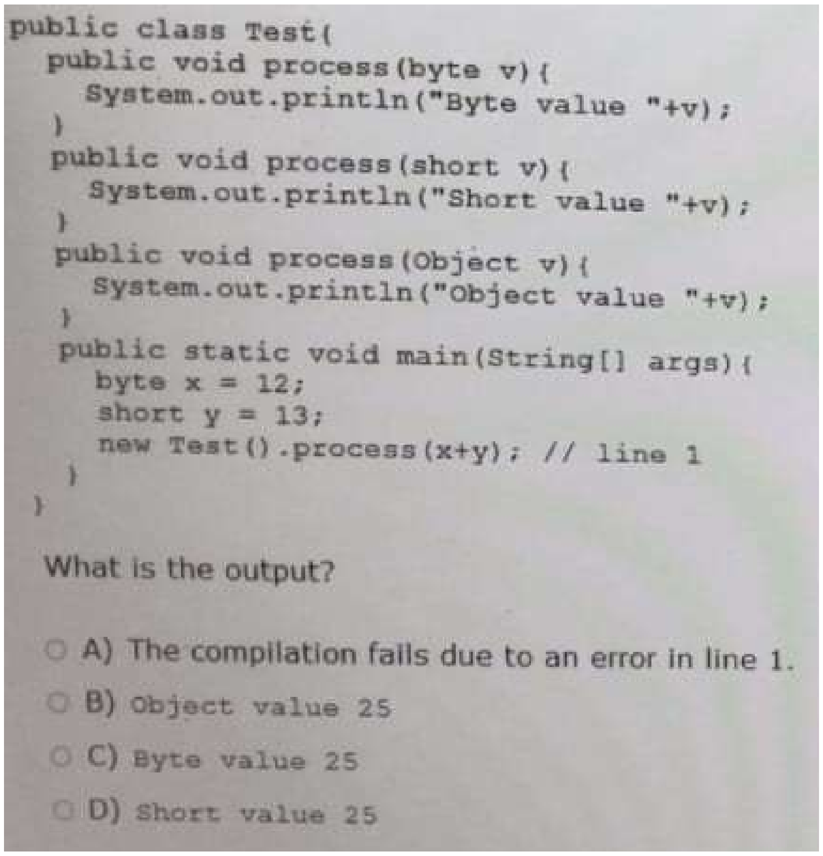
```
public class Tester {
    public static Person createPeople() {
        Person jane = new Person("Jane");
        Person john = new Person("John",jane);
        return jane;
    }
    public static Person createPerson(Person person) {
        person = new Person("Jack",person);
        return person;
    }
    public static void main(String[] args) {
        Person person = createPeople();
        /* line 1 */
        person = createPerson(person);
        /* line 2 */
        String name = person.toString();
        System.out.println(name);
    }
}
```

```
package com.venky.oracle.pkg27;

public class Person
{
    private String name;
    private Person child;
    public Person(String name, Person child) {
        this(name);
        this.child = child;
    }
        public Person(String name) {
            this.name = name;
        }
            public String tostring() {
                return name + " " + child;
            }
}
```



```java
package com.venky.oracle.pkg27;

public class Tester {
    public static Person createPeople() {
        Person jane = new Person("Jane");
        Person john = new Person("John", jane);
        return jane;
    }
        public static Person createPerson(Person person)
    {
        person = new Person("Jack",person);
        return person;
    }

        public static void main (String[]args){
            Person person = createPeople();
             //. line 1 * /
            person = createPerson(person);
            // *line 2 /
            String name = person.toString();
            System.out.println(name);
        }
    }

    /* all information is not availbale
    */
```

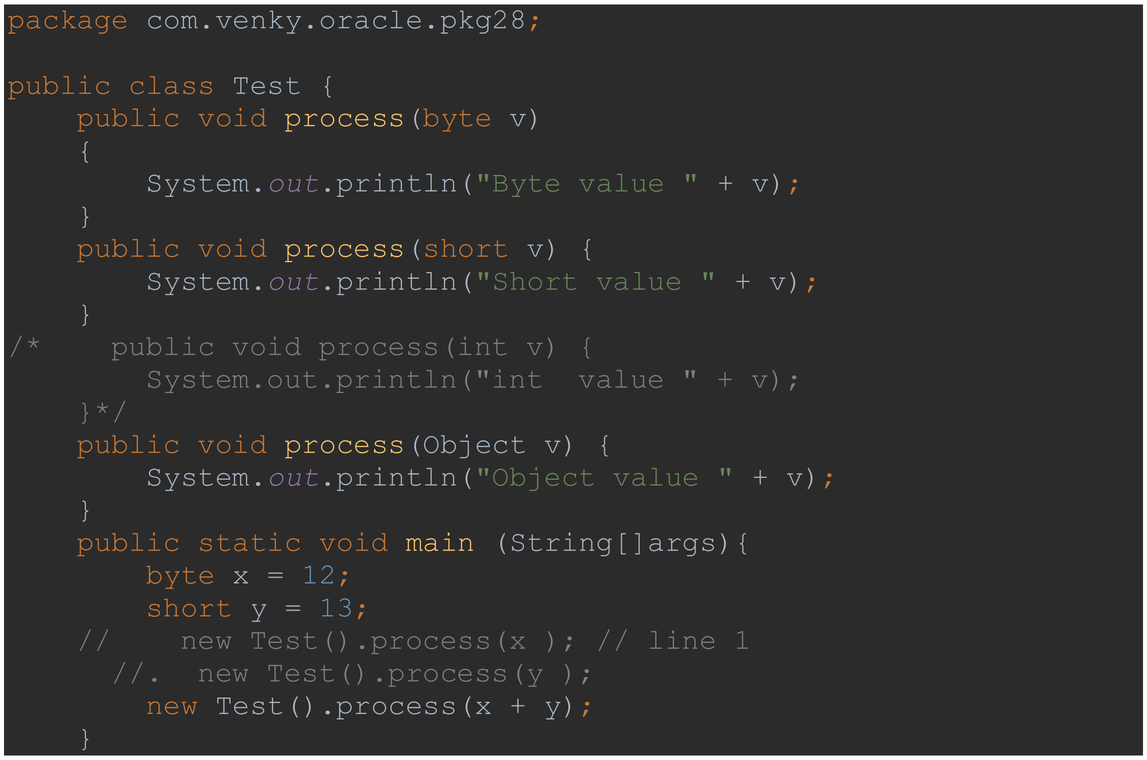
Qus 28:-



```
public class Test{
  public void process(byte v){
    System.out.println("Byte value "+v);
  }
  public void process(short v){
    System.out.println("Short value "+v);
  }
  public void process(Object v){
    System.out.println("Object value "+v);
  }
  public static void main(String[] args){
    byte x = 12;
    short y = 13;
    new Test().process(x+y); // line 1
  }
}
```

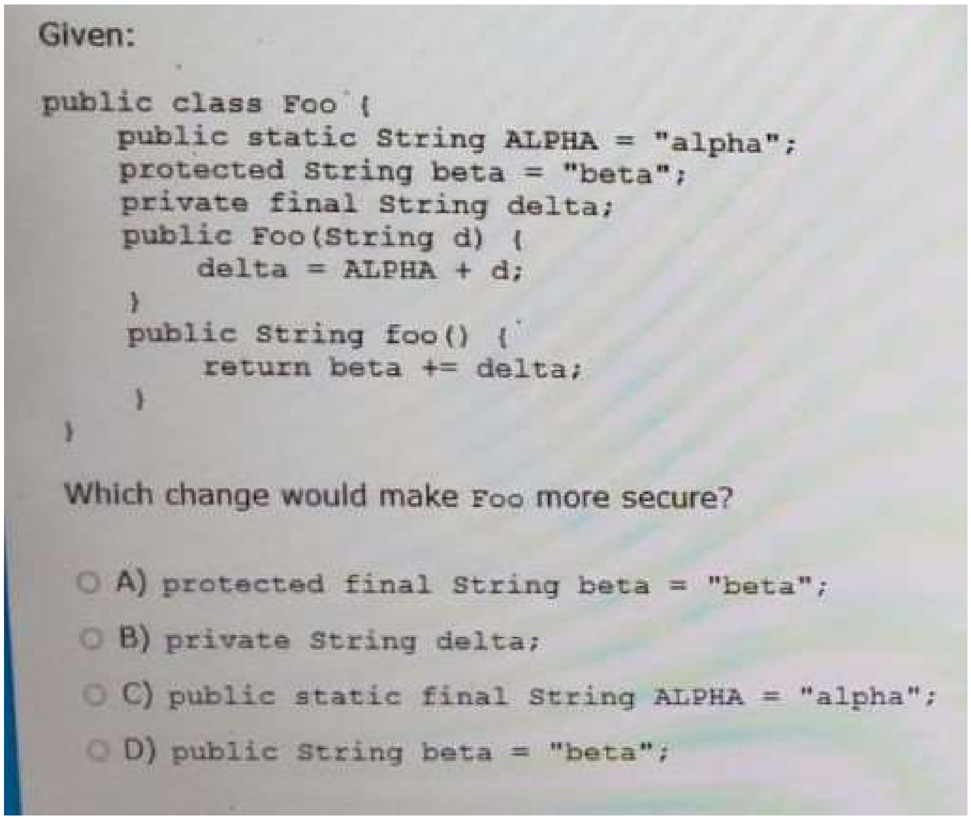
What is the output?

- A) The compilation fails due to an error in line 1.
- B) `Object value 25`
- C) `Byte value 25`
- D) `Short value 25`



```
package com.venky.oracle.pkg28;

public class Test {
    public void process(byte v)
    {
        System.out.println("Byte value " + v);
    }
    public void process(short v)  {
        System.out.println("Short value " + v);
    }
/*    public void process(int v)  {
        System.out.println("int  value " + v);
    }*/
    public void process(Object v)  {
        System.out.println("Object value " + v);
    }
    public static void main (String[]args){
        byte x = 12;
        short y = 13;
    //    new Test().process(x ); // line 1
      //.  new Test().process(y );
        new Test().process(x + y);
    }
}
// Object value 25
```

Qus no 29:-


Given:

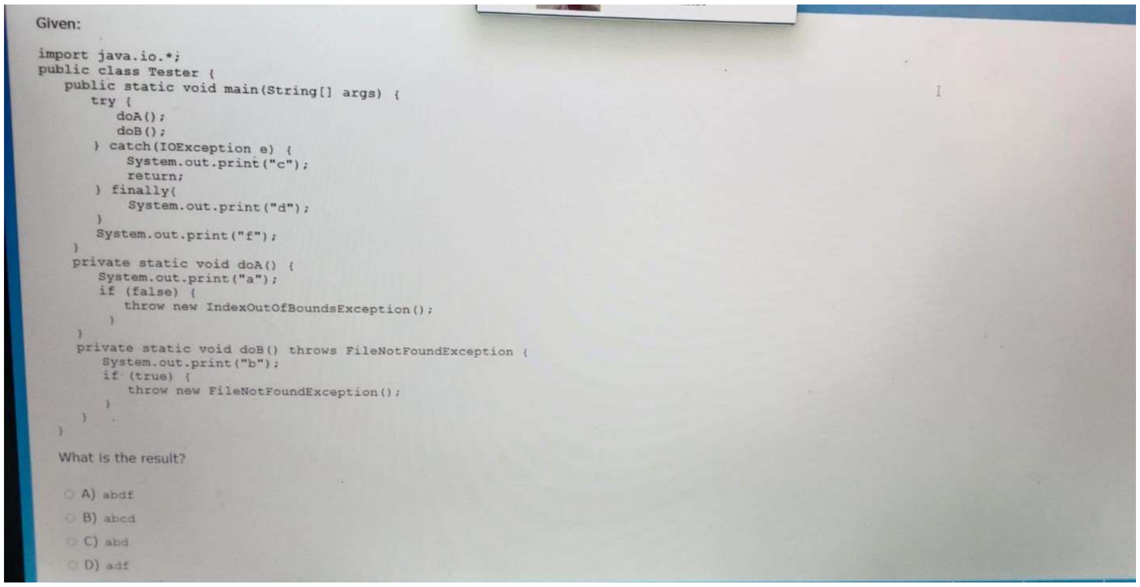
```
public class Foo {
    public static String ALPHA = "alpha";
    protected String beta = "beta";
    private final String delta;
    public Foo(String d) {
        delta = ALPHA + d;
    }
    public String foo() {
        return beta += delta;
    }
}
```

Which change would make `Foo` more secure?

- A) `protected final String beta = "beta";`
- B) `private String delta;`
- C) `public static final String ALPHA = "alpha";`
- D) `public String beta = "beta";`

```
package com.venky.oracle.pkg29;

public class Foo {
    public static String ALPHA = "alpha";
    protected String beta = "beta";
    private final String delta;

    public Foo(String d) {
        delta = ALPHA + d;
    }

    public String foo() {
        return beta += delta;
    }
}

////  C public static finala String  alpha = "Alpha"
```

Qus 30:-



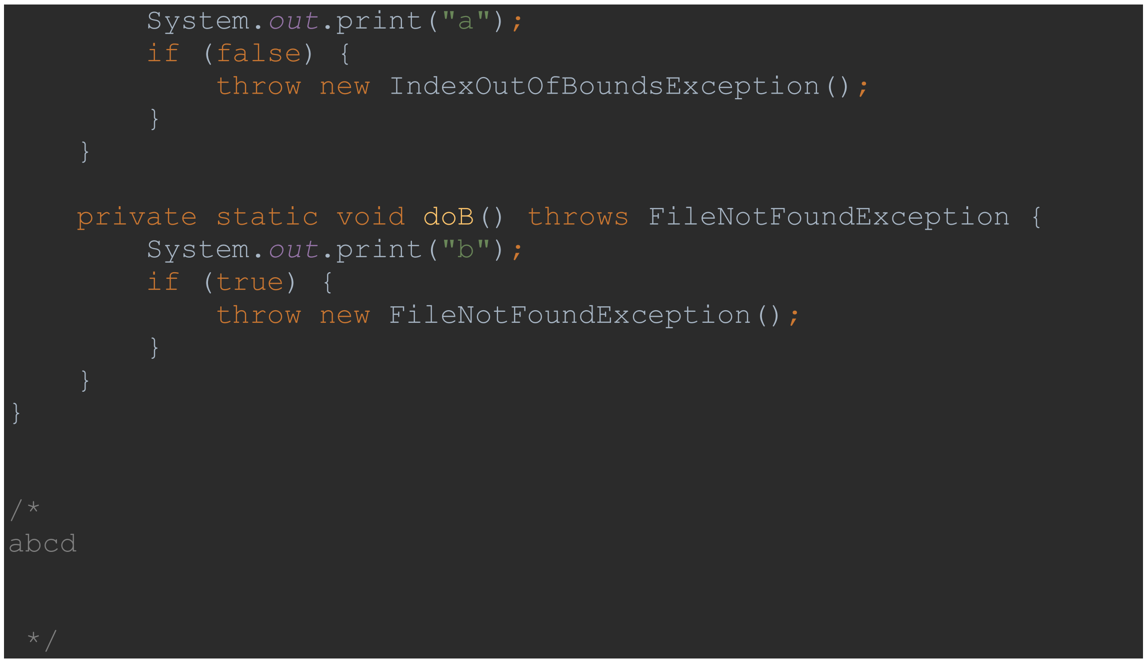
Given:

```
import java.io.*;
public class Tester {
    public static void main(String[] args) {
        try {
            doA();
            doB();
        } catch(IOException e) {
            System.out.print("c");
            return;
        } finally{
            System.out.print("d");
        }
        System.out.print("f");
    }
    private static void doA() {
        System.out.print("a");
        if (false) {
            throw new IndexOutOfBoundsException();
        }
    }
    private static void doB() throws FileNotFoundException {
        System.out.print("b");
        if (true) {
            throw new FileNotFoundException();
        }
    }
}
```

What is the result?

- A) abdf
- B) abcd
- C) abd
- D) adf

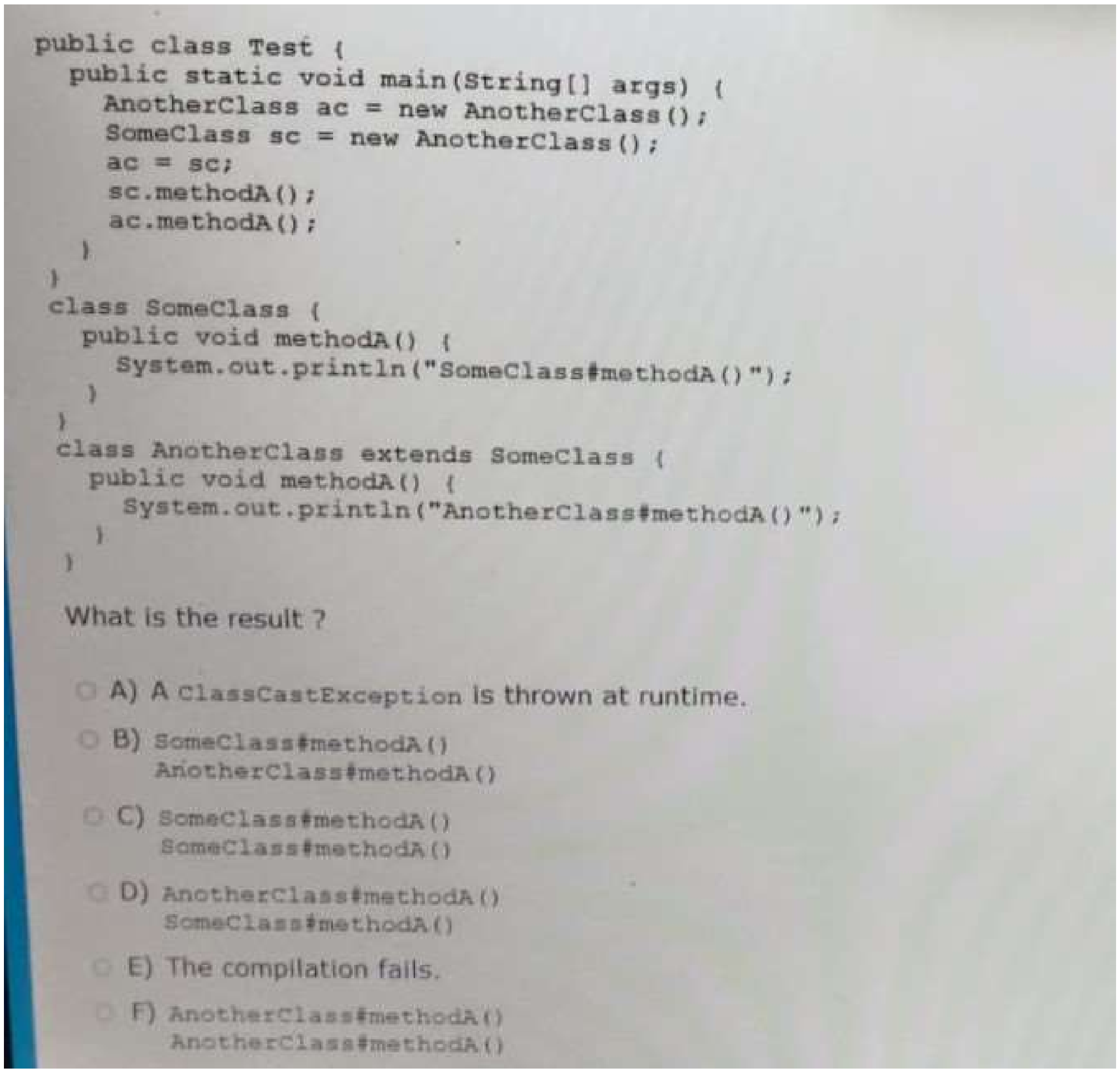
```
package com.venky.oracle.pkg30;

import java.io.*;

public class Tester {
    public static void main(String[] args) {
        try {
            doA();
            doB();
        } catch (IOException e) {
            System.out.print("c");
            return;

        } finally {
            System.out.print("d");
        }
        System.out.print("f"); // This stateMent Wiill naveer
going to be executed
    }

    private static void doA() {
```



```
        System.out.print("a");
        if (false) {
            throw new IndexOutOfBoundsException();
        }
    }

    private static void doB() throws FileNotFoundException {
        System.out.print("b");
        if (true) {
            throw new FileNotFoundException();
        }
    }
}


/*
abcd


 */
```

....

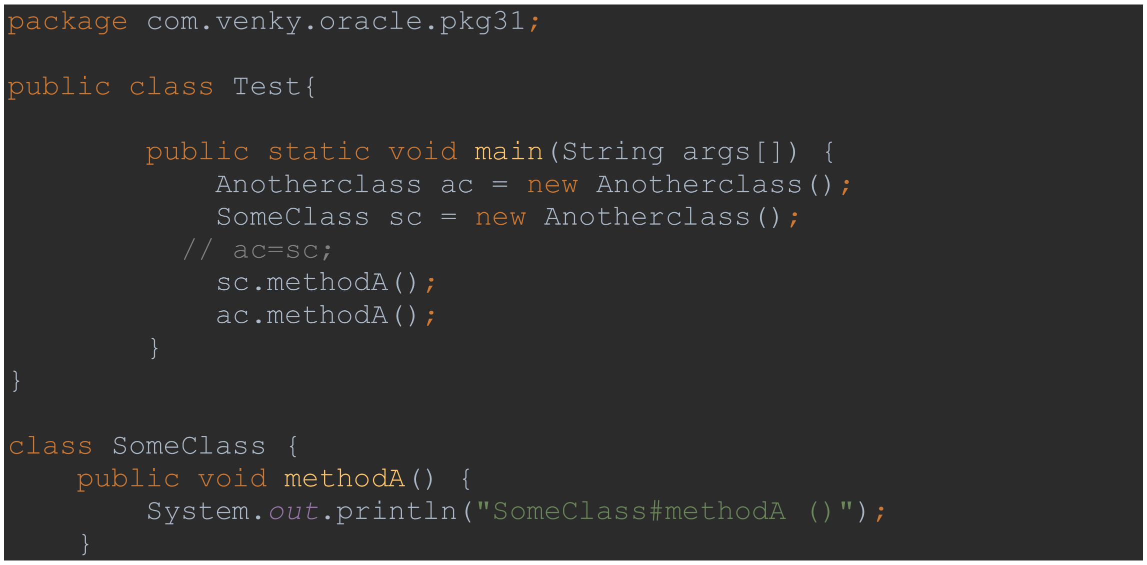
Qus 31:-



```
public class Test {
  public static void main(String[] args) {
    AnotherClass ac = new AnotherClass();
    SomeClass sc = new AnotherClass();
    ac = sc;
    sc.methodA();
    ac.methodA();
  }
}
class SomeClass {
  public void methodA() {
    System.out.println("SomeClass#methodA()");
  }
}
class AnotherClass extends SomeClass {
  public void methodA() {
    System.out.println("AnotherClass#methodA()");
  }
}
```

What is the result ?

- A) A `ClassCastException` is thrown at runtime.
- B) `SomeClass#methodA()`
  `AnotherClass#methodA()`
- C) `SomeClass#methodA()`
  `SomeClass#methodA()`
- D) `AnotherClass#methodA()`
  `SomeClass#methodA()`
- E) The compilation fails.
- F) `AnotherClass#methodA()`
  `AnotherClass#methodA()`



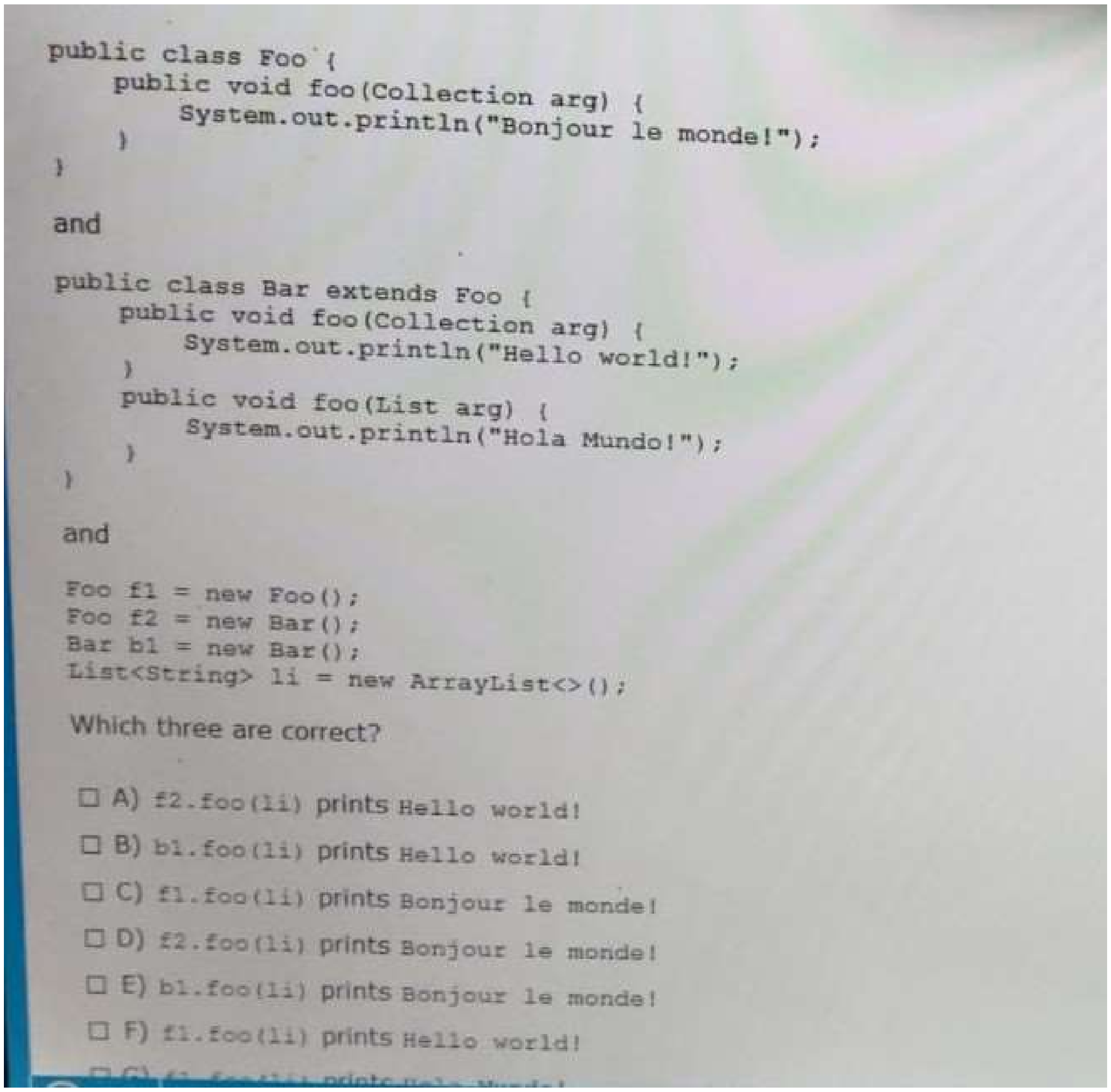
```
package com.venky.oracle.pkg31;

public class Test{

        public static void main(String args[]) {
            Anotherclass ac = new Anotherclass();
            SomeClass sc = new Anotherclass();
          // ac=sc;
            sc.methodA();
            ac.methodA();
        }
}

class SomeClass {
    public void methodA() {
        System.out.println("SomeClass#methodA ()");
    }
}

class Anotherclass extends SomeClass {
    public void methodA() {
        System.out.println("Anotherclass #methodA ()");
        System.out.println("Anotherclass #methodA ()");
    }
}

/*
Compilation fails  at line no 8
 */
```

```
package com.venky.oracle.pkg31;

public class Test{

        public static void main(String args[]) {
            Anotherclass ac = new Anotherclass();
            SomeClass sc = new Anotherclass();
          // ac=sc;
            sc.methodA();
            ac.methodA();
        }
}

class SomeClass {
    public void methodA() {
        System.out.println("SomeClass#methodA ()");
    }
}

class Anotherclass extends SomeClass {
    public void methodA() {
        System.out.println("Anotherclass #methodA ()");
        System.out.println("Anotherclass #methodA ()");
    }
}

/*
Compilation fails  at line no 8
 */
```

```
package com.venky.oracle.pkg31;
```

```java
public class Test{

        public static void main(String args[]) {
            Anotherclass ac = new Anotherclass();
            SomeClass sc = new Anotherclass();
          // ac=sc;
            sc.methodA();
            ac.methodA();
        }
}

class SomeClass {
    public void methodA() {
        System.out.println("SomeClass#methodA ()");
    }
}

class Anotherclass extends SomeClass {
    public void methodA() {
        System.out.println("Anotherclass #methodA ()");
        System.out.println("Anotherclass #methodA ()");
    }
}

/*
Compilation fails  at line no 8
 */
```

Qus no 32:-



```
public class Foo {
    public void foo(Collection arg) {
        System.out.println("Bonjour le monde!");
    }
}
```

and

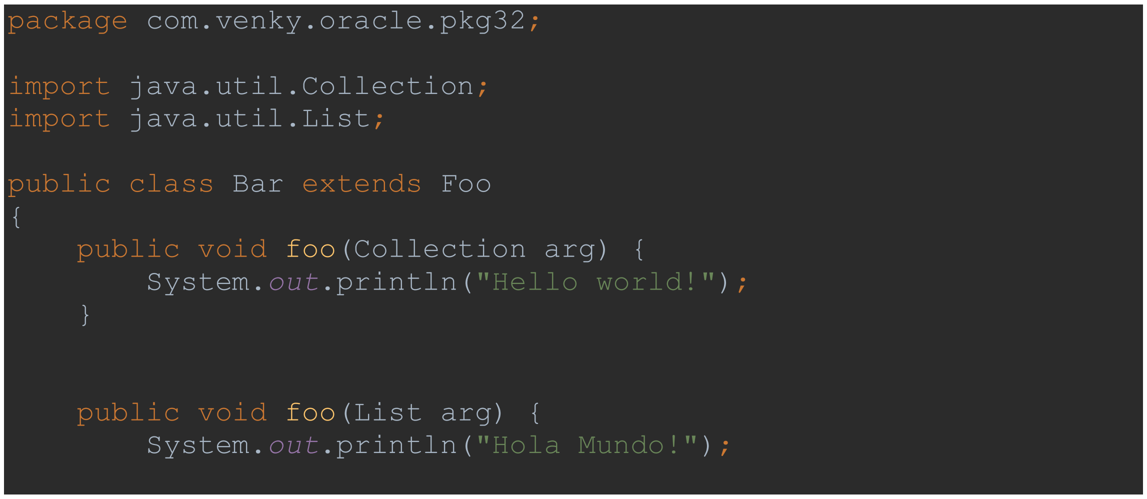
```
public class Bar extends Foo {
    public void foo(Collection arg) {
        System.out.println("Hello world!");
    }
    public void foo(List arg) {
        System.out.println("Hola Mundo!");
    }
}
```

and

```
Foo f1 = new Foo();
Foo f2 = new Bar();
Bar b1 = new Bar();
List<String> li = new ArrayList<>();
```

Which three are correct?

- ☐ A) `f2.foo(li)` prints `Hello world!`
- ☐ B) `b1.foo(li)` prints `Hello world!`
- ☐ C) `f1.foo(li)` prints `Bonjour le monde!`
- ☐ D) `f2.foo(li)` prints `Bonjour le monde!`
- ☐ E) `b1.foo(li)` prints `Bonjour le monde!`
- ☐ F) `f1.foo(li)` prints `Hello world!`
- ☐ G) `[illegible].foo(li)` prints `Hola Mundo!`

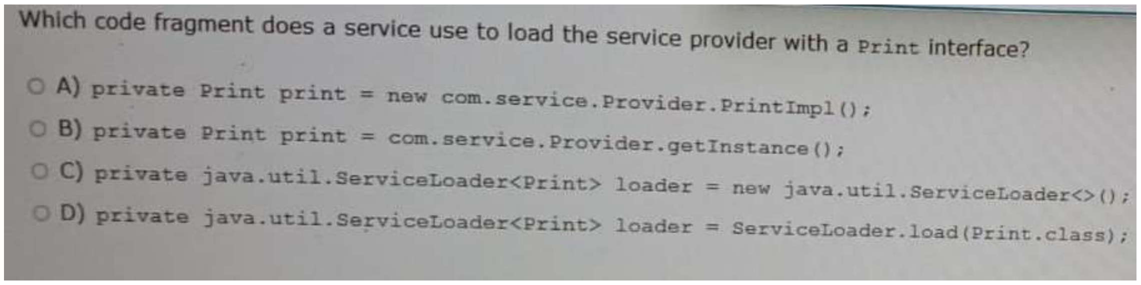


```
package com.venky.oracle.pkg32;

import java.util.Collection;
import java.util.List;

public class Bar extends Foo
{
    public void foo(Collection arg) {
        System.out.println("Hello world!");
    }


    public void foo(List arg) {
        System.out.println("Hola Mundo!");
    }
}
```

```
package com.venky.oracle.pkg32;

import java.util.Collection;

public class Foo
{
    public void foo(Collection arg)
    {
        System.out.println("Bonjour le monde!");
    }
}
```

```
package com.venky.oracle.pkg32;

import java.util.ArrayList;
import java.util.List;

public class Test {
    public static void main(String[] args) {
        Foo f1 = new Foo();
        Foo f2 = new Bar();
        Bar bi = new Bar();
        List< String > li = new ArrayList< >();
        f2.foo(li);// prints Hello world!
        bi.foo(li);//  Hola Mundo
        f1.foo(li);//Bonjour le monde!


    }
}
```

Qus no 33



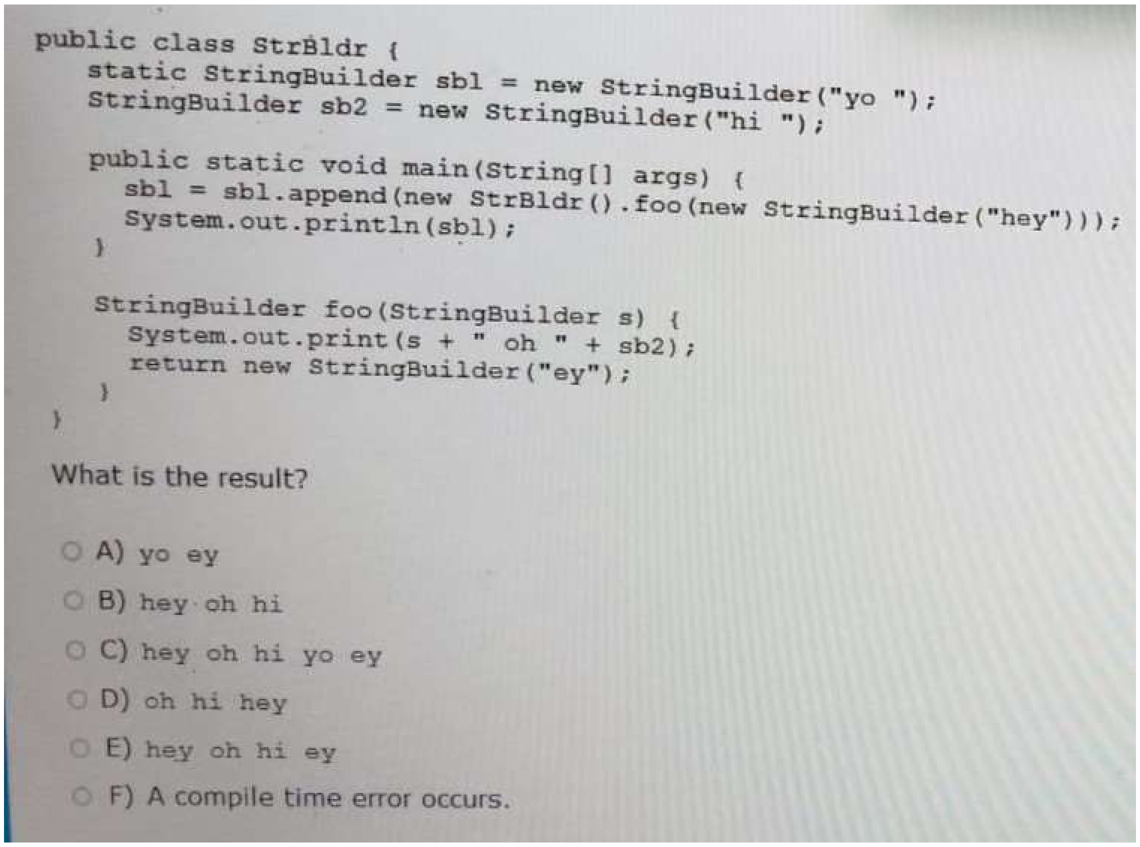
Which code fragment does a service use to load the service provider with a `Print` interface?

- ○ A) `private Print print = new com.service.Provider.PrintImpl();`
- ○ B) `private Print print = com.service.Provider.getInstance();`
- ○ C) `private java.util.ServiceLoader<Print> loader = new java.util.ServiceLoader<>();`
- ○ D) `private java.util.ServiceLoader<Print> loader = ServiceLoader.load(Print.class);`

```java
package com.venky.oracle.pkg33;
interface Print
{}
public class Test {
    public static void main(String[] args) {
      //  private Print print =
com.service.Prov.routingrules.print.Print.getInstance();
        java.util.ServiceLoader<Print> loader =
java.util.ServiceLoader.load(Print.class);
        System.out.println(loader);
    }
}


//Optoin D
//Confusion with opiotn A as well becaue if it talks about the
implementation of the interface
```

```java
package com.venky.oracle.pkg33;
interface Print
{}
public class Test {
    public static void main(String[] args) {
      //  private Print print =
com.service.Prov.routingrules.print.Print.getInstance();
        java.util.ServiceLoader<Print> loader =
java.util.ServiceLoader.load(Print.class);
        System.out.println(loader);
    }
}


//Optoin D
//Confusion with opiotn A as well becaue if it talks about the
implementation of the interface
```

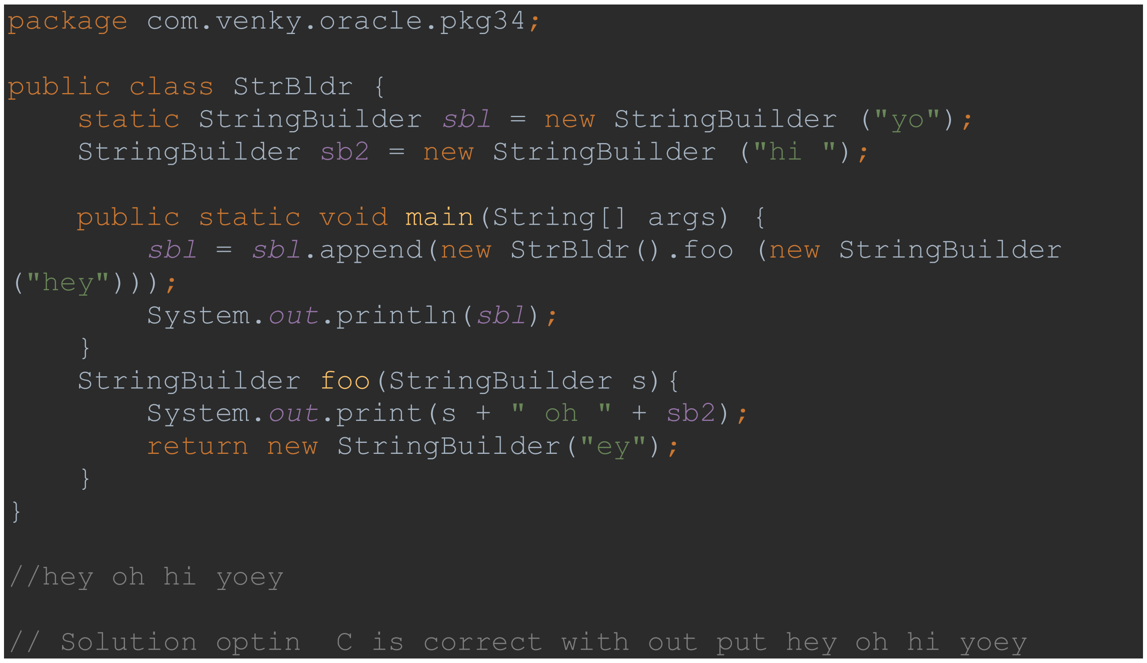
Qus no 34



```
public class StrBldr {
    static StringBuilder sbl = new StringBuilder("yo ");
    StringBuilder sb2 = new StringBuilder("hi ");

    public static void main(String[] args) {
      sbl = sbl.append(new StrBldr().foo(new StringBuilder("hey")));
      System.out.println(sbl);
    }

    StringBuilder foo(StringBuilder s) {
      System.out.print(s + " oh " + sb2);
      return new StringBuilder("ey");
    }
}
```

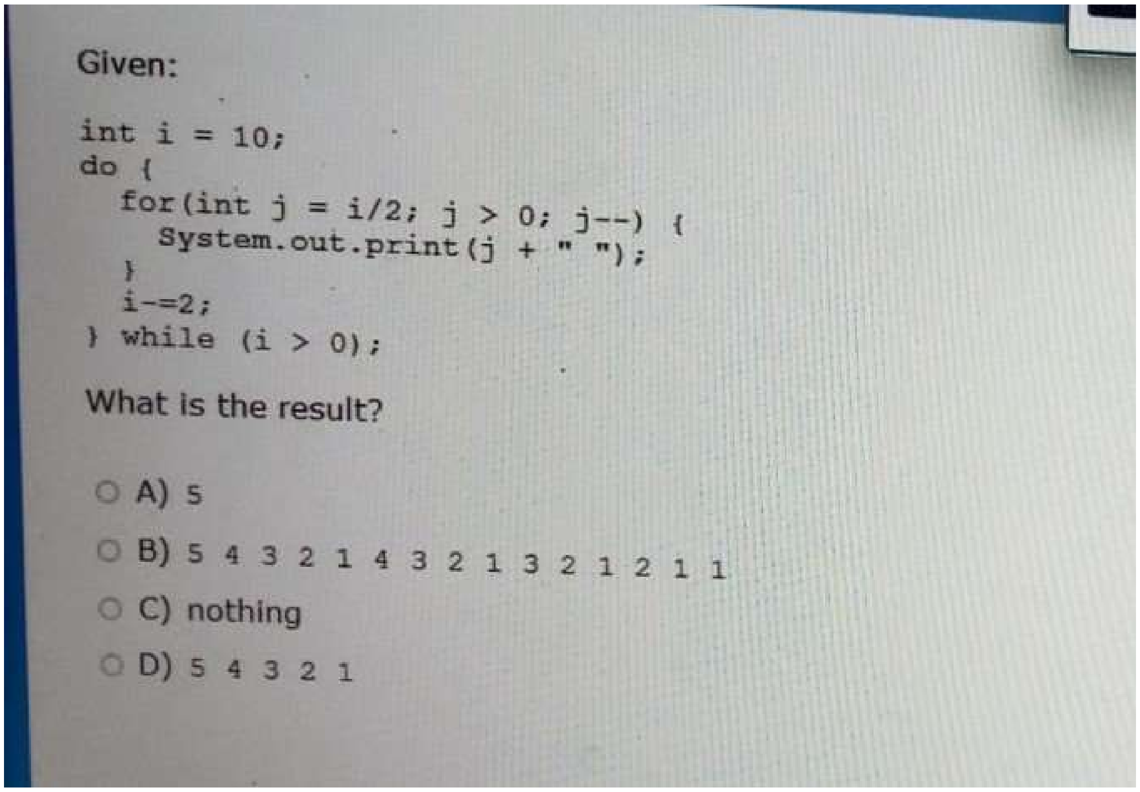
What is the result?

- A) yo ey
- B) hey oh hi
- C) hey oh hi yo ey
- D) oh hi hey
- E) hey oh hi ey
- F) A compile time error occurs.



```
package com.venky.oracle.pkg34;

public class StrBldr {
    static StringBuilder sbl = new StringBuilder ("yo");
    StringBuilder sb2 = new StringBuilder ("hi ");

    public static void main(String[] args) {
        sbl = sbl.append(new StrBldr().foo (new StringBuilder
("hey")));
        System.out.println(sbl);
    }
    StringBuilder foo(StringBuilder s){
        System.out.print(s + " oh " + sb2);
        return new StringBuilder("ey");
    }
}

//hey oh hi yoey

// Solution optin  C is correct with out put hey oh hi yoey
```

Qus no 35

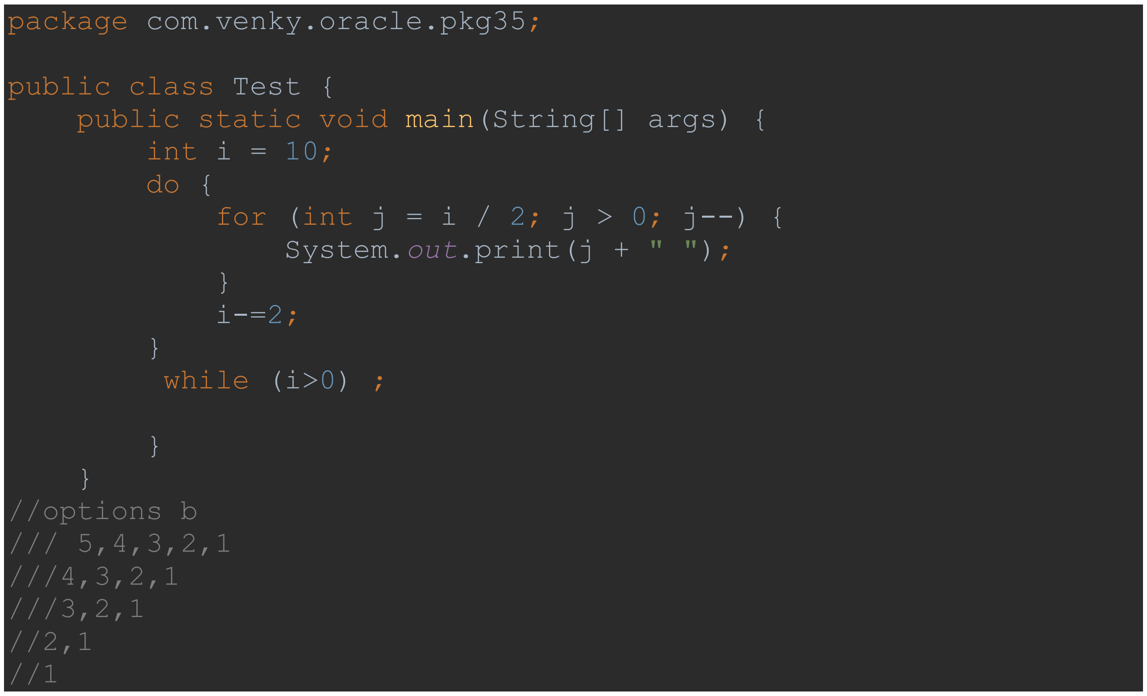


Given:

```
int i = 10;
do {
   for(int j = i/2; j > 0; j--) {
     System.out.print(j + " ");
   }
   i-=2;
} while (i > 0);
```

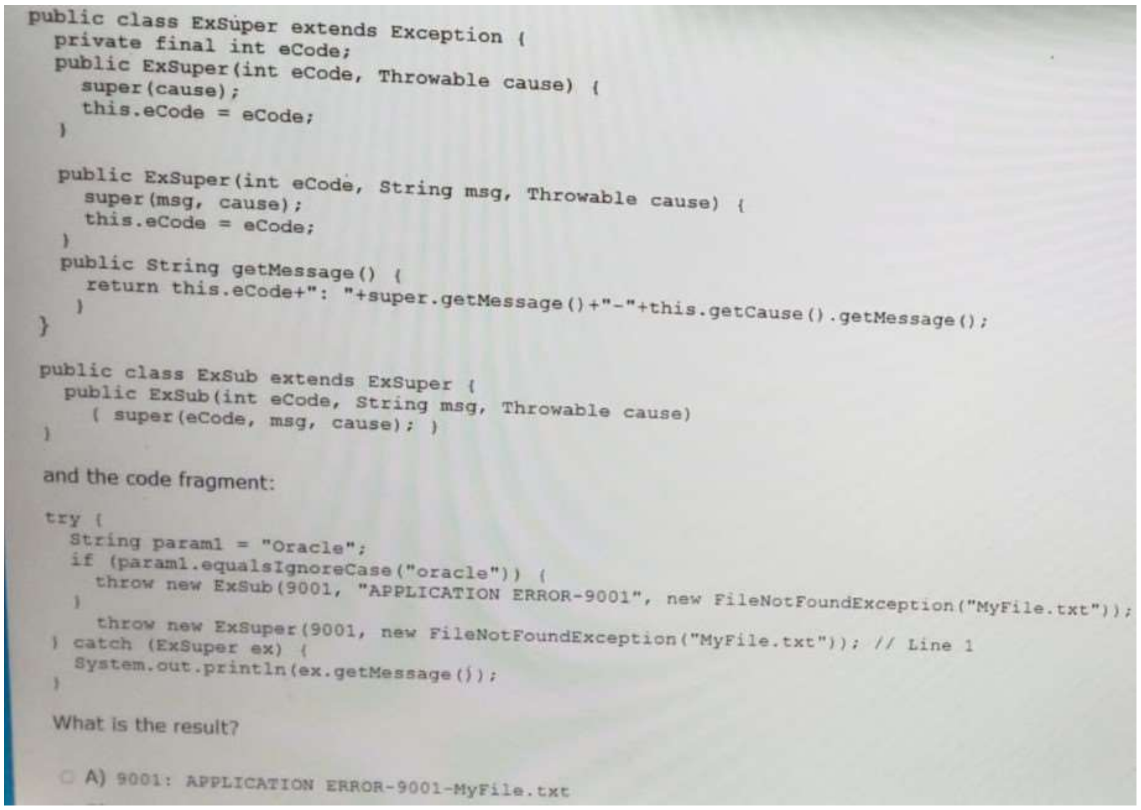
What is the result?

- A) 5
- B) 5 4 3 2 1 4 3 2 1 3 2 1 2 1 1
- C) nothing
- D) 5 4 3 2 1



```
package com.venky.oracle.pkg35;

public class Test {
    public static void main(String[] args) {
        int i = 10;
        do {
            for (int j = i / 2; j > 0; j--) {
                System.out.print(j + " ");
            }
            i-=2;
        }
         while (i>0) ;

        }
    }
//options b
/// 5,4,3,2,1
///4,3,2,1
///3,2,1
//2,1
//1
```

Qus no 36



```
public class ExSuper extends Exception {
  private final int eCode;
  public ExSuper(int eCode, Throwable cause) {
    super(cause);
    this.eCode = eCode;
  }

  public ExSuper(int eCode, String msg, Throwable cause) {
    super(msg, cause);
    this.eCode = eCode;
  }
  public String getMessage() {
    return this.eCode+": "+super.getMessage()+"-"+this.getCause().getMessage();
  }
}

public class ExSub extends ExSuper {
  public ExSub(int eCode, String msg, Throwable cause)
    { super(eCode, msg, cause); }
}
```

and the code fragment:

```
try {
  String param1 = "Oracle";
  if (param1.equalsIgnoreCase("oracle")) {
    throw new ExSub(9001, "APPLICATION ERROR-9001", new FileNotFoundException("MyFile.txt"));
  }
    throw new ExSuper(9001, new FileNotFoundException("MyFile.txt")); // Line 1
} catch (ExSuper ex) {
  System.out.println(ex.getMessage());
}
```

What is the result?

○ A) `9001: APPLICATION ERROR-9001-MyFile.txt`

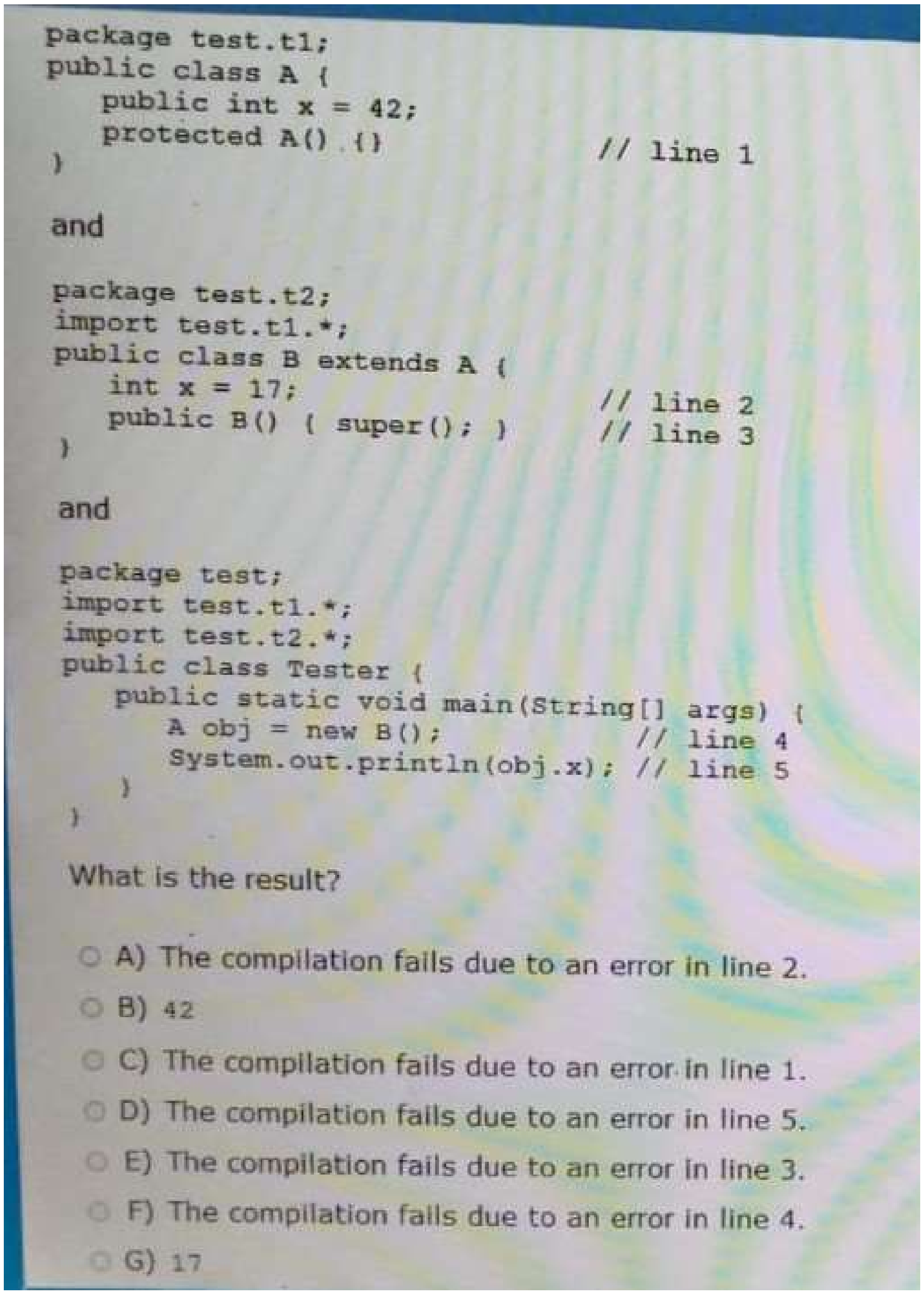
```
package com.venky.oracle.pkg36;

public class ExSub extends ExSuper
{
    public ExSub(int eCode, String msg, Throwable cause){
        super(eCode, msg, cause);
    }

}
```

```
package com.venky.oracle.pkg36;

public class ExSuper extends Exception
{
    private final int eCode;
        public ExSuper(int eCode, Throwable cause)
        {
            super(cause);
            this.eCode = eCode;
        }
public ExSuper(int eCode, String msg, Throwable cause) {
    super(msg, cause);
    this.eCode = eCode;
}
public String getMessage() {
    return this.eCode + ": " + super.getMessage() + "-" +
this.getCause().getMessage();
}

}
```

```
package com.venky.oracle.pkg36;

import java.io.FileNotFoundException;

public class Test {
    public static void main(String[] args) {
        try {
            String param1 = "Oracle";
            if (param1.equalsIgnoreCase("oracle")) {
                throw new ExSub(9001, "APPLICATION ERROR-9001",
new FileNotFoundException("MyFile.txt"));
            }
            throw new ExSuper(9001, new
FileNotFoundException("MyFile.txt")); // Line 1 } catch (ExSuper
ex) {

        }catch (ExSuper ex)
        {
            System.out.println(ex.getMessage());
        }
    }
}
// Soution for this ==>>>>9001: APPLICATION ERROR-9001-MyFile.txt
```

....

Qus no 37



```
package test.t1;
public class A {
    public int x = 42;
    protected A() {}                    // line 1
}
```

and

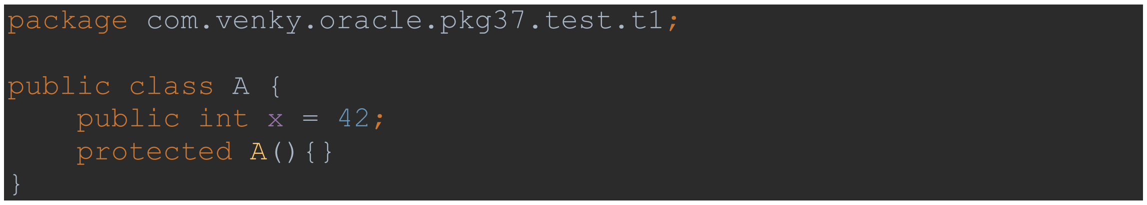
```
package test.t2;
import test.t1.*;
public class B extends A {
    int x = 17;                         // line 2
    public B() { super(); }             // line 3
}
```

and

```
package test;
import test.t1.*;
import test.t2.*;
public class Tester {
    public static void main(String[] args) {
        A obj = new B();                // line 4
        System.out.println(obj.x);      // line 5
    }
}
```

What is the result?

- A) The compilation fails due to an error in line 2.
- B) 42
- C) The compilation fails due to an error in line 1.
- D) The compilation fails due to an error in line 5.
- E) The compilation fails due to an error in line 3.
- F) The compilation fails due to an error in line 4.
- G) 17

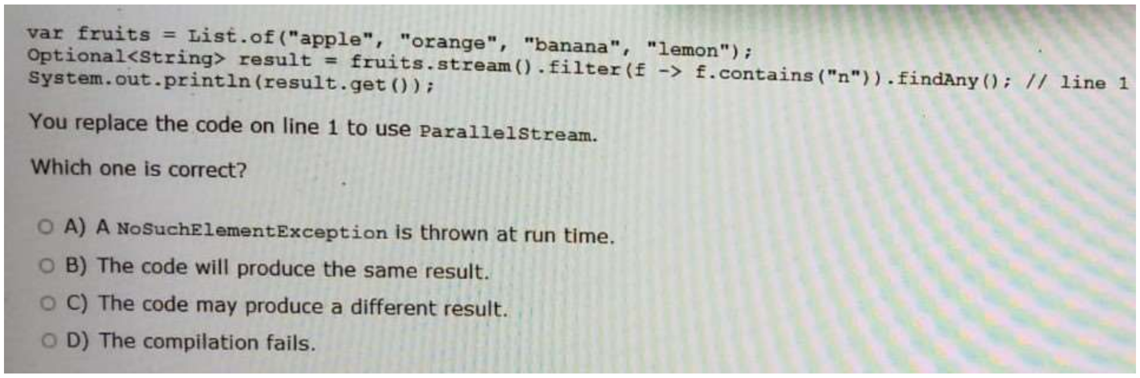


```
package com.venky.oracle.pkg37.test.t1;

public class A {
    public int x = 42;
    protected A(){}
}
```

```java
package com.venky.oracle.pkg37.test.t2;
import com.venky.oracle.pkg37.test.t1.A;
public class B extends A {
    int x = 17;
    public B() {
        super();
    }
}
```

```java
package com.venky.oracle.pkg37.test;

import com.venky.oracle.pkg37.test.t1.A;
import com.venky.oracle.pkg37.test.t2.B;

public class Tester {
    public static void main(String[] args) {
        A obj = new B();
        System.out.println(obj.x);
    }
}

/*
OPTION B is correct answer. with 42

 */
```

Qus no 38



```java
var fruits = List.of("apple", "orange", "banana", "lemon");
Optional<String> result = fruits.stream().filter(f -> f.contains("n")).findAny(); // line 1
System.out.println(result.get());
```

You replace the code on line 1 to use `ParallelStream`.

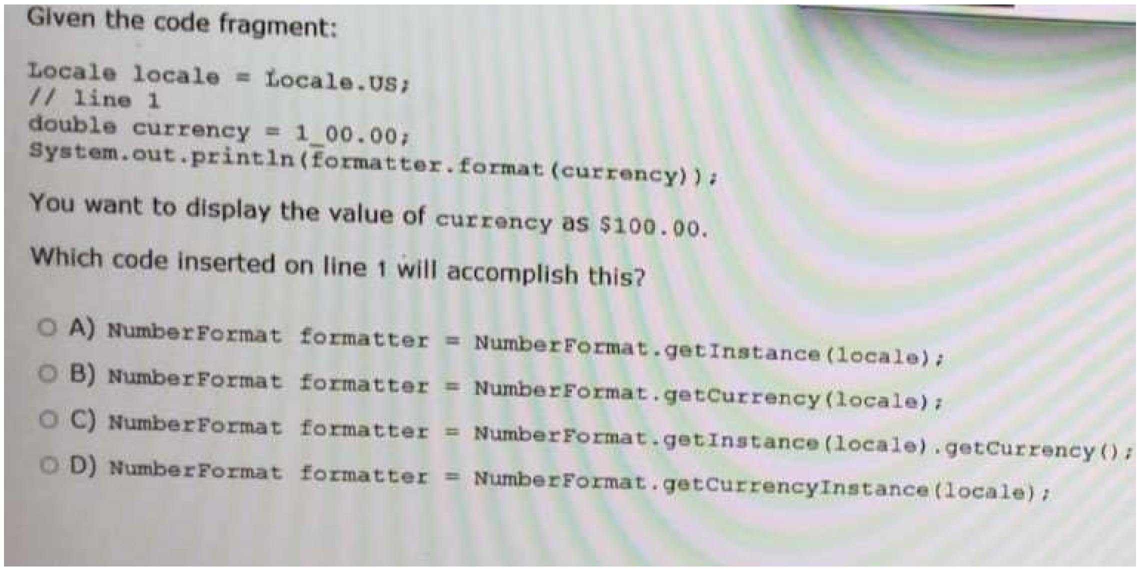
Which one is correct?

- A) A `NoSuchElementException` is thrown at run time.
- B) The code will produce the same result.
- C) The code may produce a different result.
- D) The compilation fails.

```java
package com.venky.oracle.pkg38;

import java.util.List;
import java.util.Optional;
```

```
public class Test {
    public static void main(String[] args) {
        var fruits = List.of("apple", "orange", "banana",
"lemon");
        for(int i =0; i <100; i++) {
            Optional<String> result = fruits.stream().filter(f ->
f.contains("n")).findAny();
            System.out.println(result.orElse("No fruit found"));
        }
    }
}


// Output:Option B code witll produce the same output  allways.
```

Qus no 39



Given the code fragment:

```
Locale locale = Locale.US;
// line 1
double currency = 1_00.00;
System.out.println(formatter.format(currency));
```

You want to display the value of currency as $100.00.

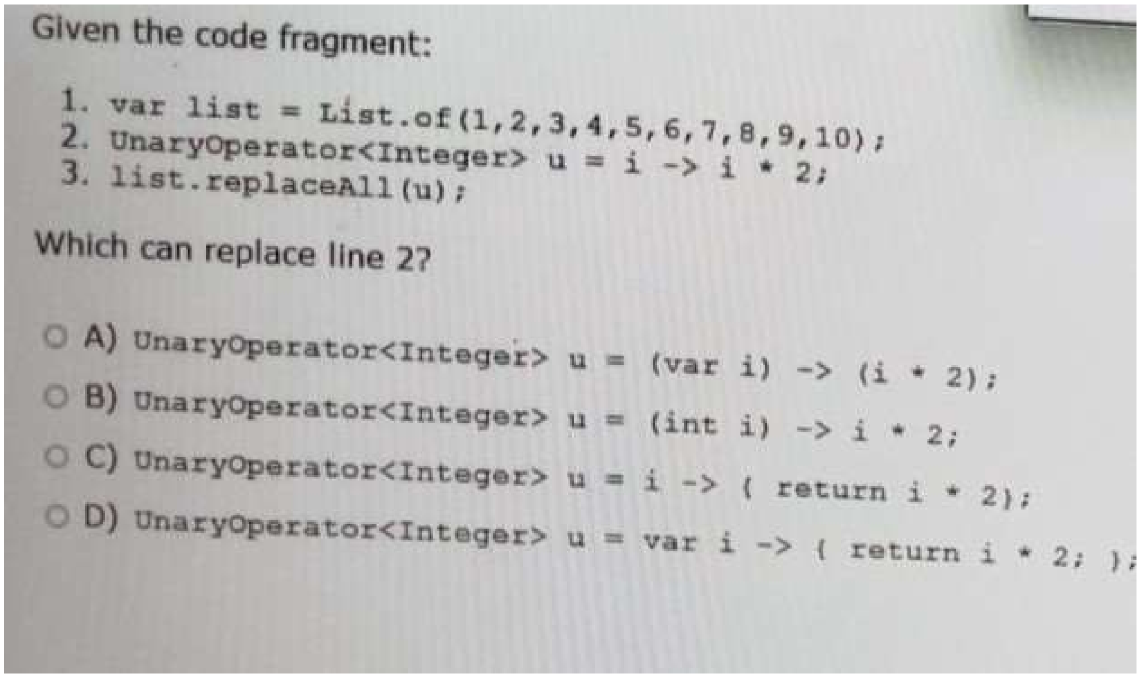
Which code inserted on line 1 will accomplish this?

- A) `NumberFormat formatter = NumberFormat.getInstance(locale);`
- B) `NumberFormat formatter = NumberFormat.getCurrency(locale);`
- C) `NumberFormat formatter = NumberFormat.getInstance(locale).getCurrency();`
- D) `NumberFormat formatter = NumberFormat.getCurrencyInstance(locale);`

```
package com.venky.oracle.pkg39;

import java.text.NumberFormat;

import java.util.Locale;

public class Test {
    public static void main(String[] args) {
        Locale locale = Locale.US;
        double currency=1_00.00;
        //Option A

        //NumberFormat formater=
NumberFormat.getInstance(locale);
```

```
         //Option C
       //  NumberFormat formater=
NumberFormat.getInstance(locale).getCurrency();
         //Option D
        NumberFormat
formater=NumberFormat.getCurrencyInstance(locale);

         System.out.println(formater.format(currency));
     }
}


// Option D ius currect
```

Qus no.40



Given the code fragment:

```
1. var list = List.of(1,2,3,4,5,6,7,8,9,10);
2. UnaryOperator<Integer> u = i -> i * 2;
3. list.replaceAll(u);
```

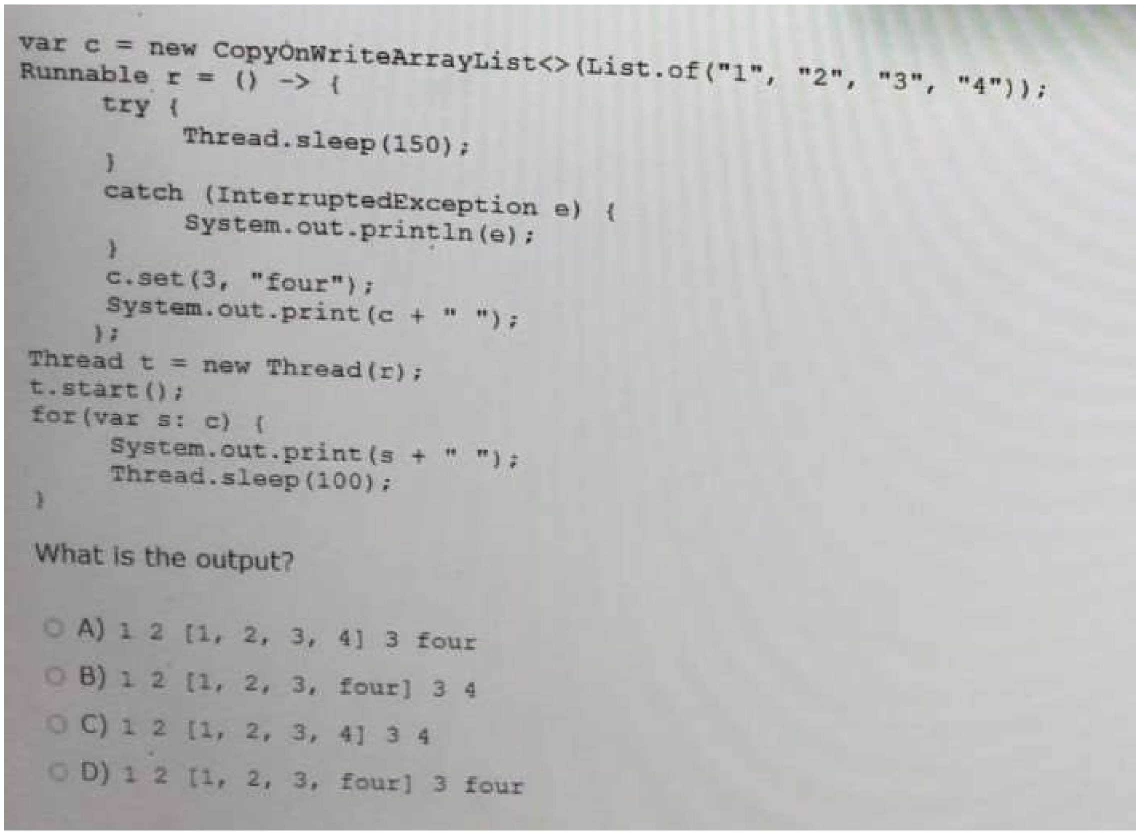
Which can replace line 2?

- ○ A) `UnaryOperator<Integer> u = (var i) -> (i * 2);`
- ○ B) `UnaryOperator<Integer> u = (int i) -> i * 2;`
- ○ C) `UnaryOperator<Integer> u = i -> { return i * 2};`
- ○ D) `UnaryOperator<Integer> u = var i -> { return i * 2; };`

```
package com.venky.oracle.pkg40;

import java.util.List;
import java.util.function.UnaryOperator;

public class Test40 {
    public static void main(String[] args) {
        var list = List.of(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);
```

```
       // UnaryOperator< Integer > u = i -> i * 2;
      UnaryOperator < Integer > u = (var i) -> (i * 2);
      //  UnaryOperator < Integer > u = (int i) -> i * 2;
      //UnaryOperator < Integer > u = i -> { return i*2};
        //   UnaryOperator < Integer > u = var i -> { return i *
2;};
        list.replaceAll(u);
      //  System.out.println(list);

    }
}




//         A) UnaryOperator < Integer > u = (var i) -> (i * 2);
ANSWER
//         B) UnaryOperator < Integer > u = (int i) -> i * 2;
//         C) UnaryOperator < Integer > u = i -> { return i.2};
//         D) UnaryOperator < Integer > u = var i -> { return i *
2;};
```

Qus no 41



```
var c = new CopyOnWriteArrayList<>(List.of("1", "2", "3", "4"));
Runnable r = () -> {
      try {
            Thread.sleep(150);
      }
      catch (InterruptedException e) {
            System.out.println(e);
      }
      c.set(3, "four");
      System.out.print(c + " ");
    };
Thread t = new Thread(r);
t.start();
for(var s: c) {
      System.out.print(s + " ");
      Thread.sleep(100);
}
```

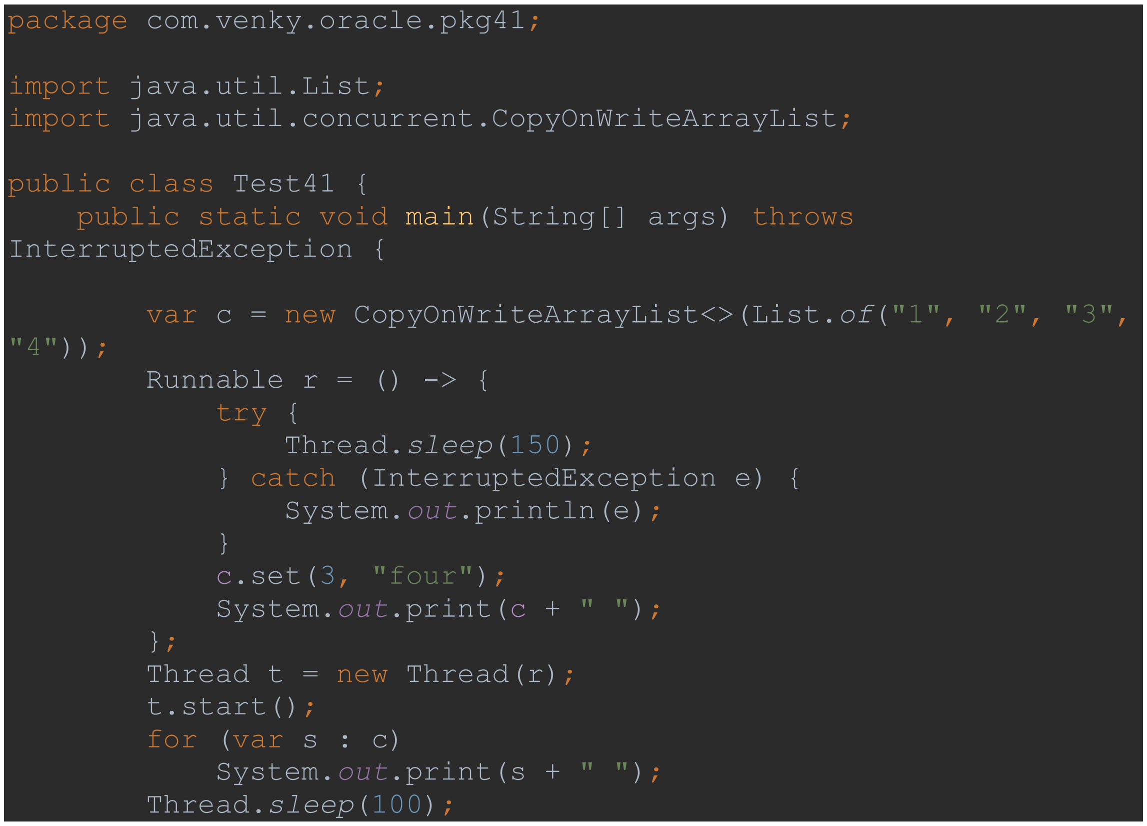
What is the output?

○ A) 1 2 [1, 2, 3, 4] 3 four

○ B) 1 2 [1, 2, 3, four] 3 4

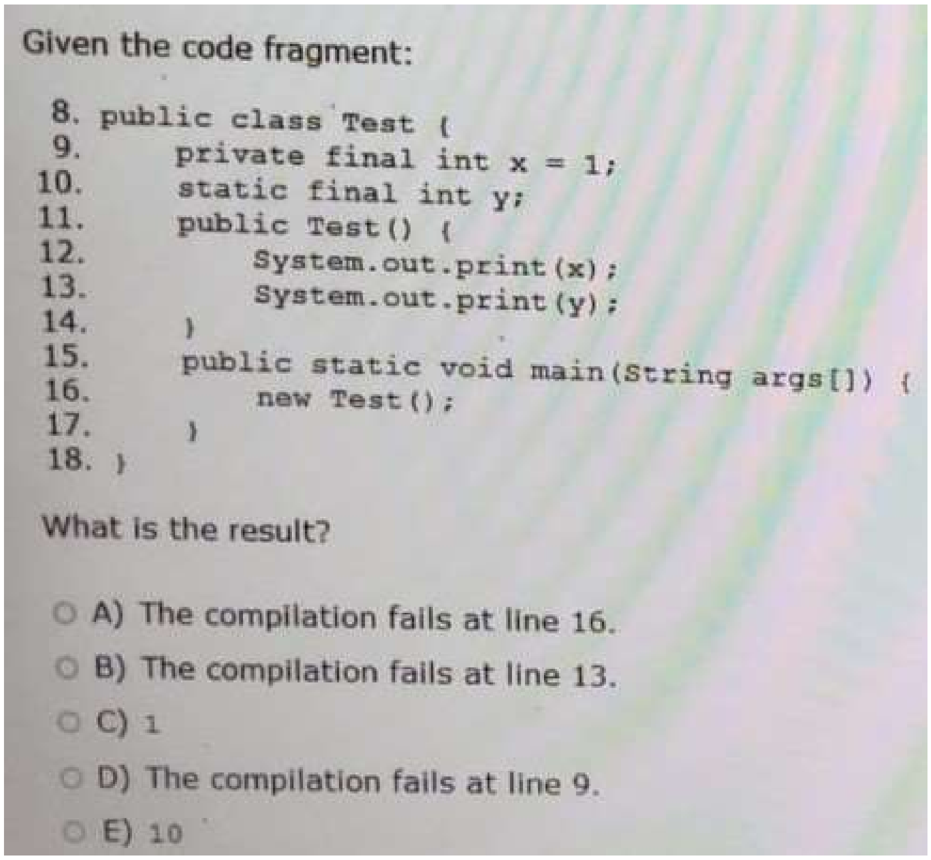
○ C) 1 2 [1, 2, 3, 4] 3 4

○ D) 1 2 [1, 2, 3, four] 3 four



```
package com.venky.oracle.pkg41;

import java.util.List;
import java.util.concurrent.CopyOnWriteArrayList;

public class Test41 {
    public static void main(String[] args) throws
InterruptedException {

        var c = new CopyOnWriteArrayList<>(List.of("1", "2", "3",
"4"));
        Runnable r = () -> {
            try {
                Thread.sleep(150);
            } catch (InterruptedException e) {
                System.out.println(e);
            }
            c.set(3, "four");
            System.out.print(c + " ");
        };
        Thread t = new Thread(r);
        t.start();
        for (var s : c)
            System.out.print(s + " ");
        Thread.sleep(100);
    }
}

//Answer== 1 2  [1, 2, 3, four] 3 4
```

Qus no42



Given the code fragment:

```
8.  public class Test {
9.        private final int x = 1;
10.       static final int y;
11.       public Test() {
12.            System.out.print(x);
13.            System.out.print(y);
14.       }
15.       public static void main(String args[]) {
16.            new Test();
17.       }
18. }
```

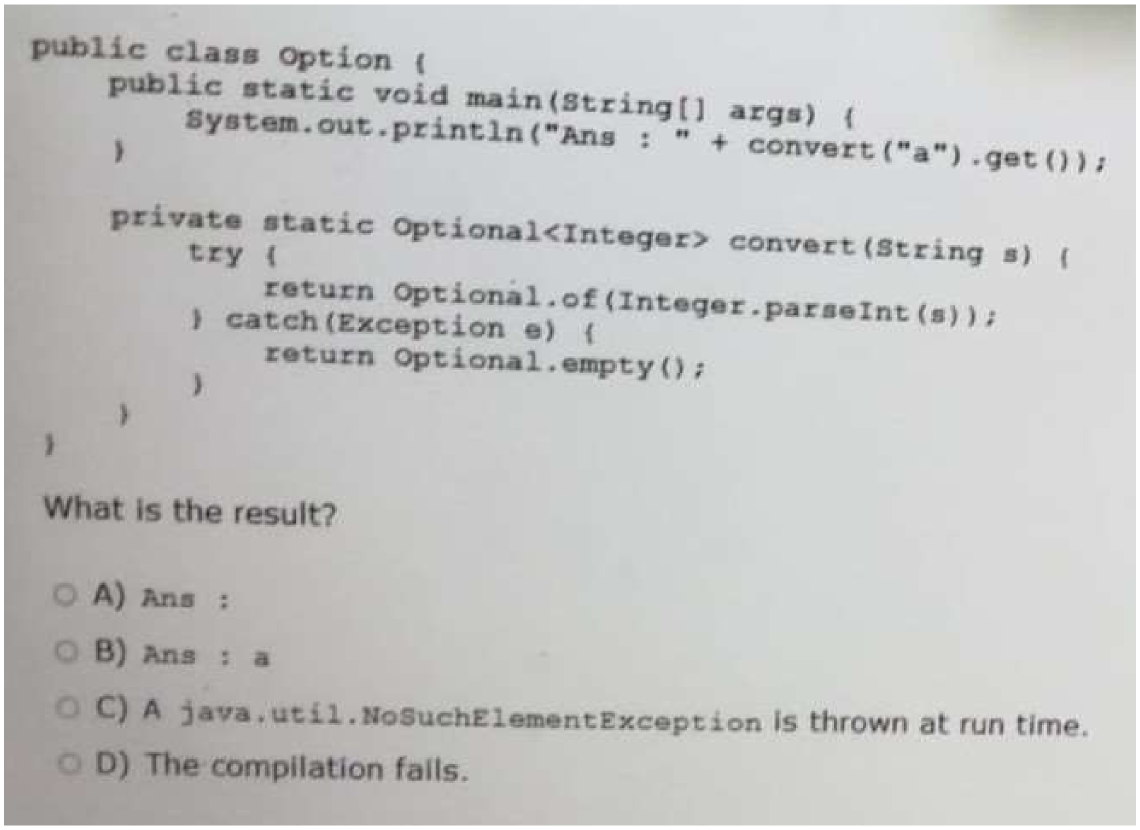
What is the result?

- A) The compilation fails at line 16.
- B) The compilation fails at line 13.
- C) 1
- D) The compilation fails at line 9.
- E) 10

```
package com.venky.oracle.pkg42;

public class Test42 {
    private final int x = 1;
   //  static final int y;

    public Test42() {
        System.out.print(x);
       //  System.out.print(y);
    }

    public static void main(String args[]) {
        new Test42();
    }
}
```

```
//Compilation error at line no. 10
//its  confustion not exactly answer mentioned in question
```

Qus no 43



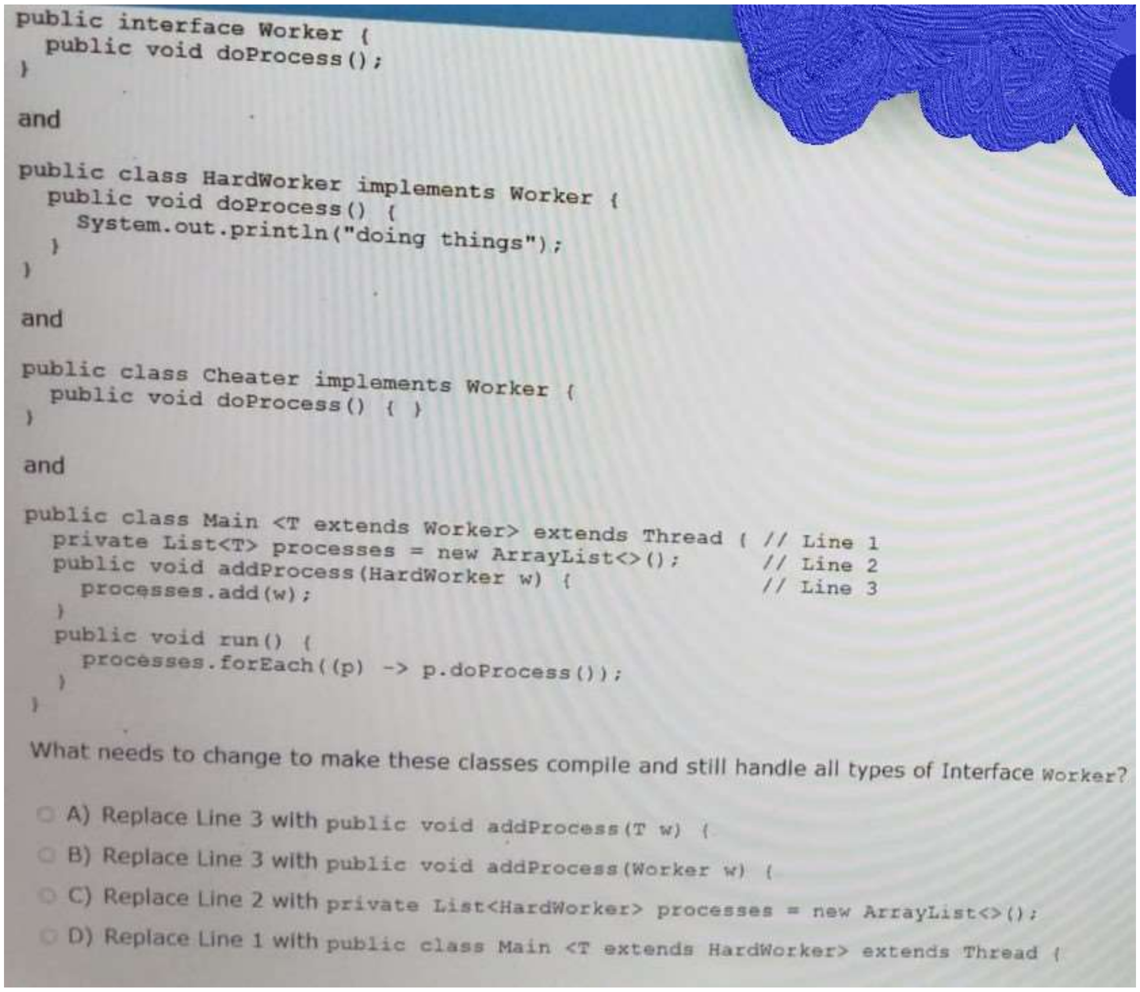
```
public class Option {
    public static void main(String[] args) {
        System.out.println("Ans : " + convert("a").get());
    }

    private static Optional<Integer> convert(String s) {
        try {
            return Optional.of(Integer.parseInt(s));
        } catch(Exception e) {
            return Optional.empty();
        }
    }
}
```

What is the result?

- A) Ans :
- B) Ans : a
- C) A java.util.NoSuchElementException is thrown at run time.
- D) The compilation fails.

```
package com.venky.oracle.pkg43;

import java.util.Optional;

public class Option {
    public static void main(String[] args) {
        System.out.println("Ans : " + convert("a").get());
    }

    private static Optional<Integer> convert(String s) {
        try {
            return Optional.of(Integer.parseInt(s));

        } catch (Exception e) {
            return Optional.empty();

        }

    }

}

//Exception in thread "main" java.util.NoSuchElementException: No
value present  === ANSWER
```

Qus no 44



[redacted]

```
public interface Worker {
  public void doProcess();
}
```

and

```
public class HardWorker implements Worker {
  public void doProcess() {
    System.out.println("doing things");
  }
}
```

and

```
public class Cheater implements Worker {
  public void doProcess() { }
}
```

and

```
public class Main <T extends Worker> extends Thread {  // Line 1
  private List<T> processes = new ArrayList<>();       // Line 2
  public void addProcess(HardWorker w) {               // Line 3
    processes.add(w);
  }
  public void run() {
    processes.forEach((p) -> p.doProcess());
  }
}
```

What needs to change to make these classes compile and still handle all types of Interface `Worker`?

- A) Replace Line 3 with `public void addProcess(T w) {`
- B) Replace Line 3 with `public void addProcess(Worker w) {`
- C) Replace Line 2 with `private List<HardWorker> processes = new ArrayList<>();`
- D) Replace Line 1 with `public class Main <T extends HardWorker> extends Thread {`

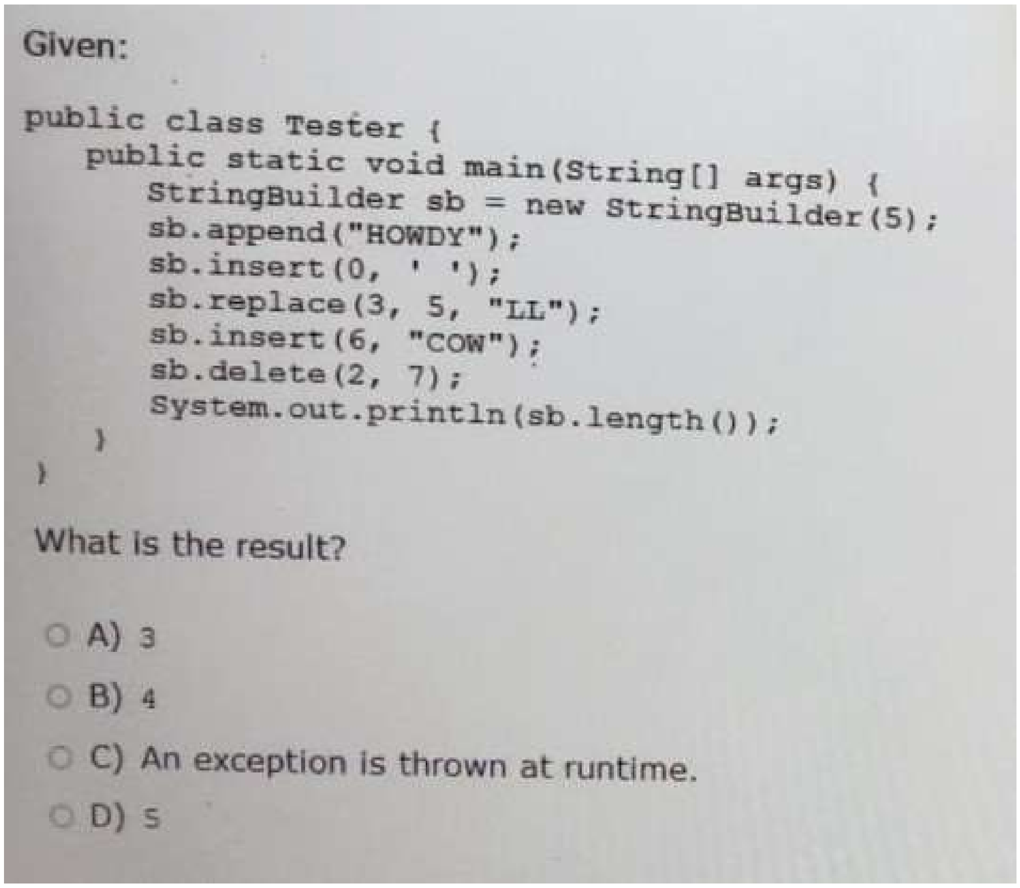
```
package com.venky.oracle.pkg44;

public class Cheater implements Worker{
    public void doProcess() {

    }
}
```

```
package com.venky.oracle.pkg44;

public class Hardworker implements Worker {
    public void doProcess() {
        System.out.println("doing things");
    }
}
```

```
package com.venky.oracle.pkg44;

import java.util.ArrayList;
import java.util.List;

public class Main < T extends Worker > extends Thread {
    private List<T> processes = new ArrayList<>();

    public void addProcess(T w) {
        processes.add(w);
    }

    public void run() {
        processes.forEach((p) -> p.doProcess());

    }
}
//Answer: A) Replace Line 3 with public void addProcess(TW) {
```

```
package com.venky.oracle.pkg44;

public interface Worker {
    public void doProcess();
}
```

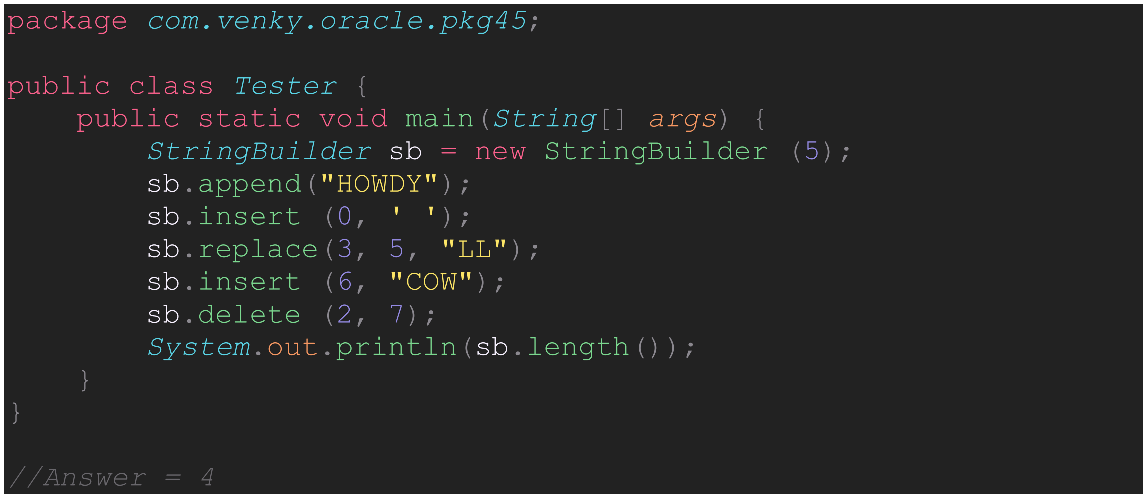
Qus no 45


Given:

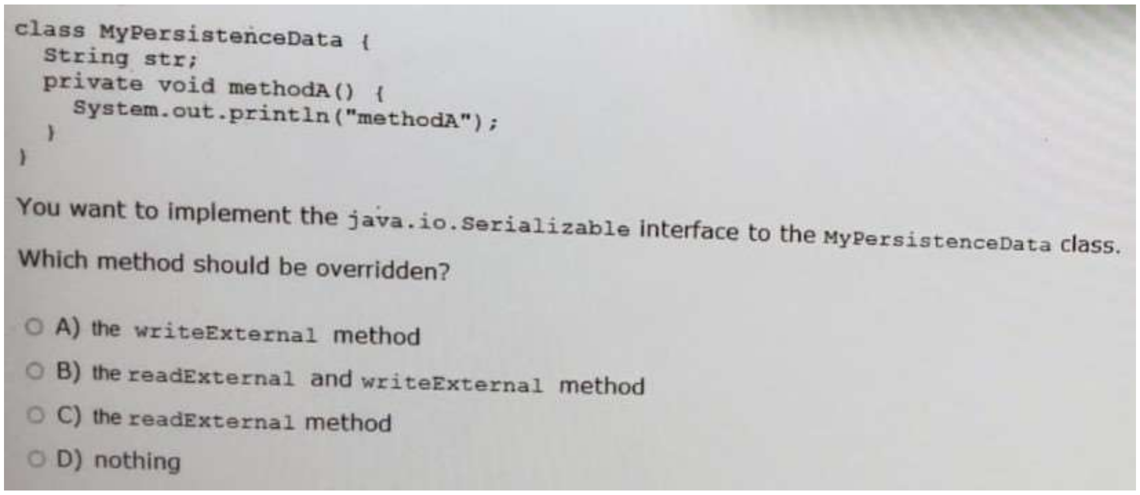
```
public class Tester {
    public static void main(String[] args) {
        StringBuilder sb = new StringBuilder(5);
        sb.append("HOWDY");
        sb.insert(0, ' ');
        sb.replace(3, 5, "LL");
        sb.insert(6, "COW");
        sb.delete(2, 7);
        System.out.println(sb.length());
    }
}
```

What is the result?

- A) 3
- B) 4
- C) An exception is thrown at runtime.
- D) 5


```
package com.venky.oracle.pkg45;

public class Tester {
    public static void main(String[] args) {
        StringBuilder sb = new StringBuilder (5);
        sb.append("HOWDY");
        sb.insert (0, ' ');
        sb.replace(3, 5, "LL");
        sb.insert (6, "COW");
        sb.delete (2, 7);
        System.out.println(sb.length());
    }
}

//Answer = 4
```

Qus no 46



```
class MyPersistenceData {
  String str;
  private void methodA() {
    System.out.println("methodA");
  }
}
```

You want to implement the java.io.Serializable interface to the MyPersistenceData class.

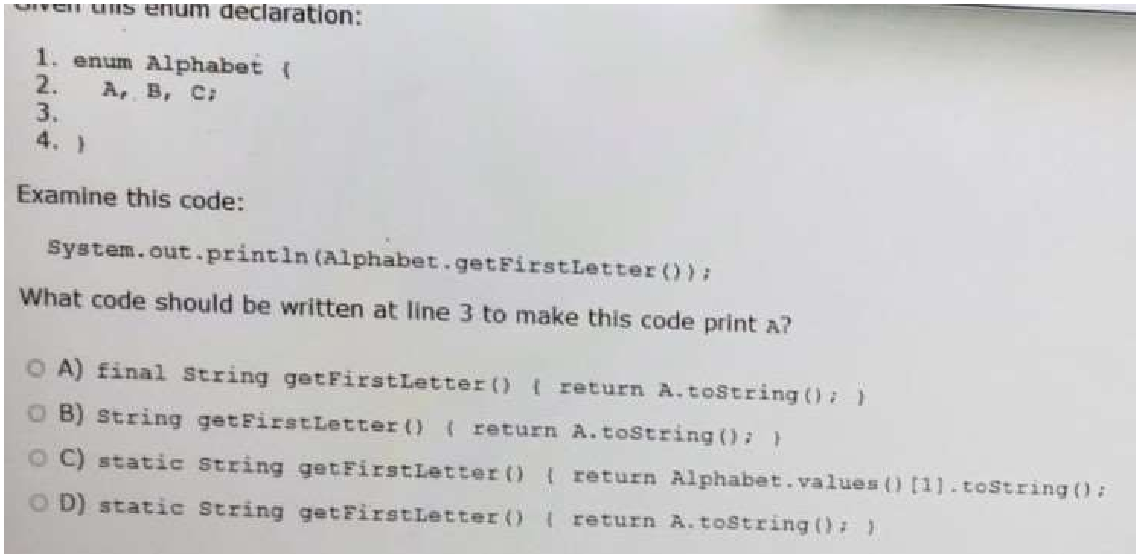
Which method should be overridden?

- A) the writeExternal method
- B) the readExternal and writeExternal method
- C) the readExternal method
- D) nothing

```
package com.venky.oracle.pkg46;

import java.io.Serializable;

public class Main implements Serializable {
    //No need to implement serializable interface

}
```

```
package com.venky.oracle.pkg46;

class MyPersistenceData {
    String str;
    private void methodA() {
        System.out.println("methodA");

    }
}

//  You want to implement the java.io.Serializable interface to
the My PersistenceData class.
//    Which method should be overridden ?
//              OA) the writeExternal method
//              OB) the readExternal and writeExternal method
//              OC) the readExternal method
//              OD) nothing    ##Answer
```

Qus no 47



Given this enum declaration:

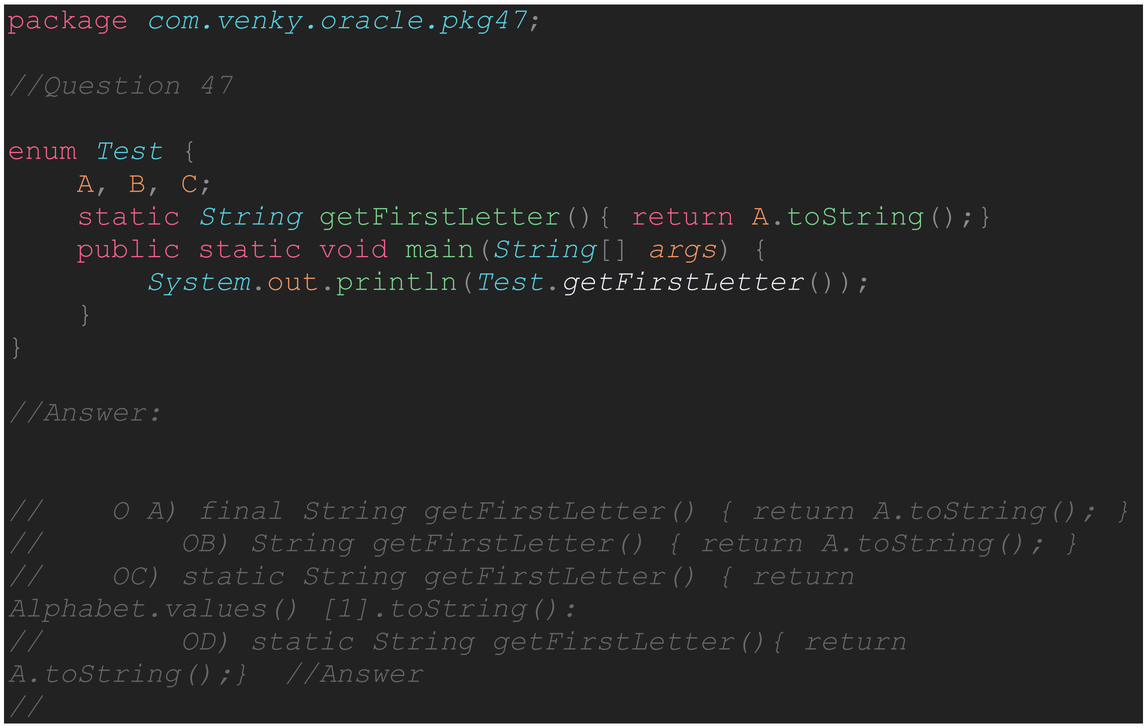
```
1. enum Alphabet {
2.     A, B, C;
3.
4. }
```

Examine this code:

```
System.out.println(Alphabet.getFirstLetter());
```

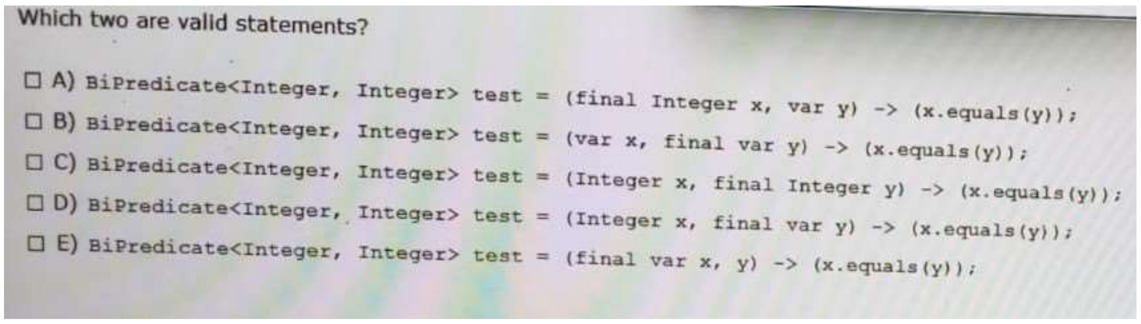
What code should be written at line 3 to make this code print A?

- A) `final String getFirstLetter() { return A.toString(); }`
- B) `String getFirstLetter() { return A.toString(); }`
- C) `static String getFirstLetter() { return Alphabet.values()[1].toString();`
- D) `static String getFirstLetter() { return A.toString(); }`



```
package com.venky.oracle.pkg47;

//Question 47

enum Test {
    A, B, C;
    static String getFirstLetter(){ return A.toString();}
    public static void main(String[] args) {
        System.out.println(Test.getFirstLetter());
    }
}

//Answer:


//     O A) final String getFirstLetter() { return A.toString(); }
//          OB) String getFirstLetter() { return A.toString(); }
//     OC) static String getFirstLetter() { return
Alphabet.values() [1].toString():
//          OD) static String getFirstLetter(){ return
A.toString();}  //Answer
//
```

Qus no 48



Which two are valid statements?

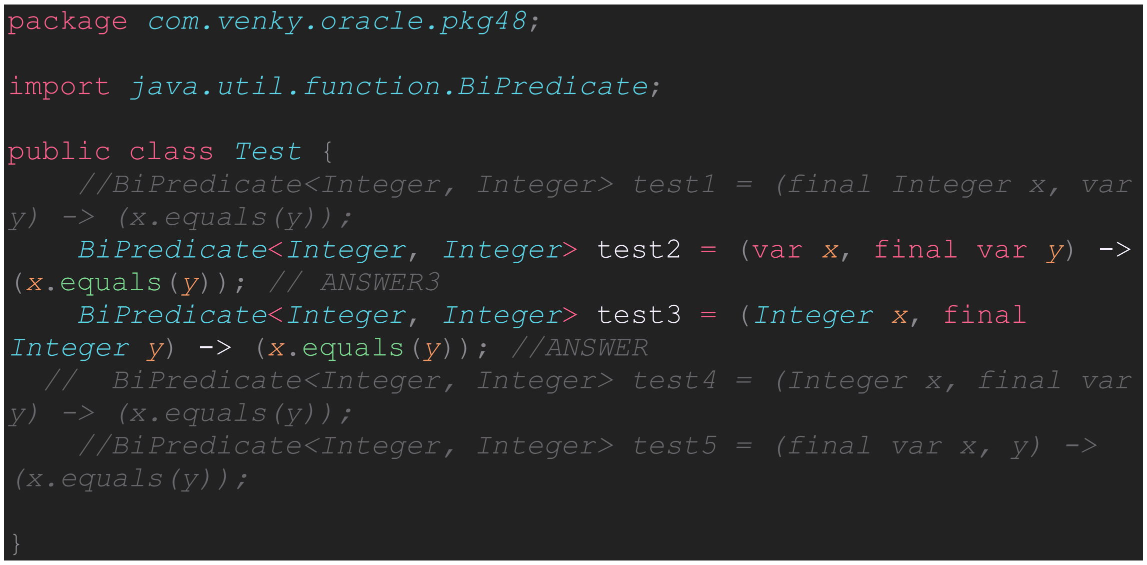
☐ A) `BiPredicate<Integer, Integer> test = (final Integer x, var y) -> (x.equals(y));`

☐ B) `BiPredicate<Integer, Integer> test = (var x, final var y) -> (x.equals(y));`

☐ C) `BiPredicate<Integer, Integer> test = (Integer x, final Integer y) -> (x.equals(y));`

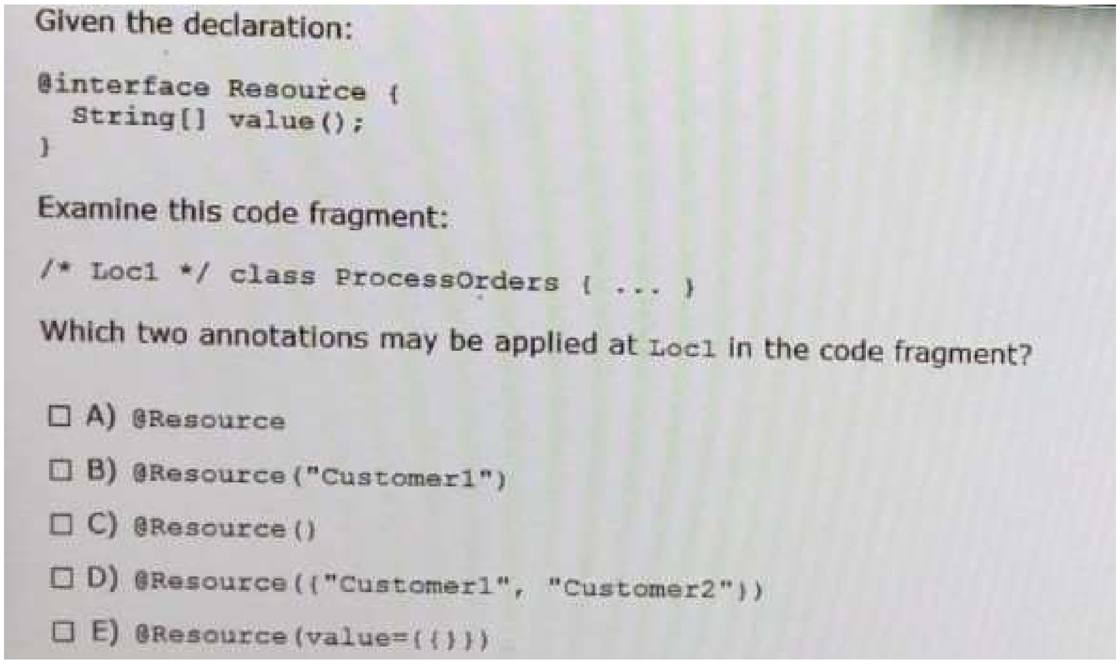
☐ D) `BiPredicate<Integer, Integer> test = (Integer x, final var y) -> (x.equals(y));`

☐ E) `BiPredicate<Integer, Integer> test = (final var x, y) -> (x.equals(y));`



```
package com.venky.oracle.pkg48;

import java.util.function.BiPredicate;

public class Test {
    //BiPredicate<Integer, Integer> test1 = (final Integer x, var y) -> (x.equals(y));
    BiPredicate<Integer, Integer> test2 = (var x, final var y) -> (x.equals(y)); // ANSWER3
    BiPredicate<Integer, Integer> test3 = (Integer x, final Integer y) -> (x.equals(y)); //ANSWER
  //  BiPredicate<Integer, Integer> test4 = (Integer x, final var y) -> (x.equals(y));
    //BiPredicate<Integer, Integer> test5 = (final var x, y) -> (x.equals(y));

}
```

Qus no 49



Given the declaration:

```
@interface Resource {
  String[] value();
}
```

Examine this code fragment:

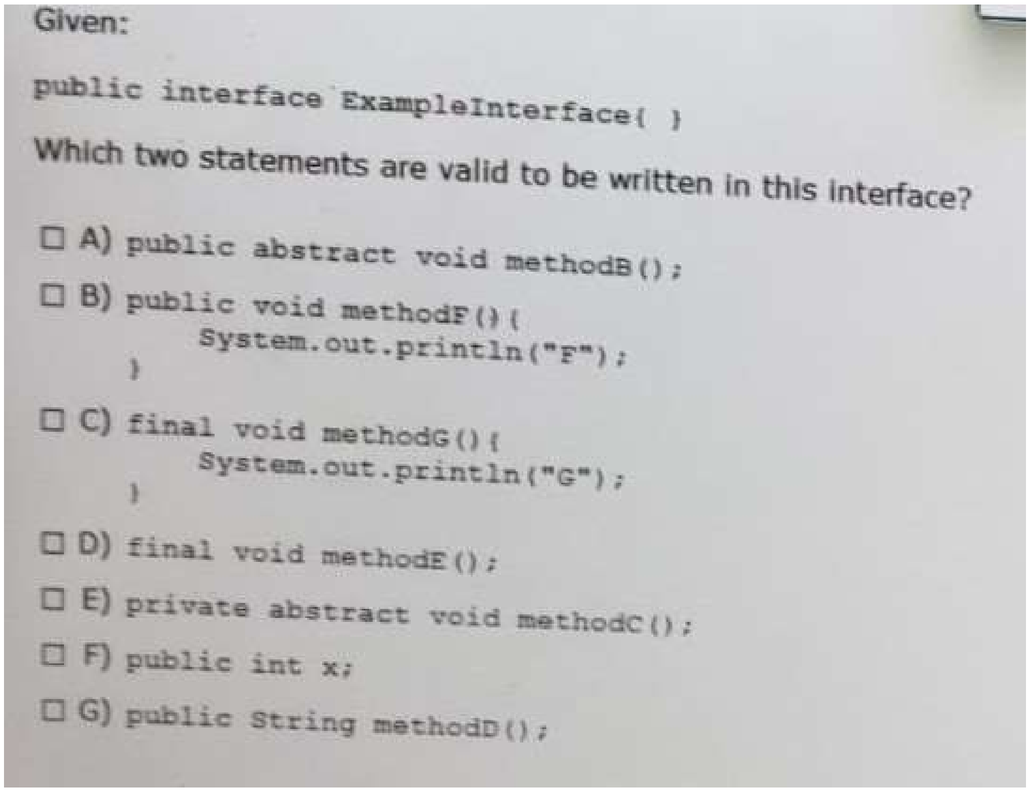
```
/* Loc1 */ class ProcessOrders { ... }
```

Which two annotations may be applied at `Loc1` in the code fragment?

- ☐ A) `@Resource`
- ☐ B) `@Resource("Customer1")`
- ☐ C) `@Resource()`
- ☐ D) `@Resource({"Customer1", "Customer2"})`
- ☐ E) `@Resource(value={{}})`

```
    /*
    Given the declaration:
  @interface Resource
String[] value();
Examine this code fragment: /* Loci */ class Processorders(...)
Which two annotations may be applied at Loci in the code fragment
?
    A) Resource OB) @Resource("Customer1")
    C) Resource() OD) Resource({
        "Customer1",
                "Customer2"))
        E) Resource(value = {
                {}
     */
}
```

Qus no 50

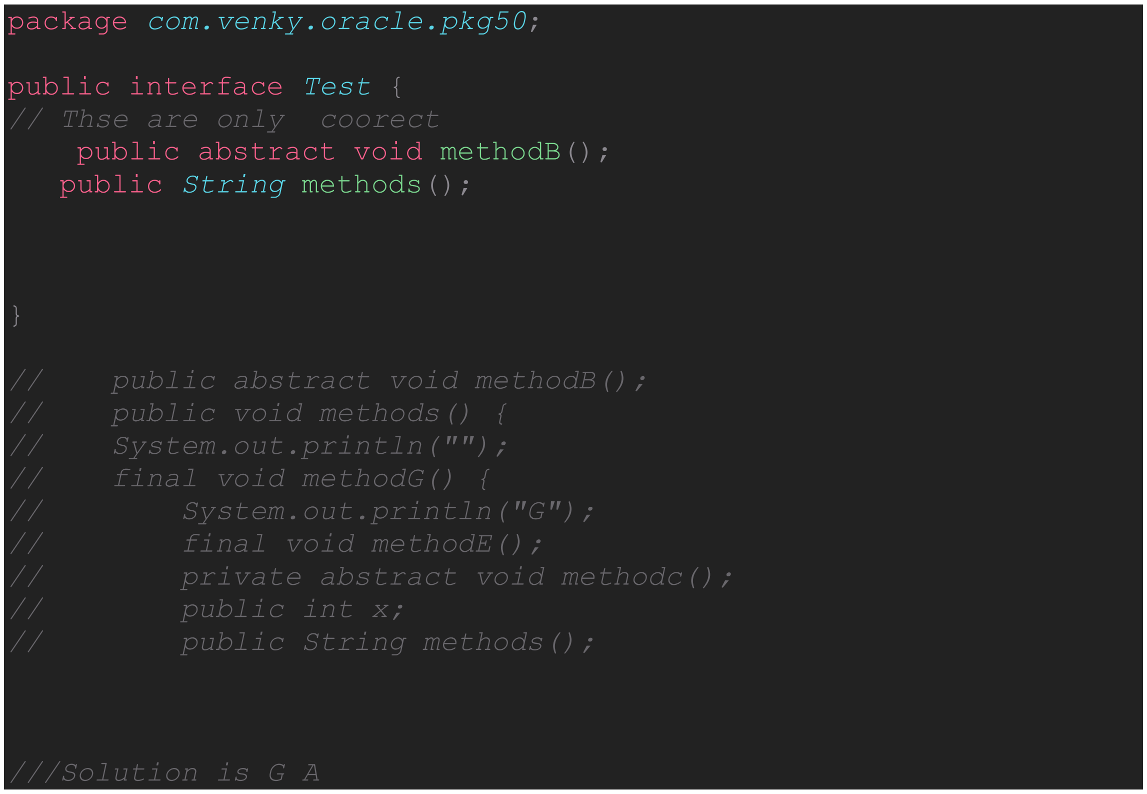


Given:

```
public interface ExampleInterface{ }
```

Which two statements are valid to be written in this interface?

- ☐ A) `public abstract void methodB();`
- ☐ B)
```
public void methodF(){
        System.out.println("F");
    }
```
- ☐ C)
```
final void methodG(){
        System.out.println("G");
    }
```
- ☐ D) `final void methodE();`
- ☐ E) `private abstract void methodC();`
- ☐ F) `public int x;`
- ☐ G) `public String methodD();`



```
package com.venky.oracle.pkg50;

public interface Test {
// Thse are only  coorect
    public abstract void methodB();
   public String methods();




}

//    public abstract void methodB();
//    public void methods() {
//    System.out.println("");
//    final void methodG() {
//        System.out.println("G");
//        final void methodE();
//        private abstract void methodc();
//        public int x;
//        public String methods();



///Solution is G A
```