Instructions:

- **Questions 1 to 35 are mandatory**
- **You can code for all given programs but attempt at least 7 of them ;)**
- **All the best! You need it… :)**

1. Given:

```
 2. public class Bunnies {
 3. static int count = 0;
 4. Bunnies() {
 5. while(count < 10) new Bunnies(++count);
 6. }
 7. Bunnies(int x) { super(); }
 8. public static void main(String[] args) {
 9. new Bunnies();
10. new Bunnies(count);
11. System.out.println(count++);
12. }
13. }
```

What is the result?
A. 9
B. 10
C. 11
D. 12
E. Compilation fails.
F. An exception is thrown at runtime.

Answer – B. 10

2. Given:

```
 2. public class Jail {
 3. private int x = 4;
 4. public static void main(String[] args) {
 5. protected int x = 6;
 6. new Jail().new Cell().slam();
 7. }
 8. class Cell {
 9. void slam() { System.out.println("throw away key " + x); }
10. }
11. }
```

Which are true? (Choose all that apply.)

A. Compilation succeeds.
B. The output is "throw away key 4".
C. The output is "throw away key 6".
D. Compilation fails due to an error on line 5.

E. Compilation fails due to an error on line 6.
F. Compilation fails due to an error on line 9.

Answer D. Compilation fails due to an error on line 5.

3. Given:

```
 2. public class Fabric extends Thread {
 3. public static void main(String[] args) {
 4. Thread t = new Thread(new Fabric());
 5. Thread t2 = new Thread(new Fabric());
 6. t.start();
 7. t2.start();
 8. }
 9. public static void run() {
10. for(int i = 0; i < 2; i++)
11. System.out.print(Thread.currentThread().getName() + " ");
12. }
13. }
```

Which are true? (Choose all that apply.)

A. Compilation fails.
B. No output is produced.
C. The output could be Thread-1 Thread-3 Thread-1 Thread-2
D. The output could be Thread-1 Thread-3 Thread-1 Thread-3
E. The output could be Thread-1 Thread-1 Thread-2 Thread-2
F. The output could be Thread-1 Thread-3 Thread-3 Thread-1
G. The output could be Thread-1 Thread-3 Thread-1 Thread-1

Answer B. No output is produced.

4. Given:

```
2. class Feline { }
 3. public class BarnCat2 extends Feline {
 4. public static void main(String[] args) {
 5. Feline ff = new Feline();
 6. BarnCat2 b = new BarnCat2();
 7. // insert code here
 8. }
 9. }
```

Which, inserted independently at line 7, compile? (Choose all that apply.)

A. if(b instanceof ff) System.out.print("1 ");
B. if(b.instanceof(ff)) System.out.print("2 ");
C. if(b instanceof Feline) System.out.print("3 ");

D. if(b instanceOf Feline) System.out.print("4 ");
E. if(b.instanceof(Feline)) System.out.print("5 ");

Answer C. if(b instanceof Feline) System.out.print("3 ");

5. Given:

```
2. public class Choosy {
 3. public static void main(String[] args) {
 4. String result = "";
 5. int x = 7, y = 8;
 6. if(x == 3) { result += "1"; }
 7. else if (x > 9) { result += "2"; }
 8. else if (y < 9) { result += "3"; }
 9. else if (x == 7) { result += "4"; }
10. else { result += "5"; }
11. System.out.println(result);
12. }
13. }
```

What is the result? (Choose all that apply.)

A. 3
B. 34
C. 35
D. 345
E. Compilation fails due to an error on line 5.
F. Compilation fails due to errors on lines 8 and 9.
G. Compilation fails due to errors on lines 7, 8, and 9.

Answer A

6. Given:

```
 1. public class Twine {
 2. public static void main(String[] args) {
 3. String s = "";
 4. StringBuffer sb1 = new StringBuffer("hi");
 5. StringBuffer sb2 = new StringBuffer("hi");
 6. StringBuffer sb3 = new StringBuffer(sb2);
 7. StringBuffer sb4 = sb3;
 8. if(sb1.equals(sb2)) s += "1 ";
 9. if(sb2.equals(sb3)) s += "2 ";
10. if(sb3.equals(sb4)) s += "3 ";
11. String s2 = "hi";
12. String s3 = "hi";
13. String s4 = s3;
14. if(s2.equals(s3)) s += "4 ";
15. if(s3.equals(s4)) s += "5 ";
```

```
16. System.out.println(s);
17. }
18. }
```

What is the result?
A. 1 3
B. 1 5
C. 1 2 3
D. 1 4 5
E. 3 4 5
F. 1 3 4 5
G. 1 2 3 4 5
H. Compilation fails.

Answer E

7. Which are true? (Choose all that apply.)

A. All classes of Exception extend Error.
B. All classes of Error extend Exception.
C. All Errors must be handled or declared.
D. All classes of Exception extend Throwable. --
E. All Throwables must be handled or declared.
F. All Exceptions must be handled or declared.
G. RuntimeExceptions need never be handled or declared. --

Answer D G

8. Given:

```
2. import java.util.*;
 3. public class Birthdays {
 4. public static void main(String[] args) {
 5. Map<Friends, String> hm = new HashMap<Friends, String>();
 6. hm.put(new Friends("Charis"), "Summer 2009");
 7. hm.put(new Friends("Draumur"), "Spring 2002");
 8. Friends f = new Friends(args[0]);
 9. System.out.println(hm.get(f));
10. }
11. }
12. class Friends {
13. String name;
14. Friends(String n) { name = n; }
15. }
```

And the command line invocation:
 # java Birthdays Draumur

What is the result?

A. null

B. Draumur
C. Spring 2002
D. Compilation fails.
E. The output is unpredictable.
F. An exception is thrown at runtime.
G. Friends@XXXX (where XXXX is a representation of a hashcode)

Answer F. An exception is thrown at runtime.

9. Given:

```
2. import java.util.*;
 3. class Cereal { }
 4. public class Flakes extends Cereal {
 5. public static void main(String[] args) {
 6. List<Flakes> c0 = new List<Flakes>();
 7. List<Cereal> c1 = new ArrayList<Cereal>();
 8. List<Cereal> c2 = new ArrayList<Flakes>();
 9. List<Flakes> c3 = new ArrayList<Cereal>();
10. List<Object> c4 = new ArrayList<Flakes>();
11. ArrayList<Cereal> c5 = new ArrayList<Flakes>();
12. }
13. }
```

Which are true? (Choose all that apply.)

A. Compilation succeeds.
B. Compilation fails due to an error on line 6.
C. Compilation fails due to an error on line 7.
D. Compilation fails due to an error on line 8.
E. Compilation fails due to an error on line 9.
F. Compilation fails due to an error on line 10.
G. Compilation fails due to an error on line 11.

Answer B. Compilation fails due to an error on line 6.

10. Given:

```
3. public class RediMix extends Concrete {
 4. RediMix() { System.out.println("r "); }
 5. public static void main(String[] args) {
 6. new RediMix();
 7. }
 8. }
 9. class Concrete extends Sand {
10. Concrete() { System.out.print("c "); }
11. private Concrete(String s) { }
12. }
13. abstract class Sand {
```

```
14. Sand() { System.out.print("s "); }
15. }
```

What is the result?

A. r
B. c r
C. r c
D. s c r
E. r c s
F. Compilation fails due to a single error in the code.
G. Compilation fails due to multiple errors in the code.

Answer D. s c r

11. Which statement(s) are true? (Choose all that apply.)

A. Coupling is the OO principle most closely associated with hiding a class's implementation details.
B. Coupling is the OO principle most closely associated with making sure classes know about other classes only through their APIs.
C. Coupling is the OO principle most closely associated with making sure a class is designed with a single, well-focused purpose.
D. Coupling is the OO principle most closely associated with allowing a single object to be seen as having many types.

Answer C

12. Given:

```
2. class Mosey implements Runnable {
 3. public void run() {
 4. for(int i = 0; i < 1000; i++) {
 5. System.out.print(Thread.currentThread().getId() + "-" + i + " ");
 6. } } }
 7. public class Stroll {
 8. public static void main(String[] args) throws Exception {
 9. Thread t1 = new Thread(new Mosey());
10. // insert code here
11. }
12. }
```

Which of the following code fragments, inserted independently at line 10, will probably run most (or all) of the main thread's run() method invocation before running most of the t1 thread's run() method invocation? (Choose all that apply.)

A. t1.setPriority(1);
new Mosey().run();
t1.start();
B. t1.setPriority(9);

new Mosey().run();
t1.start();
C. t1.setPriority(1);
t1.start();
new Mosey().run();
D. t1.setPriority(8);
t1.start();
new Mosey().run();
Answer B

13. Given:

```
37. boolean b = false;
38. int i = 7;
39. double d = 1.23;
40. float f = 4.56f;
41.
42. // insert code here
```

Which line(s) of code, inserted independently at line 42, will compile and run without exception? (Choose all that apply.)

A. System.out.printf(" %b", b); --
B. System.out.printf(" %i", i);
C. System.out.format(" %d", d);
D. System.out.format(" %d", i); --
E. System.out.format(" %f", f); --

Answer A, D, E

14. Given:

```
1. import java.util.*;
 2. public class MyPancake implements Pancake {
 3. public static void main(String[] args) {
 4. List<String> x = new ArrayList<String>();
 5. x.add("3"); x.add("7"); x.add("5");
 6. List<String> y = new MyPancake().doStuff(x);
 7. y.add("1");
 8. System.out.println(x);
 9. }
10. List<String> doStuff(List<String> z) {
11. z.add("9");
12. return z;
13. }
14. }
15. interface Pancake {
16. List<String> doStuff(List<String> s);
17. }
```

What is the most likely result?

A. [3, 7, 5]
B. [3, 7, 5, 9]
C. [3, 7, 5, 9, 1]
D. Compilation fails.
E. An exception is thrown at runtime.

Answer D

15. Given:

```
3. import java.util.*;
 4. public class VLA2 implements Comparator<VLA2> {
 5. int dishSize;
 6. public static void main(String[] args) {
 7. VLA2[] va = {new VLA2(40), new VLA2(200), new VLA2(60)};
 8.
 9. Arrays.sort(va, va[0]);
10. int index = Arrays.binarySearch(va, new VLA2(40), va[0]);
11. System.out.print(index + " ");
12. index = Arrays.binarySearch(va, new VLA2(80), va[0]);
13. System.out.print(index);
14. }
15. public int compare(VLA2 a, VLA2 b) {
16. return b.dishSize - a.dishSize;
17. }
18. VLA2(int d) { dishSize = d; }
19. }
```

What is the result?

A. 0 -2
B. 0 -3
C. 2 -1
D. 2 -2
E. Compilation fails.
F. An exception is thrown at runtime.

Answer E. Compilation fails.

16. Given a directory structure:
- baseDir
- testDir
- subDir2
- Shackelton.txt

and given the following code:

```
12. String name = "testDir" + File.pathSeparator + "subDir2"
 + File.pathSeparator + "Shackelton.txt";
```

13. File f = new File(name);
14. System.out.println("exists " + f.exists());

Assuming the proper import statements and exception handling, which statements must be true in order for the output to be "exists true"? (Choose three.)

A. Line 12 is correct as it stands.
B. Line 14 is correct as it stands.
C. The program must be invoked from the baseDir directory.
D. The program must be invoked from the testDir directory.
E. Line 12 must use File.separator instead of File.pathSeparator.
F. Line 14 must use the method fileExists() instead of exists().

Answer F, C, D

17. Given:

1. import java.io.*;
2. import java.util.*;
3. import static java.lang.Short.*;
4. import static java.lang.Long.*;
5. public class MathBoy {
6. public static void main(String[] args) {
7. long x = 123456789;
8. short y = 22766; // maximum value of a short is 32767
9. System.out.printf("%1$+10d %2$010d ", x, MAX_VALUE - y);
10. System.out.println(new Date());
11. }
12. }

Which are true? (Choose all that apply.)

A. Compilation fails.
B. The output will include "+"
C. The output will include "10001"
D. The output will include "0000010001"
E. The output will include today's date.
F. The output will include the number of milliseconds from January 1, 1970 until today.

Answer A. Compilation fails.

18. Given:

1. public class WeatherTest {
2. static Weather w;
3. public static void main(String[] args) {

4. System.out.print(w.RAINY.count + " " + w.Sunny.count + " ");
5. }
6. }

```
 7. enum Weather {
 8. RAINY, Sunny;
 9. int count = 0;
10. Weather() {
11. System.out.print("c ");
12. count++;
13. }
14. }
```

What is the result?

A. c 1 c 1
B. c 1 c 2
C. c c 1 1
D. c c 1 2
E. c c 2 2
F. Compilation fails.
G. An exception is thrown at runtime.

Answer C. c c 1 1

19. Given:

```
2. import java.text.*;
 3. public class Gazillion {
 4. public static void main(String[] args) throws Exception {
 5. String s = "123.456xyz";
 6. NumberFormat nf = NumberFormat.getInstance();
 7. System.out.println(nf.parse(s));
 8. nf.setMaximumFractionDigits(2);
 9. System.out.println(nf.format(s));
10. }
11. }
```

Which are true? (Choose all that apply.)

A. Compilation fails.
B. The output will contain "123.45 "
C. The output will contain "123.456"
D. The output will contain "123.456xyz"
E. An exception will be thrown at runtime.

Answer B. The output will contain "123.45 "

20. Given that the current directory is bigApp, and the following directory structure:

```
bigApp
 |-- classes
 |-- com
 |-- wickedlysmart
```

|-- BigAppMain.class

And the code:

```
package com.wickedlysmart;
public class BigAppMain {
 public static void main(String[] args) {
 System.out.println("big app");
 }
}
```

Which will invoke BigAppMain? (Choose all that apply.)

A. java classes/com.wickedlysmart.BigAppMain
B. java classes com/wickedlysmart/BigAppMain
C. java classes.com.wickedlysmart.BigAppMain
D. java -cp classes com.wickedlysmart.BigAppMain
E. java -cp /classes com.wickedlysmart.BigAppMain
F. java -cp .:classes com.wickedlysmart.BigAppMain
G. java -cp classes/com/wickedlysmart com.wickedlysmart.BigAppMain

21. Given:

```
2. class Game {
 3. static String s = "-";
 4. String s2 = "s2";
 5. Game(String arg) { s += arg; }
 6. }
 7. public class Go extends Game {
 8. Go() { super(s2); }
 9. { s += "i "; }
10. public static void main(String[] args) {
11. new Go();
12. System.out.println(s);
13. }
14. static { s += "sb "; }
15. }
```

What is the result?

A. -sb i s2
B. -sb s2 i
C. -s2 i sb
D. -s2 sb i
E. Compilation fails.
F. An exception is thrown at runtime.

Answer E. Compilation fails.
s2 should be static

22. Given:

```
2. public class Salmon extends Thread {
 3. public static long id;
 4. public void run() {
 5. for(int i = 0; i < 4; i++) {
 6. // insert code here
 7. new Thread(new Salmon()).start();
 8. throw new Error();
 9. }
10. System.out.print(i + " ");
11. } }
12. public static void main(String[] args) {
13. Thread t1 = new Salmon();
14. id = t1.getId();
15. t1.start();
16. } }
```

And the two code fragments:

```
I. if(i == 2 && id == Thread.currentThread().getId()) {
II. if(i == 2) {
```

When inserting either fragment, independently at line 6, which are true? (Choose all that apply.)

A. Both fragments produce the same output.
B. Both fragments will end in about the same amount of time.
C. Compilation fails, regardless of which fragment is inserted.
D. Regardless of which fragment is inserted, output ends once the Error is thrown.
E. Regardless of which fragment is inserted, output continues after the Error is thrown.

Answer C

23. Given:

```
2. public class Internet {
 3. private int y = 8;
 4. public static void main(String[] args) {
 5. new Internet().go();
 6. }
 7. void go() {
 8. int x = 7;
 9. TCPIP ip = new TCPIP();
10. class TCPIP {
11. void doit() { System.out.println(y + x); }
12. }
13. ip.doit();
14. }
```

```
15. }
```

What is the result? (Choose all that apply.)

A. Compilation succeeds.
B. Compilation fails due to an error on line 3.
C. Compilation fails due to an error on line 8.
D. Compilation fails due to an error on line 9.
E. Compilation fails due to an error on line 10.
F. Compilation fails due to accessing x on line 11.
G. Compilation fails due to accessing y on line 11.

Answer D. Compilation fails due to an error on line 9.
F. Compilation fails due to accessing x on line 11.
G. Compilation fails due to accessing y on line 11.

24. Given:

```
4. public static void main(String[] args) {
 5. try {
 6. if(args.length == 0) throw new Exception();
 7. }
 8. catch (Exception e) {
 9. System.out.print("done ");
10. doStuff(); // assume this method compiles
11. }
12. finally {
13. System.out.println("finally ");
14. }
15. }
```

Which are possible outputs? (Choose all that apply.)

A. "done "
B. "finally "
C. "done finally "
D. Compilation fails.
E. No output is produced.

Answer C. "done finally "

25. Given:

```
3. class A { }
 4. class B extends A { }
 5. class C extends B { }
 6. public class Carpet<V extends B> {
 7. public <X extends V> Carpet<? extends V> method(Carpet<? super X> e) {
 8. // insert code here
```

```
 9. } }
```

Which, inserted independently at line 8, will compile? (Choose all that apply.)

A. return new Carpet<X>(); --
B. return new Carpet<V>(); --
C. return new Carpet<A>();
D. return new Carpet<B>();
E. return new Carpet<C>();

Answer A, B

26. Given:

```
1. class One {
 2. int x = 0;
 3. { assert x == 1; }
 4. }
 5. public class Two {
 6. public static void main(String[] args) {
 7. int y = 0;
 8. assert y == 0;
 9. if(args.length > 0)
10. new One();
11. }
12. }
```

Which of the following will run without error? (Choose all that apply.)

A. java Two --
B. java Two x --
C. java -ea Two --
D. java -ea Two x
E. java -ea:One Two --
F. java -ea:One Two x
G. java -ea:Two Two x

Answer A,B,C,E

27. Given:

```
2. class SafeDeposit {
 3. private static SafeDeposit singleton;
 4. public static SafeDeposit getInstance(int code) {
 5. if(singleton == null)
 6. singleton = new SafeDeposit(code);
 7. return singleton;
 8. }
 9. private int code;
10. private SafeDeposit(int c) { code = c; }
11. int getCode() { return code; }
```

```
12. }
13. public class BeSafe {
14. // insert lots of code here
25. }
```

Which are true? (Choose all that apply.)

A. Compilation fails.
B. Class BeSafe can create many instances of SafeDeposit.
C. Class BeSafe CANNOT create any instances of SafeDeposit.
D. Class BeSafe can create only one instance of SafeDeposit.
E. Class BeSafe can create instances of SafeDeposit without using the getInstance() method.
F. Once class BeSafe has created an instance of SafeDeposit, it cannot change the value of the instance's "code" variable.

Answer D, F

28. Given:

```
2. public class DMV implements Runnable {
 3. public static void main(String[] args) {
 4. DMV d = new DMV();
 5. new Thread(d).start();
 6. Thread t1 = new Thread(d);
 7. t1.start();
 8. }
 9. public void run() {
10. for(int j = 0; j < 4; j++) { do1(); do2(); }
11. }
12. void do1() {
13. try { Thread.sleep(1000); }
14. catch (Exception e) { System.out.print("e "); }
15. }
16. synchronized void do2() {
17. try { Thread.sleep(1000); }
18. catch (Exception e) { System.out.print("e "); }
19. } }
```

Which are true? (Choose all that apply.)

A. Compilation fails.
B. The program's minimum execution time is about 8 seconds.
C. The program's minimum execution time is about 9 seconds.
D. The program's minimum execution time is about 12 seconds.
E. The program's minimum execution time is about 16 seconds.
F. Un-synchronizing do2() changes the program's minimum execution time by only a few milliseconds.

Answer C. The program's minimum execution time is about 9 seconds.

29. Given the current directory is bigApp, and the directory structure:

```
bigApp
   |-- classes
         |-- Cloned.class
```

And the file:

```
public class Cloned {
  public static void main(String[] args) {
    System.out.println("classes");
    assert(Integer.parseInt(args[0]) > 0);
    }
    }
```

Which will produce the output "classes" followed by an AssertionError? (Choose all that apply.)

A. java-cpclassesCloned-4

B. java-cpclasses-eaCloned

C. java -ea-cp classes Cloned -4

D. java -ea -cp classes Cloned 4

E. java-ea,cpclassesCloned4

F. java -ea -cp classes Cloned -4

G. java -cp classes Cloned -4 -ea

30. Given:

```
1. interface Syrupable {
2.   void getSugary();
3. }

4. abstract class Pancake implements Syrupable { }
5.
6. class BlueBerryPancake implements Pancake {
7.   public void getSugary() { ; }
8. }
9. class SourdoughBlueBerryPancake extends BlueBerryPancake {
10.   void getSugary(int s) { ; }
11. }
```

Which are true? (Choose all that apply.)

A. Compilation succeeds.

B. Compilation fails due to an error on line 2.

C. Compilation fails due to an error on line 4.

D. Compilation fails due to an error on line 6. --

E. Compilation fails due to an error on line 7.

F. Compilation fails due to an error on line 9. --

G. Compilation fails due to an error on line 10. –

Answer DFG

31. Given:

```
1. public class Endless {
2.   public static void main(String[] args) {
3. int i = 0;
4. short s = 0;
5. for(int j = 0, k = 0; j < 3; j++) ;
6. for(int j = 0; j < 3; counter(j)) ;
7. for(int j = 0, int k = 0; j < 3; j++) ;
8. for(; i < 5; counter(5), i++) ;
9. for(i = 0; i < 3; i++, System.out.print("howdy ")) ;
10.  }
11.  static int counter(int y) { return y + 1; }
12. }
```

What is the result? (Choose all that apply.)

A. howdy howdy howdy
B. The code runs in an endless loop.

C. Compilation fails due to an error on line 5.

D. Compilation fails due to an error on line 6.

E. Compilation fails due to an error on line 7.

F. Compilation fails due to an error on line 8.

G. Compilation fails due to an error on line 9.

Answer E. Compilation fails due to an error on line 7.

32. Given:

```
2. class Big {
3.   void doStuff(int x) { }
4. }
5. class Heavy extends Big {
6.         6.   // void doStuff(byte b) { }
7.
```

```
8.              7.    // protected void doStuff(int x) throws Exception { }
9.
        8. }
        9. public class Weighty extends Heavy {
10.             10.   // void doStuff(int x) { }
11.
12.             11.   // String doStuff(int x) { return "hi"; }
13.
14.             12.   // public int doStuff(int x) { return 7; }
15.
16.             13.   // private int doStuff(char c) throws Error { return 1; }
17.
18. }
```

Which method(s), if uncommented independently, compile? (Choose all that apply.)

A. Line 6 --

B. Line 7

C. Line 10 --

D. Line 11

E. Line 12

F. Line 13 –

Answer A, C, F

33. Which are true? (Choose all that apply.)

A. A given TreeSet's ordering cannot be changed once it's created.

B. The java.util.Properties class is conceptually more like a List than like a Map.

C. It's a reasonable design choice to use a LinkedList when you want to design queue-like functionality.

D. Of the main types of collections flavors (Lists, Sets, Maps), Queues are conceptually most like Sets.

E. It's programmatically easier to perform a non-destructive traversal of a PriorityQueue than a LinkedList.

F. Classes that implement the Set interface are usually well suited for applications that require access to a collection based on an index.

Answer A, C , F

34. Given the following directory structure:

```
test - |
     | - Finder.class
     | - testdir -|
                  | - subdir
                  | - subdir2
                  | - testfile.txt
```

If test, testdir, subdir, and subdir2 are all directories, and Finder.class and testfile.txt are files, and given:

```
import java.io.*;
public class Finder {
  public static void main(String[] args) throws IOException {
    String[] files = new String[100];
    File dir = new File(args[0]);
    files = dir.list();
System.out.println(files.length); }}
```

And, if the code compiles, the invocation:
java Finder testdir

What is the result?

A. 1

B. 2

C. 3

D. 4

E. 5

F. 100

G. Compilation fails.

Answer H. An exception is thrown at runtime.

35. Given:

```
1. public class Grids {
2.   public static void main(String[] args) {
3. } }
4. int [][] ia2; --
5. int [] ia1 = {1,2,3};--
6. Object o = ia1;
7. ia2 = new int[3][3];
8. ia2[0] = (int[])o;
```

9. ia2[0][0] = (int[])o;

What is the result? (Choose all that apply.)

A. Compilation fails due to an error on line 4.

B. Compilation fails due to an error on line 5.

C. Compilation fails due to an error on line 6.

D. Compilation fails due to an error on line 7.

E. Compilation fails due to an error on line 8.

F. Compilation succeeds and the code runs without exception.

G. Compilation succeeds and an exception is thrown at runtime.

Answer – A,B,C,D,E

PROGRAMS

1.

Whenever a customer details is updated, notify the client objects. At least one client will save that change in the DB and another client object may just have some simple processing.

Implement this scenario using Observer pattern.

2.

Debbie is a statistician who deals with numbers stored in files. She would like a program to help her calculate the mean, median and mode of the numbers in a given file.

Write a program that takes as an input the fully qualified path to a file containing the input data.

The program should prompt for one of the following execution options and perform the corresponding action:

– Calculate the mean
- Calculate the median
- Calculate the mode

3.

Given an array of ints, return true if the sequence of numbers 1, 2, 3 appears in the array somewhere.

array123([1, 1, 2, 3, 1]) → true
array123([1, 1, 2, 4, 1]) → false
array123([1, 1, 2, 1, 2, 3]) → true

```
package com.Programs;

import java.util.Scanner;

public class program_3 {

	public static boolean array123(int[] abc) {
		for (int i = 0; i < (abc.length - 2); i++) {
			if (abc[i] == 1 && abc[i + 1] == 2 && abc[i + 2] == 3)
				return true;
		}
		return false;
	}

	public static void main(String[] args) {

		System.out.println("Program 3");
		@SuppressWarnings("resource")
		Scanner input = new Scanner(System.in);
		System.out.println("How many numbers do you want to enter?");
		int num = input.nextInt();

		int array[] = new int[num];

		System.out.println("Enter the " + num + " numbers now");

		for (int i = 0; i < array.length; i++) {
			array[i] = input.nextInt();
		}
		Boolean b = array123(array);
		System.out.println("Output ::" + b);
	}

}
```

4.

Given a string and an int n, return a string made of the first n characters of the string, followed by the first n-1 characters of the string, and so on. You may assume that n is between 0 and the length of the string, inclusive (i.e. n >= 0 and n <= str.length()).

repeatFront("Chocolate", 4) → "ChocChoChC"
repeatFront("Chocolate", 3) → "ChoChC"

repeatFront("Ice Cream", 2) → "IcI"

```
package com.Programs;

import java.util.Scanner;

public class Program_4 {

	public static String method(String str, int num) {
		// int len = str.length();

		String Word = "";

		for (int i = num; num > 0; num--) {

			Word += str.substring(0, num);
		}
		return Word;
	}

	public static void main(String[] args) {

		System.out.println("program-4");
		Scanner scanner = new Scanner(System.in);
		System.out.println("Enter The string");
		String str = scanner.next();
		System.out.println("Enter the integer");
		int num = scanner.nextInt();
		System.out.println(str + " " + num);
		String str1 = method(str, num);
		System.out.println("Output is ::" + str1);

	}

}
```

5.

The "key" array is an array containing the correct answers to an exam, like {"a", "a", "b", "b"}. the "answers" array contains a student's answers, with "?" representing a question left blank. The two arrays are not empty and are the same length. Return the score for this array of answers, giving +4 for each correct answer, -1 for each incorrect answer, and +0 for each blank answer.

scoreUp(["a", "a", "b", "b"], ["a", "c", "b", "c"]) → 6
scoreUp(["a", "a", "b", "b"], ["a", "a", "b", "c"]) → 11
scoreUp(["a", "a", "b", "b"], ["a", "a", "b", "b"]) → 16

```java
package com.Programs;

import java.util.Scanner;

public class Program_5 {

	public static int score(String[] key, String[] answer) {
		int score = 0;

		for (int i = 0; i < key.length; i++) {
			if (key[i].charAt(0) == answer[i].charAt(0)) {
				score += 4;
			} else if (answer[i].charAt(0) != '?') {
				score -= 1;
			}
		}

		return score;
	}

	@SuppressWarnings("resource")
	public static void main(String[] args) {

		System.out.println("program-5");
		Scanner scanner = new Scanner(System.in);
		int num = 4;
		System.out.println("Key array and answer array should be same");
		System.out.println("enter the 4 key array elements");

		String[] str = new String[num];
		for (int i = 0; i < str.length; i++) {
			str[i] = scanner.nextLine();
		}
		System.out.println("enter the 4 answer array elements");

		String[] str1 = new String[num];
		for (int i = 0; i < str.length; i++) {
			str1[i] = scanner.nextLine();
		}

		int score = score(str, str1);
		System.out.println("Output is ::" + score);
	}

}
```

6.

This class tests the Worker class. It invokes the run() method of a Worker
* instance.

The Worker class holds an instance of a queue of Order objects.

The max size of this queue is 5. 10 Orders are generated asynchronously and placed in the queue.

Those 10 Orders are processed concurrently as they are placed in the queue. Any time we are waiting for Orders to be FULFILLED, this should be printed to standard output. An Order is simply a class with a state field and an order number (1-10). An Order can be in either NEW or FULFILLED states. When a new Order is created, the default state is NEW. Whenever there is a state change, this is printed to standard output. The test ends when all Orders are FULFILLED.

7.

A zero-indexed array A consisting of N integers is given. Rotation of the array means that each element is shifted right by one index, and the last element of the array is also moved to the first place.

For example, the rotation of array A = [3, 8, 9, 7, 6] is [6, 3, 8, 9, 7]. The goal is to rotate array A K times; that is, each element of A will be shifted to the right by K indexes.

Write a function:

```
struct Results solution(int A[], int N, int K);
```

that, given a zero-indexed array A consisting of N integers and an integer K, returns the array A rotated K times.

For example, given array A = [3, 8, 9, 7, 6] and K = 3, the function should return [9, 7, 6, 3, 8].

Assume that:

- N and K are integers within the range [0..100];
- each element of array A is an integer within the range [−1,000..1,000].

In your solution, focus on **correctness**. The performance of your solution will not be the focus of the assessment.

```java
package com.Programs;

import java.util.Scanner;

public class Program_7 {

	public static int[] rotation(int[] arr, int N, int K) {

		int size = arr.length;
		int[] empty_array = null;
		int[] return_array = new int[N];
```

```
		if (K < 0 || K > 100 || size == 0) {
			return empty_array;
		}

		if (size == 1) {
			return arr;
		}

		for (int i = 0; i < size; i++) {
			return_array[(i + K) % size] = arr[i];
		}

		return return_array;
	}

	public static void main(String[] args) {
		System.out.println("Program-7");
		Scanner scanner = new Scanner(System.in);
		System.out.println("Enter the number of Integers N");
		int num = scanner.nextInt();
		System.out.println("Enter the number of Integers K for rotation");
		int num1 = scanner.nextInt();
		int[] arr = new int[num];
		System.out.println("Enter the array elements of size N ");
		for (int i = 0; i < arr.length; i++) {
			arr[i] = scanner.nextInt();
		}
		int[] result = new int[num];
		result = rotation(arr, num, num1);
		System.out.println("Output array is ::");
		for (int i = 0; i < result.length; i++) {
			System.out.print(result[i] + " ");

		}

	}

}
```

8.

Write a function:

```
class Solution { public int solution(int[] A); }
```

that, given a non-empty zero-indexed array A of N integers, returns the minimal positive integer (greater than 0) that does not occur in A.

For example, given:

```
A[0] = 1
A[1] = 3
A[2] = 6
A[3] = 4
A[4] = 1
A[5] = 2
```

the function should return 5.

Assume that:

- N is an integer within the range [1..100,000];
- each element of array A is an integer within the range [−2,147,483,648..2,147,483,647].

Complexity:

- expected worst-case time complexity is O(N);
- expected worst-case space complexity is O(N), beyond input storage (not counting the storage required for input arguments).

Elements of input arrays can be modified.

```java
package com.Programs;

import java.util.HashSet;

public class Program_8 {

	public static int solution(int[] A) {
		int max = Integer.MIN_VALUE;

		// Return 1 when array size is zero
		if (A.length == 0) {
			return 1;
		}

		// Use the HashSet to store existence of number in given array
		HashSet<Integer> hashSet = new HashSet<Integer>();
		for (int i = 0; i < A.length; i++) {
			if (A[i] > 0) {
				hashSet.add(A[i]);
			}

			if (A[i] > max) {
				max = A[i];
			}
		}

		// Return 1 when the maximum number in array is negative value
		if (max < 0) {
			return 1;
		}
```

```
            // Return the minimal positive integer that does not occur in A
            for (int i = 1; i <= max; i++) {
                  if (hashSet.contains(i) == false) {
                        return i;
                  }
            }

            return max + 1;
      }

      public static void main(String[] args) {
            int[] A = { -1, -3, -6, 4, };
            System.out.println("solution " + solution(A));
      }

}
```

9.

A non-empty zero-indexed array A consisting of N integers is given. The consecutive elements of array A represent consecutive cars on a road.

Array A contains only 0s and/or 1s:

- 0 represents a car traveling east,
- 1 represents a car traveling west.

The goal is to count passing cars. We say that a pair of cars (P, Q), where $0 \leq P < Q < N$, is passing when P is traveling to the east and Q is traveling to the west.

For example, consider array A such that:

```
  A[0] = 0
  A[1] = 1
  A[2] = 0
  A[3] = 1
  A[4] = 1
```

We have five pairs of passing cars: (0, 1), (0, 3), (0, 4), (2, 3), (2, 4).

Write a function:

```
class Solution { public int solution(int[] A); }
```

that, given a non-empty zero-indexed array A of N integers, returns the number of pairs of passing cars.

The function should return −1 if the number of pairs of passing cars exceeds 1,000,000,000.

For example, given:

```
A[0] = 0
A[1] = 1
A[2] = 0
A[3] = 1
A[4] = 1
```
the function should return 5, as explained above.

Assume that:

- N is an integer within the range [1..100,000];
- each element of array A is an integer that can have one of the following values: 0, 1.

Complexity:

- expected worst-case time complexity is O(N);
- expected worst-case space complexity is O(1), beyond input storage (not counting the storage required for input arguments).

Elements of input arrays can be modified.

10.

A non-empty array A consisting of N integers is given.
A *permutation* is a sequence containing each element from 1 to N once, and only once.
For example, array A such that:
```
A[0] = 4
A[1] = 1
A[2] = 3
A[3] = 2
```
is a permutation, but array A such that:
```
A[0] = 4
A[1] = 1
A[2] = 3
```
is not a permutation, because value 2 is missing.
The goal is to check whether array A is a permutation.
Write a function:
```
int solution(int A[], int N);
```
that, given an array A, returns 1 if array A is a permutation and 0 if it is not.
For example, given array A such that:
```
A[0] = 4
A[1] = 1
A[2] = 3
A[3] = 2
```
the function should return 1.
Given array A such that:
```
A[0] = 4
```

```
A[1] = 1
A[2] = 3
```

the function should return 0.

Assume that:

- N is an integer within the range [1..100,000];
- each element of array A is an integer within the range [1..1,000,000,000].

Complexity:

- expected worst-case time complexity is O(N);
- expected worst-case space complexity is O(N) (not counting the storage required for input arguments).

```java
package com.Programs;

public class Program_10 {
	public static final int NOT_PERMUTATION = 0;
	public static final int PERMUTATION = 1;

	public static int solution(int[] A) {

		// write your code in Java SE 8
		int[] mark = new int[A.length + 1];
		int counter = 0;

		for (int i = 0; i < A.length; ++i) {
			int value = A[i];
			if (value >= mark.length) {
				return NOT_PERMUTATION;
			}
			if (mark[value] == 0) {
				mark[value] = 1;
				++counter;
			} else {
				return NOT_PERMUTATION;
			}
		}

		return counter == A.length ? PERMUTATION : NOT_PERMUTATION;
	}

	public static void main(String[] args) {
		int[] A = { 1, 2, 3, 7, 4, 5, 8, 6, 9, 10, 11 };
		int i = solution(A);
		System.out.println("output is :: " + i);
	}

}
```