**Section: MCQs**

1) What will be the output of following code?

**class** Test

{

**public** **static** **void** main(String[] args)

{

 int a[]={1, 2, 3, 4, 5, 6};

int i=a.length-1;

while(i>=0)

{

        System.out.println(a[i]);

        a[i] = a[i] \* a[i - (i + i - i)];

i--;

}

}

}

a)  1 2 3 4 5 6

b)  6 5 4 3 2

**c)  6 5 4 3 2 1**

d) 1 2 3 4 5

2) Which of these methods can be used to convert all characters in a String into a character array?

a) charAt()

b) both getChars() & charAt()

**c) both toCharArray() & getChars()**

d) all of the mentioned

3) Which of the following statements are incorrect?

a) static methods can call other static methods only

b) static methods must only access static data

c) static methods can not refer to this or super in any way

**d) when object of class is declared, each object contains its own copy of static variables**

4) Predict the output of below code.

class A {

private void printName()

{

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

}

}

class B extends A

{

public void printName()

{

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

}

}

public class mcqTest {

public static void main(String args[])

{

B b= new B();

b.printName();

}

}

a) Value-A

**b) Value-B**

c) Value-A Value-B

d) Compilation fails – private methods can’t be override.

5) Which is true ?

a) "X extends Y" is correct if and only if X is a class and Y is an interface

b) "X extends Y" is correct if and only if X is an interface and Y is a class

**c) "X extends Y" is correct if X and Y are either both classes or both interfaces**

d) "X extends Y" is correct for all combinations of X and Y being classes and/or interfaces

6) What is the output of the following program?

class A

{

int a = 50;

}

public class B extends A

{

int a = 100;

void show()

{

System.out.println(super.a);

}

public static void main(String args[])

{

B b = new B();

b.show();

}

}

a) Compile Time Error: Re-Declaration of variable a

b) 50 100

c) 100

**d) 50**

7) What is the output of below program?

public class Test1

{

static void show()

{

this=null;

}

public static void main(String args[])

{

show();

System.out.println("JAVA ST1");

}

}

1. JAVA ST1
2. **Complie Time Error**
3. Run Time Error
4. None of the above.

8) What is the output of following program code?

publicclass mcqTest {

publicstaticvoid main(String args[])

{

try

{

inti;

return;

}

catch(Exception e)

{

System.*out*.print("In Catch Block ");

}

finally

{

System.*out*.print("In Finally Block");

}

}

}

a) In Catch Block In Finally Block

**b) In Finally Block**

c) In Catch Block

d) Program will return without printing anything

9) Which is a correct option about java interface?

a) An interface is used to achieve multiple inheritances in java

b) An object of an interface cannot be created.

c) An interface can extend another interface.

**d) All of the above**

10) What is the output of below program?

class Base {

public final void show() {

System.out.println("Base::show() called");

}

}

class Derived extends Base {

public void show() {

System.out.println("Derived::show() called");

}

}

public class Main {

public static void main(String[] args) {

Base b = new Derived();

b.show();

}

}

a) Derived::show() called

b) Base::show() called

c) Exception

**d) Compiler Error**

11) Determine the output of following java code.

class Small {

public Small()

{

System.out.print("A ");

}

}

class Small2 extends Small

{

public Small2()

{

System.out.print("B ");

}

}

class Small3 extends Small2

{

public Small3()

{

System.out.print("C ");

}

}

public class mcqTest {

public static void main(String args[])

{

new Small3();

}

}

**a) A B C**

b) C B A

c) C

d) A

12) Determine the output of following java code.

class X

{

int i = 202020;

public X()

{

i = i++ + i-- - i;

}

static int staticMethod(int i)

{

return --i;

}

}

class Y extends X

{

public Y()

{

System.out.println(staticMethod(i));

}

}

public class Mcq1

{

public static void main(String[] args)

{

Y y = new Y();

}

}

1. 220220
2. **202020**
3. 202022
4. 202220

13) What will be the output of following code?

abstract class A

{

abstract void firstMethod();

void secondMethod()

{

System.out.println("SECOND");

firstMethod();

}

}

abstract class B extends A

{

void firstMethod()

{

System.out.println("FIRST");

thirdMethod();

}

abstract void thirdMethod();

}

class C extends B

{

void thirdMethod()

{

System.out.println("THIRD");

}

}

public class MainClass

{

public static void main(String[] args)

{

C c = new C();

c.firstMethod();

c.secondMethod();

c.thirdMethod();

}

}

1. Compilation error
2. **FIRST, THIRD, SECOND, FIRST, THIRD, THIRD**
3. THIRD, THIRD, SECOND, FIRST, FIRST, SECOND
4. SECOND, THIRD, THIRD, FIRST, FIRST, SECOND

14) What will be the output of following code?

interface Alice{

public void sender();

}

interface Bob extends Alice{

public void receiver();

}

class Comm implements Bob {

public void sender() {

System.out.println("Data Sender");

}

public void receiver() {

System.out.println("Data Receiver");

}

}

public class McqMain

{

public static void main(String ar[]) {

Comm com = new Comm();

com.sender();

}

}

1. **Data Sender**
2. Data Receiver
3. Data SenderData Receiver
4. None of the above

15) Determine the output of following java code.

class A

{

public void doA()

{

B b=new B();

b.dobB();

System.out.print("doA");

}

}

class B

{

public void dobB()

{

C c=new C();

c.doC();

System.out.print("doB");

}

}

class C

{

public void doC()

{

if(true)

throw new NullPointerException();

System.out.print("doC");

}

}

class McqTest

{

public static void main(String ar[]) {

try

{

A a=new A();

a.doA();

}catch(Exception ex) {

System.out.print("error");

}

}

}

a) doAdoBdoC

b) doCdoBdoA

c)doAdoBdoCerror

**d) error**

**Section: Coding**

**16) AVERAGE COUNT ARRAY**

You are given an integer ' n ' which denote the number of elements in an array a[ ] and an integer ' x '. You have to calculate the average of element a[i] and x and find out if that number exist in array or not. Do it for all the elements of array and store the count result in another array for each index i.

**Input:**

First line contains integer t denoting number of test cases. The next subsequent lines contain numbers ' n ' and ' x '. Elements of array a[ ].

**Output:**

You have to print the resultant array in which you have to calculate the number of times the average of a[i] and x occurs in the array for each index i.

**Constraints:**

0 < t < 100

0 < n < 100

0 < x < 100

0 < a[i] < 100

**Example:**

**Input:**

2

5 2

2 4 8 6 2

6 3

9 5 2 4 0 3

**Output**:

2 0 0 1 2

0 1 1 1 0 1

**Explanation:**

**For eg 1:** value of n is 5 and that of x is 2. The array is 2 4 8 6 2. We take x i.e. 2 and take average with a[0] which is equal to 2. We found 2 resides in array at two positions (1st and 5th element) thus storing 2 in another array at 0th index. Similarly do for all elements and store the count in second array.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2  6 3  6 4 2 5 4 3  6 3  1 2 3 4 5 6 | 3  7 4  11 12 13 14 15 16 17  7 4  7 6 5 7 6 5 7  5 5  5 10 15 10 5 | 1  10 6  6 12 6 9 18 24 6 12 6 9 | 4  3 1  2 3 2  4 10  10 10 10 10  10 20  40 48 50 20 60 40 40 60 68 70  3 8  8 8 8 | 5  1 1  1  2 2  2 2  3 3  3 3 3  4 4  4 4 4 4  5 5  5 5 5 5 5 |
| 2 1 1 2 1 1  1 1 1 1 1 1 | 0 0 0 0 0 0 0  2 2 0 2 2 0 2  2 0 2 0 2 | 4 2 4 0 2 0 4 2 4 0 | 0 2 0  4 4 4 4  0 0 0 1 3 0 0 3 0 0  3 3 3 | 1  2 2  3 3 3  4 4 4 4  5 5 5 5 5 |

**17) *Colored Balls***

Alisha has a box full of N different colored balls. She wants to count the number of each colored ball that she has and also wants to check the maximum number of balls of specific color. If two or more colored balls are same in number, then she needs to order their names alphabetically and wants to tell her friends the last colored ball in the alphabetical order. Her mother is busy in household work. Therefore she seeks your help for the same.

Help Alisha in counting the colored balls that are maximum in number from a box of N number of balls.

For example, colors of the balls are in the names [“Red”,“Blue”,”Red”,”Green”,”Blue”]. There are 2 “Red” balls, 2 Blue balls and 1 Green ball. But Red comes last alphabetically so output will be **Red.**

Sample Input:

5

Red Blue Red Green Blue

Sample Output:

Red

Explanation:

First Line of sample input denotes the total number of balls, N.

Next N values denotes the color of each ball

Sample output represents colored ball i.e. maximum in number in alphabetical order.

Solution:

**import** java.util.\*;

**public** **class** file {

**static** String maxColor(String[] colors,**int** n) {

**int max=0,c=0,flag=0;**

**String fina=colors[0];**

**for(int i=0;i<n;i++)**

**{**

**c=0;**

**flag=0;**

**for(int j=i-1;j>0;j--)**

**{**

**if(colors[i].equals(colors[j]))**

**{**

**flag=1;**

**break;**

**}**

**}**

**if(flag==0)**

**{**

**for(int j=i;j<n;j++)**

**{**

**if(colors[i].equals(colors[j]))**

**{**

**c++;**

**}**

**}**

**if(c==max)**

**{**

**if(colors[i].compareToIgnoreCase(fina)>0)**

**{**

**fina=colors[i];**

**}**

**}**

**if(c>max)**

**{**

**max=c;**

**fina=colors[i];**

**}**

**}**

**}**

**return fina;**

**}**

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

Scanner sc = **new** Scanner(System.*in*);

**int** balls = sc.nextInt();

sc.nextLine();

String[] colors = **new** String[balls];

**for** (**int** i = 0; i < balls; i++) {

String votesItem = sc.nextLine();

colors[i] = votesItem;

}

String res = *maxColor*(colors,balls);

System.*out*.println(res);

}

}

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 5  White  Yellow  White  White  Yellow | 6  Orange  Blue  Orange  Blue  White  Green | 1  Pink | 8  Pink  Purple  Purple  Purple  Purple  Pink  Pink  Pink | 10  White  Orange  Yellow  Brown  Pink  Brown  Blue  Blue  NavyBlue  NavyBlue |
| White | Orange | Pink | Purple | NavyBlue |

18) **Increasing Sequence**

Given an integer array having N elements, Your task is to find the length of the longest Increasing sequence such that every element in that sequence is strictly greater than its previous element in the same sequence. Also, print the elements of that longest sequence. If there are two or more Longest Increasing sequence of same length, then display the first occurring sequence (as shown in Sample Output 2).

Sample Input1:

1

10

11 3 1 4 7 8 12 2 3 7

Sample Output1:

5

1 4 7 8 12

Sample Input 2:

2

10

12 13 1 5 4 7 8 10 10 11

7

5 8 9 7 6 7 8

Sample Output 2:

4

4 7 8 10

3

5 8 9

Explanation:

First line of sample input represents the no. of test cases

Second line of sample input denotes the number of elements, i.e. N.

Third line denotes the array elements.

First Line of sample output denotes the length of the longest Increasing sequence.

Second line displays the elements of that longest sequence.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2  6  6 4 2 5 4 3  6  1 2 3 4 5 6 | 3  7  11 12 13 14 15 16 17  7  7 6 5 7 6 5 7  5  5 10 15 10 5 | 1  10  6 12 6 9 18 24 6 12 6 9 | 4  3  2 3 2  4  10 10 10 10  11  40 48 50 20 60 40 40 60 68 70 75  3  8 8 8 | 5  1  11  2  21 22  3  33 34 35  4  41 42 43 43  5  51 52 53 54 54 |
| 2  2 5  6  1 2 3 4 5 6 | 7  11 12 13 14 15 16 17  2  5 7  3  5 10 15 | 4  6 9 18 24 | 2  2 3  1  10  5  40 60 68 70 75  1  8 | 1  11  2  21 22  3  33 34 35  3  41 42 43  4  51 52 53 54 |

**Solution**:

**import** java.util.\*;

**import** java.math.\*;

**public** **class** IncSeq {

**static** **void** lenOfLongIncSubArr(**int** arr[],**int** n){

**int** max = 1, len = 1, maxIndex = 0;

**for** (**int** i=1; i<n; i++)

{

**if** (arr[i] > arr[i-1])

len++;

**else**

{

**if** (max < len)

{ max = len;

maxIndex = i-max;

}

len = 1;

}

}

**if** (max < len)

{

max = len;

maxIndex=n-max;

}

System.***out***.println(max);

**for** (**int** i = maxIndex; i < max+maxIndex; i++)

System.***out***.print(arr[i] + " ");

System.***out***.println();

}

**public** **static** **void** main(String[] args)

{

Scanner sc=**new** Scanner(System.***in***);

**int** t=sc.nextInt();

**for**(**int** i=0;i<t;i++)

{

**int** n=sc.nextInt();

**int** arr[] = **new** **int**[n];

**for**(**int** j=0;j<n;j++)

arr[j]=sc.nextInt();

*lenOfLongIncSubArr*(arr, arr.length);

}

}

}