**SET B CORE JAVA (CSL 5303) RETEST ST1**

1. Which is a perfect example of runtime polymorphism?
2. Method overloading
3. **Method overriding**
4. Constructor overloading
5. None of the above
6. What will be the output of the program?

public class Mcq1{

void Mcq1() //Line 3

{

System.out.println("Class Mcq1");

}

public static void main(String[] args) {

new Mcq1();

}

}

1. Class Mcq1
2. Compilation fails.
3. An exception is thrown at Line 3
4. **The code executes with no output.**
5. Which component is used to compile, debug and execute java program?
6. JVM
7. **JDK**
8. JIT
9. JRE
10. Which of these cannot be declared static?
11. Class
12. **Object**
13. Variable
14. method
15. Which cause a compiler error?
16. int[ ] scores = {3, 5, 7};
17. String cats[ ] = {"Fluffy", "Spot", "Zeus"};
18. boolean results[ ] = new boolean [] {true, false, true};
19. **int [ ][ ] scores = {2,7,6}, {9,3,45};**
20. Predict the output:

**public** **class** Mcq1 {

**public** **int** aMethod() {

**static** **int** i = 0;

i++;

**return** i;

}

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

Test test = **new** Test();

test.aMethod();

**int** j = test.aMethod();

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

}

}

1. 0
2. 1
3. **Compilation fails**
4. 2
5. Predict the output?

|  |
| --- |
| class Base{  Base() {  System.out.print("Base ");  }  }  public class Alpha extends Base{  public static void main(String[] args) {  new Alpha();  new Base();  }  } |

1. Base
2. Complier error
3. **Base Base**
4. The code executes with no output.
5. What will be the output of the program?

public class MyProgram {

public static void main(String args[]) {

try {

System.out.print("Hello world ");

}

finally {

System.out.println("Finally executing ");

}

}

}

1. The program will not compile because no exceptions are specified
2. The program will not compile because no catch clauses are specified.
3. Hello world.
4. **Hello world Finally executing**
5. Which of the following method call gives the position of ‘X’ that occurs after nth position in the string S1.

|  |
| --- |
| 1. S1.indexOf(‘X’) 2. S1.index(‘X’, 1) 3. **S1.indexOf(‘X’ , n)** 4. S1.index(‘X’, n) |

1. Which of the following are incorrect form of StringBuffer class constructor?  
   **a) StringBuffer(int size , String str)**b) StringBuffer()  
   c) StringBuffer(int size)  
   d) StringBuffer(String str)
2. What is the output of this program?

public class Mcq1{

public static void aMethod() throws Exception {

try {

throw new Exception();

}

finally {

System.out.print("finally ");

}

}

public static void main(String args[]) {

try {

aMethod();

}

catch (Exception e) {

System.out.print("exception ");

}

System.out.print("finished");

}

}

**a) finally exception finished**b) finally  
c) exception finished  
d) finished

1. What is the output of this program?

class jump\_statments {

public static void main(String args[]) {

int x = 2;

int y = 0;

for ( ; y < 10; ++y) {

if (y % x == 0)

continue;

else if (y == 8)

break;

else

System.out.print(y + " ");

}

}

}

1. 1 3 5 7
2. 2 4 6 8
3. **1 3 5 7 9**
4. 1 2 3 4 5 6 7 8 9
5. What is the output of the following program?

|  |
| --- |
| class Base {      public static String s = " Super Class ";      public Base()      {          System.out.printf("1");      }  }  public class Derived extends Base {      public Derived()     {          System.out.printf("2");          super();      }   public static void main(String[] args)     {          Derived obj = new Derived();          System.out.printf(s);      }  } |

1. 21 Super Class
2. Super Class 21
3. **Compilation error**
4. 12 Super Class
5. Output of following java program.

// Main.java

public class Main {

    public static void main(String args[])     {

        String s1 = "abc";

        String s2 = s1;

        s1 += "d";

        System.out.println(s1 + " " + s2 + " " + (s1 == s2));

        StringBuffer sb1 = new StringBuffer("abc");

        StringBuffer sb2 = sb1;

        sb1.append("d");

        System.out.println(sb1 + " " + sb2 + " " + (sb1 == sb2));

    }

}

1. **abcd abc false**

**abcd abcd true**

1. abcd abcd true

abcd abc false

1. abcd abcd true

abcd abc true

1. abcd abc true

abcd abcd true

1. Output of following java program.

**public** **class** Mcq1 {

**public** Mcq1() {

System.***out***.printf("1");

**new** Mcq1(10);

System.***out***.printf("5");

}

**public** Mcq1(**int** temp) {

System.***out***.printf("2");

**new** Mcq1(10, 20);

System.***out***.printf("4");

}

**public** Mcq1(**int** data, **int** temp) {

System.***out***.printf("3");

}

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

Mcq1 obj = **new** Mcq1();

}

}

1. **12345**
2. 123
3. 12354
4. 54

***Coding:***

1. ***WordCount***

killCode has got java fever. He capitalizes every first letter of the word except the first word. His teacher told him to write an essay and he wrote it all in java style. Your job is to count the total number of words he wrote in the essay.

Every string contains only characters from a-z and A-Z.

***Sample Input:***

3

helloJava

hi

wishingYou,goodMorning

***Sample Output:***

2

1

3

***Input Format:***

1. First line of input contains t denoting the number of test cases.
2. Next t contains string s representing the essay writing by killCode

***Output Format:***

Print the number of words in the essay writen by killCode for each case.

***Code Constraints:***

1 ≤ t ≤ 10

1 ≤ s ≤ 105

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2  youAreBeautiful  hi,IAmAStudent | 3  thisIsCompleted.  hello,GoodEvening  iAmALecturer | 2  hello,MyNameIsCoder.  thisIsVeryEasyQuestion. | 4  thisIsNewCode.  multiLineComments  goodStudentsDon'tCheat  comment// | 3  \_helloWorld  multi-threadingIsAnIntrestingTopic  keepItUp |
| 3  5 | 3  3  4 | 5  5 | 4  3  4  1 | 2  5  3 |

**Solution:**

**import** java.util.Scanner;

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

**static** **int** wordCount(String eassy)

{

**int** count=1;

**for**(**int** j=0; j < eassy.length();j++ ){

**if**(eassy.charAt(j) >= 65 && eassy.charAt(j) <= 90){

count++;

}

}

**return** count;

}

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

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

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

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

String eassy = sc.next();

**int** res=*wordCount*(eassy);

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

}

}

}

1. ***Array elements and Sum***

Given an array A of n numbers and another number x, determines whether or not there exist a pair in A whose sum is exactly x.

If pair is found in the array having sum equal to x then print pair of element on screen. If more than one pair is found then print pair which is having smallest element of array.

***Pair will always contain 1st element smaller and 2nd element larger.***

***Input***:

First line denotes the T testcases.

Second Line denotes N, number of elements in array

Next Line represents N array elements.

Last line denotes input number x.

***Output***:

Output the pair of elements according to the required format in new line.

***Sample Input***

6 //number of elements in array

1

4

45

6

10

-8

16 //number x

***Sample Output***

(6,10)

***Explanation***

As we have to find 16, there is one pair 6+10=16. So it print pair (6,10) on screen. Pair will always contain 1st smaller element and 2nd large element to make sum 16.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2  5  1  2  3  4  5  5  5  -2  -3  -1  0  1  -5 | 2  3  3  4  2  7  4  2  1  2  3  4 | 1  6  3  4  5  2  1  6  7 | 1  2  4  5  9 | 1  5  -3  -1  0  5  2  1 |
| (1,4)  (-3,-2) | (3,4)  (1,3) | (1,6) | (4,5) | (-1,2) |

**SOLUTION:**

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

**public** **class** PairSum {

**static** **void** hasArrayTwoCandidates(**int** A[],**int** arr\_size, **int** sum)

{

**int** l, r;

Arrays.*sort*(A);

l = 0;

r = arr\_size-1;

**while** (l < r)

{

**if**(A[l] + A[r] == sum)

{

System.***out***.println("("+A[l]+","+A[r]+")");

**break**;

}

**else** **if**(A[l] + A[r] < sum)

l++;

**else** // A[i] + A[j] > sum

r--;

}

}

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

{

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

**int** t,n,x;

t=sc.nextInt();

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

{

n=sc.nextInt();

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

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

A[j]=sc.nextInt();

x=sc.nextInt();

*hasArrayTwoCandidates*(A,n,x);

}

}

}

1. ***Count Substrings***

Given a string S consisting of only 1s and 0s, find the number of substrings which start and end both in 1.

In this problem, a substring is defined as a sequence of continuous characters Si, Si+1, ..., Sj where 1 ≤ i ≤ j ≤ N.

***Sample Input:***

2

4

1111

5

10001

***Sample Output:***

10

3

***Explanation:***

#test1: All substrings satisfy.

#test2: Three substrings S[1,1], S[5,5] and S[1,5] satisfy.

***Input Format:***

First line contains T, the number of testcases. Each testcase consists of N (the length of string) in one line and string in second line.

***Output Format:***

For each testcase, print the required answer in one line.

***Constraint***

1 ≤ T ≤ 105

1 ≤ N ≤ 105

Sum of N over all testcases ≤ 105

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2  3  100  5  10101 | 2  4  1011  5  11111 | 2  16  1111111111111111  6  010101 | 2  4  0000  15  010111111011010 | 1  6  000011 |
| 1  6 | 6  15 | 136  6 | 0  55 | 3 |

**Solution**:

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

**import** java.lang.\*;

**import** java.io.\*;

**public** **class** file

{

**static** **long** stringOnes(String number,**long** len)

{

**long** count=0;

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

**if**(number.charAt(j)=='1')

count++;

**long** res=count\*(count+1)/2;

**return** res;

}

**public** **static** **void** main (String[] args) **throws** java.lang.Exception

{

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

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

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

{

**long** len=sc.nextLong();

String number=sc.next();

System.***out***.println(*stringOnes*(number,len));

}

}

}