SAVEETHA SCHOOL OF ENGINEERING

SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES

INSTITUTE OF PLACEMENT AND TRAINING

CSA09 –JAVA PROGRAMMING

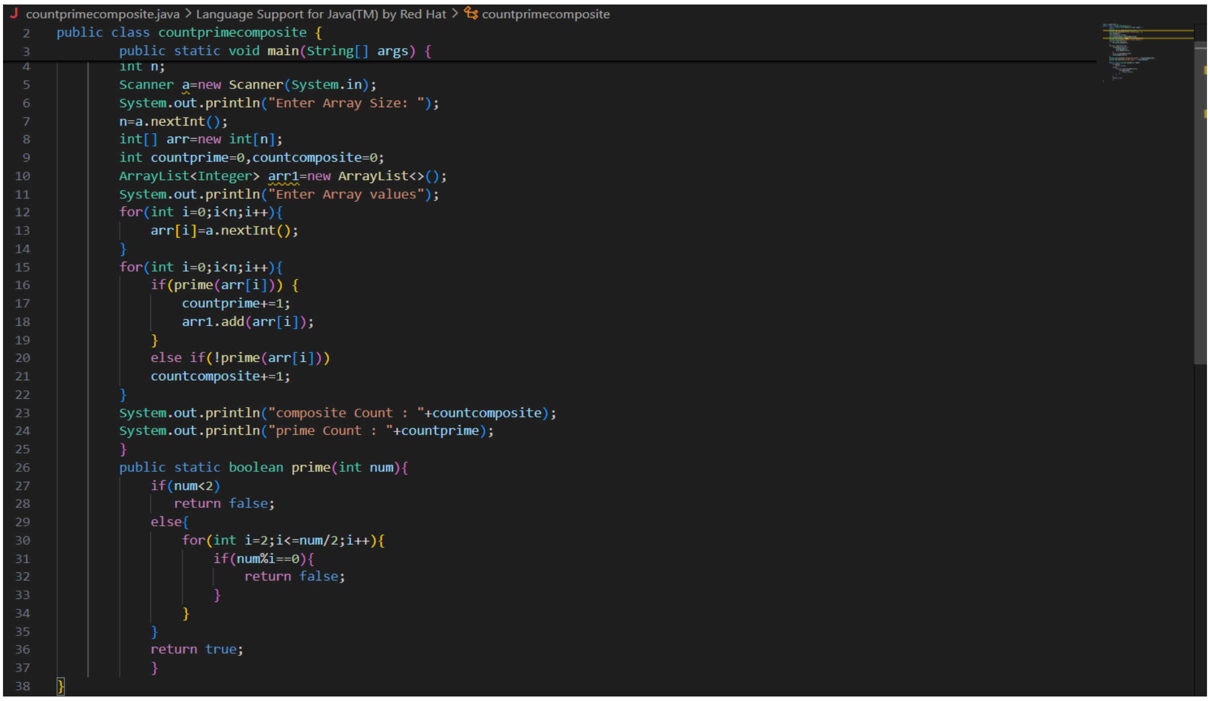
Name:V.Balaram Krishna
Reg.No:192324049

1. Write a program to count all the prime and composite numbers entered by the user.

Sample Input:
Enter the numbers
4
54
29
71
7
59
98
23
Sample Output:
Composite number:3
Prime number:5
Test cases:

1. 33, 41, 52, 61,73,90
2. TEN, FIFTY, SIXTY-ONE, SEVENTY-SEVEN, NINE
3. 45, 87, 09, 5.0 ,2.3, 0.4
4. -54, -76, -97, -23, -33, -98
5. 45, 73, 00, 50, 67, 44

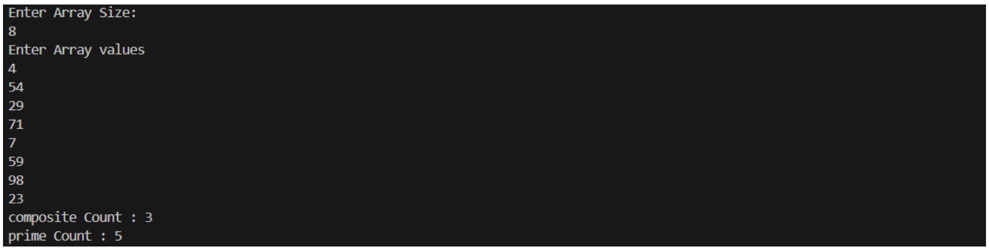
CODE:



countprimecomposite.java > Language Support for Java(TM) by Red Hat > countprimecomposite

```java
public class countprimecomposite {
        public static void main(String[] args) {
        int n;
        Scanner a=new Scanner(System.in);
        System.out.println("Enter Array Size: ");
        n=a.nextInt();
        int[] arr=new int[n];
        int countprime=0,countcomposite=0;
        ArrayList<Integer> arr1=new ArrayList<>();
        System.out.println("Enter Array values");
        for(int i=0;i<n;i++){
            arr[i]=a.nextInt();
        }
        for(int i=0;i<n;i++){
            if(prime(arr[i])) {
                countprime+=1;
                arr1.add(arr[i]);
            }
            else if(!prime(arr[i]))
            countcomposite+=1;
        }
        System.out.println("composite Count : "+countcomposite);
        System.out.println("prime Count : "+countprime);
        }
        public static boolean prime(int num){
            if(num<2)
              return false;
            else{
                for(int i=2;i<=num/2;i++){
                    if(num%i==0){
                        return false;
                    }
                }
            }
            return true;
            }
}
```

OUTPUT:



```
Enter Array Size:
8
Enter Array values
4
54
29
71
7
59
98
23
composite Count : 3
prime Count : 5
```

2. Find the Mth maximum number and Nth minimum number in an array and then find the sum of it and difference of it.

   Sample Input:

   Array of elements = {14, 16, 87, 36, 25, 89, 34}

   M = 1

   N = 3

   Sample Output:

   1stMaximum Number = 89

   3rdMinimum Number = 25

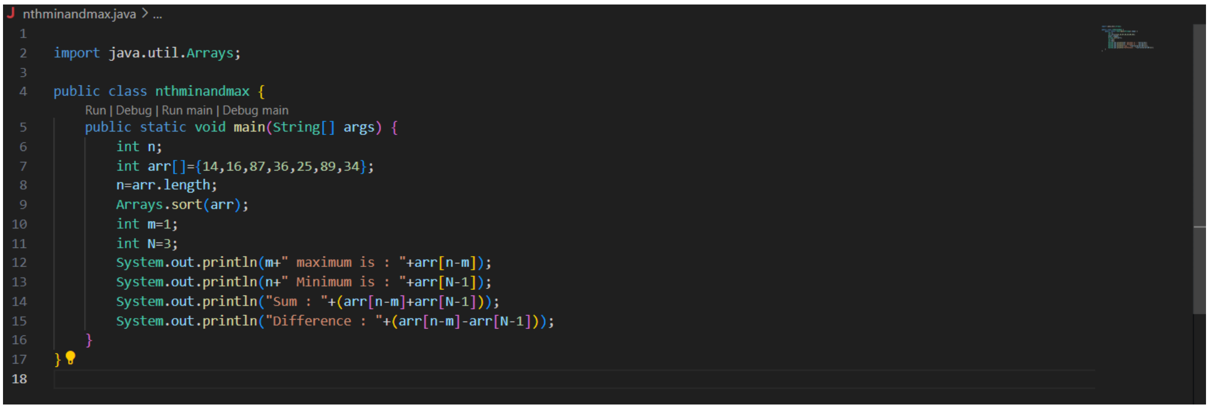
   Sum = 114

   Difference = 64

Test cases:

1. {16, 16, 16 16, 16}, M = 0, N = 1
2. {0, 0, 0, 0}, M = 1, N = 2
3. {-12, -78, -35, -42, -85}, M = 3 , N = 3
4. {15, 19, 34, 56, 12}, M = 6 , N = 3
5. {85, 45, 65, 75, 95}, M = 5 , N = 7

CODE:



nthminandmax.java > ...

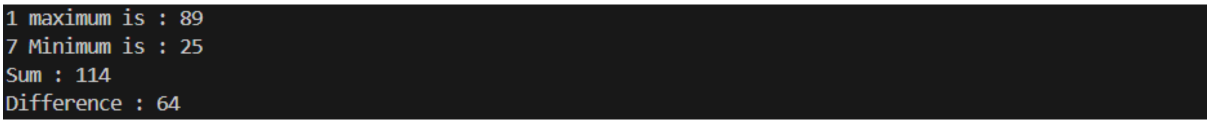
```java

import java.util.Arrays;

public class nthminandmax {
    public static void main(String[] args) {
        int n;
        int arr[]={14,16,87,36,25,89,34};
        n=arr.length;
        Arrays.sort(arr);
        int m=1;
        int N=3;
        System.out.println(m+" maximum is : "+arr[n-m]);
        System.out.println(n+" Minimum is : "+arr[N-1]);
        System.out.println("Sum : "+(arr[n-m]+arr[N-1]));
        System.out.println("Difference : "+(arr[n-m]-arr[N-1]));
    }
}

```

OUTPUT:



```
1 maximum is : 89
7 Minimum is : 25
Sum : 114
Difference : 64
```

3. Write a program to print the total amount available in the ATM machine with the conditions applied.

Total denominations are 2000, 500, 200, 100, get the denomination priority from the user and the total number of notes from the user to display the total available balance to the user

Sample Input:

Enter the 1st Denomination: 500

Enter the 1st Denomination number of notes: 4

Enter the 2nd Denomination: 100

Enter the 2nd Denomination number of notes: 20

Enter the 3rd Denomination: 200

Enter the 3rd Denomination number of notes: 32

Enter the 4th Denomination: 2000

Enter the 4th Denomination number of notes: 1

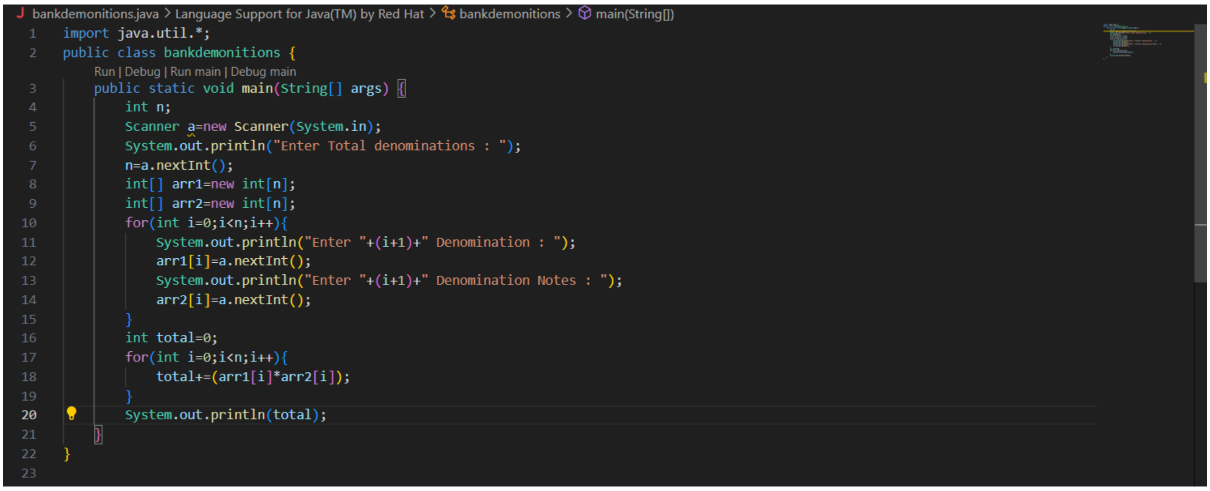
Sample Output:

Total Available Balance in ATM: 12400

Test Cases:

3 Hidden Test cases (Think Accordingly based on Denominations)

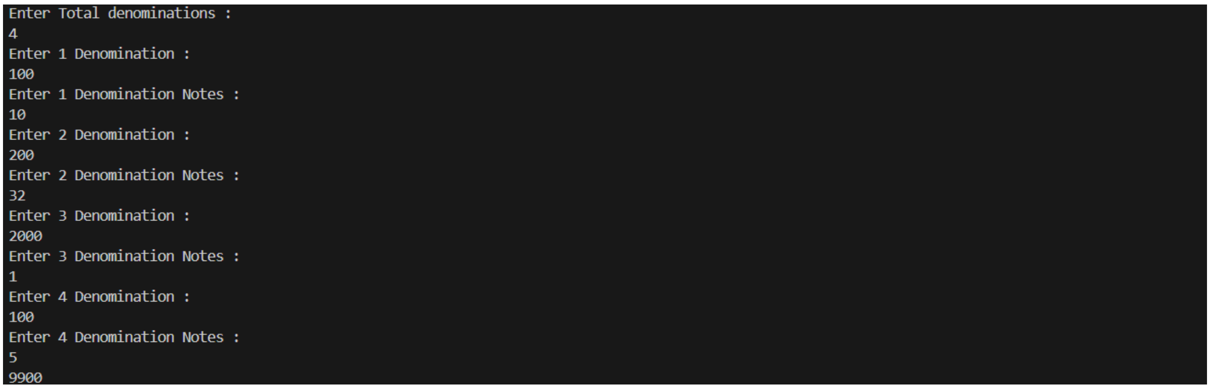
CODE:



bankdemonitions.java > Language Support for Java(TM) by Red Hat > bankdemonitions > main(String[])
Run | Debug | Run main | Debug main

```java
import java.util.*;
public class bankdemonitions {
    public static void main(String[] args) {
        int n;
        Scanner a=new Scanner(System.in);
        System.out.println("Enter Total denominations : ");
        n=a.nextInt();
        int[] arr1=new int[n];
        int[] arr2=new int[n];
        for(int i=0;i<n;i++){
            System.out.println("Enter "+(i+1)+" Denomination : ");
            arr1[i]=a.nextInt();
            System.out.println("Enter "+(i+1)+" Denomination Notes : ");
            arr2[i]=a.nextInt();
        }
        int total=0;
        for(int i=0;i<n;i++){
            total+=(arr1[i]*arr2[i]);
        }
        System.out.println(total);
    }
}
```

OUTPUT:



```
Enter Total denominations :
4
Enter 1 Denomination :
100
Enter 1 Denomination Notes :
10
Enter 2 Denomination :
200
Enter 2 Denomination Notes :
32
Enter 3 Denomination :
2000
Enter 3 Denomination Notes :
1
Enter 4 Denomination :
100
Enter 4 Denomination Notes :
5
9900
```

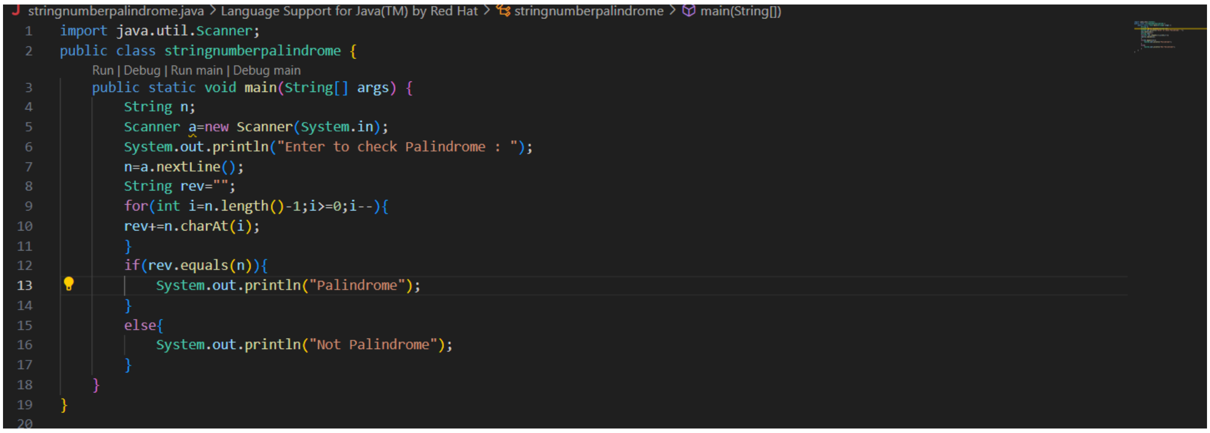
4. Write a program using choice to check

    Case 1: Given string is palindrome or not

    Case 2: Given number is palindrome or not

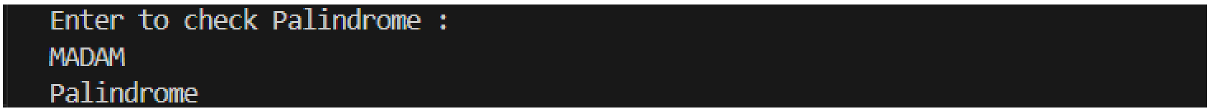
Sample Input:
Case = 1
String = MADAM
Sample Output:
Palindrome
Test cases:
1. MONEY
2. 5678765
3. MALAY12321ALAM
4. MALAYALAM
5. 1234.4321

CODE:


stringnumberpalindrome.java > Language Support for Java(TM) by Red Hat > stringnumberpalindrome > main(String[])
Run | Debug | Run main | Debug main

```java
import java.util.Scanner;
public class stringnumberpalindrome {
    public static void main(String[] args) {
        String n;
        Scanner a=new Scanner(System.in);
        System.out.println("Enter to check Palindrome : ");
        n=a.nextLine();
        String rev="";
        for(int i=n.length()-1;i>=0;i--){
        rev+=n.charAt(i);
        }
        if(rev.equals(n)){
            System.out.println("Palindrome");
        }
        else{
            System.out.println("Not Palindrome");
        }
    }
}
```

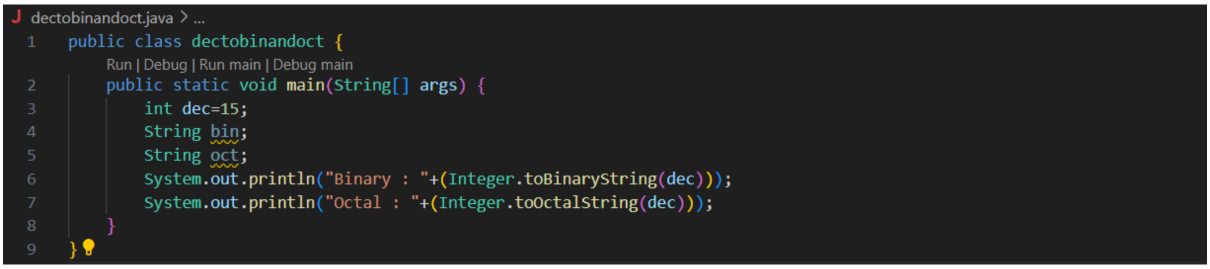
OUTPUT:



```
Enter to check Palindrome :
MADAM
Palindrome
```

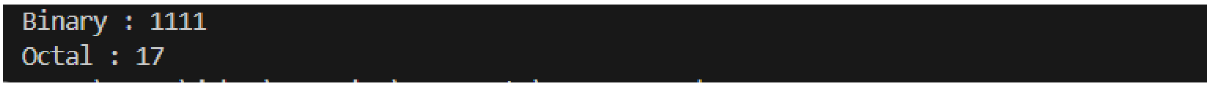
5. Write a program to convert Decimal number equivalent to Binary number and octal numbers?
   Sample Input:
   Decimal Number: 15
   Sample Output:
   Binary Number = 1111
   Octal = 17
   Test cases:
   1. 111
   2. 15.2
   3. 0
   4. B12
   5. 1A.2

   CODE:


dectobinandoct.java > ...
Run | Debug | Run main | Debug main

```java
public class dectobinandoct {
    public static void main(String[] args) {
        int dec=15;
        String bin;
        String oct;
        System.out.println("Binary : "+(Integer.toBinaryString(dec)));
        System.out.println("Octal : "+(Integer.toOctalString(dec)));
    }
}
```

OUTPUT:



```
Binary : 1111
Octal : 17
```

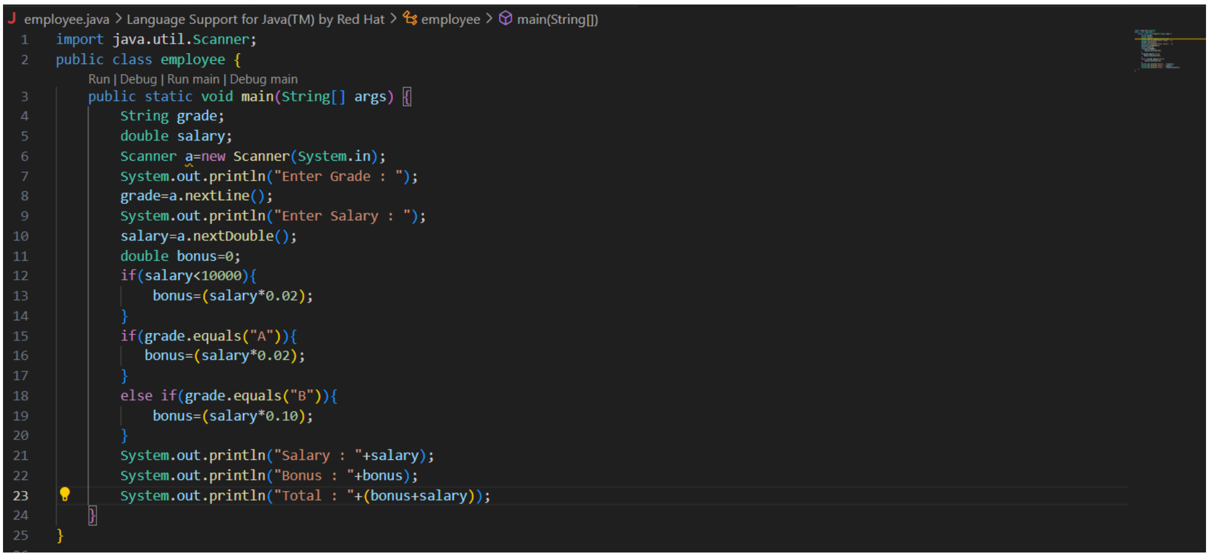
6. In an organization they decide to give bonus to all the employees on New Year. A 5% bonus on salary is given to the grade A workers and 10% bonus on salary to the grade B workers. Write a program to enter the salary and grade of the employee. If the salary of the employee is less than $10,000 then the employee gets an extra 2% bonus on salary Calculate the bonus that has to be given to the employee and print the salary that the employee will get.

    Sample Input & Output:
    Enter the grade of the employee: B
    Enter the employee salary: 50000
    Salary=50000
    Bonus=5000.0
    Total to be paid:55000.0

Test cases:

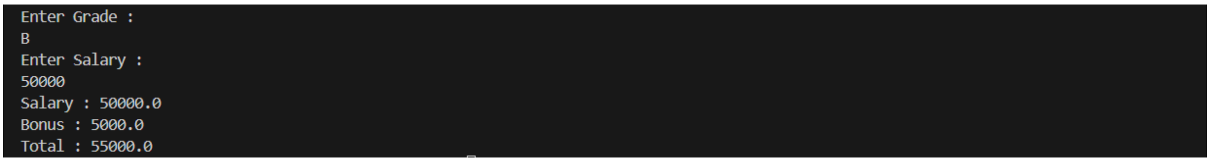
1. Enter the grade of the employee: A
   Enter the employee salary: 8000
2. Enter the grade of the employee: C
   Enter the employee salary: 60000
3. Enter the grade of the employee: B
   Enter the employee salary: 0
4. Enter the grade of the employee: 38000
   Enter the employee salary: A
5. Enter the grade of the employee: B
   Enter the employee salary: -8000

CODE:


employee.java > Language Support for Java(TM) by Red Hat > employee > main(String[])
Run | Debug | Run main | Debug main

```java
import java.util.Scanner;
public class employee {
    public static void main(String[] args) {
        String grade;
        double salary;
        Scanner a=new Scanner(System.in);
        System.out.println("Enter Grade : ");
        grade=a.nextLine();
        System.out.println("Enter Salary : ");
        salary=a.nextDouble();
        double bonus=0;
        if(salary<10000){
            bonus=(salary*0.02);
        }
        if(grade.equals("A")){
            bonus=(salary*0.02);
        }
        else if(grade.equals("B")){
            bonus=(salary*0.10);
        }
        System.out.println("Salary : "+salary);
        System.out.println("Bonus : "+bonus);
        System.out.println("Total : "+(bonus+salary));
    }
}
```

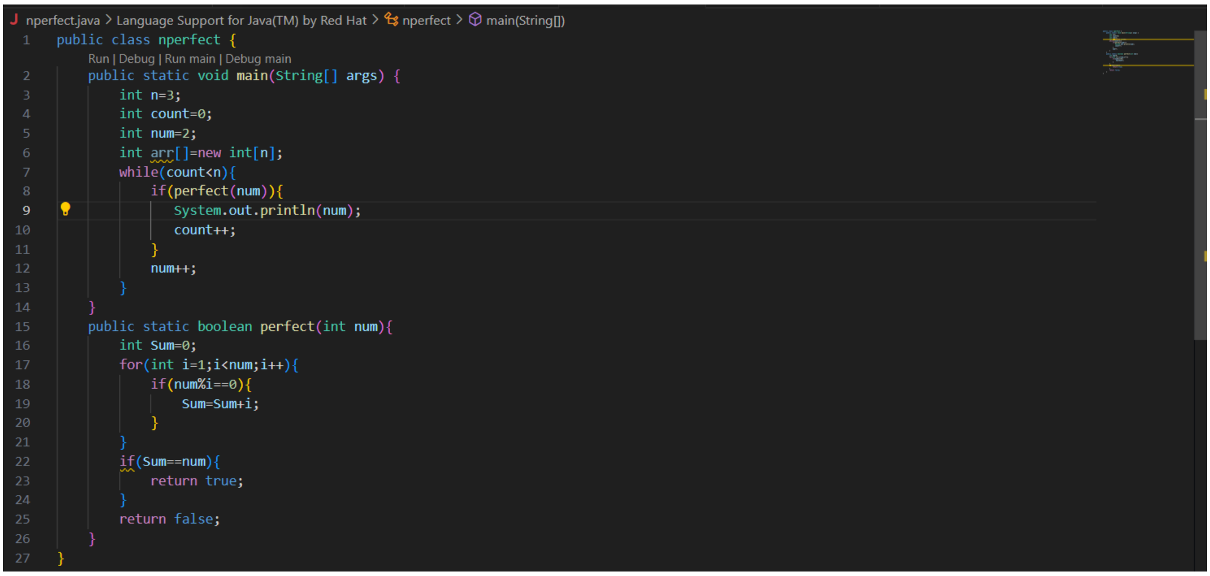
OUTPUT:



```
Enter Grade :
B
Enter Salary :
50000
Salary : 50000.0
Bonus : 5000.0
Total : 55000.0
```

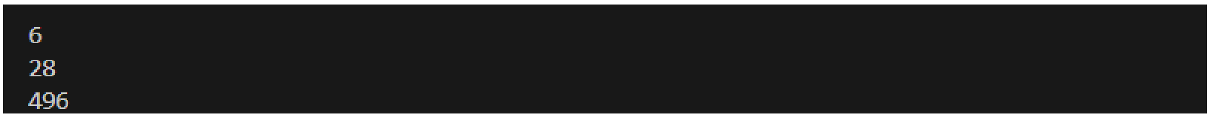
7. Write a program to print the first n perfect numbers. (Hint Perfect number means **a positive integer that is equal to the sum of its proper divisors)**
   Sample Input:
   N = 3
   Sample Output:
   First 3 perfect numbers are: 6 , 28 , 496
   Test Cases:
   1. N = 0
   2. N = 5
   3. N = -2
   4. N = -5
   5. N = 0.2

CODE:


nperfect.java > Language Support for Java(TM) by Red Hat > nperfect > main(String[])
Run | Debug | Run main | Debug main

```java
public class nperfect {
    public static void main(String[] args) {
        int n=3;
        int count=0;
        int num=2;
        int arr[]=new int[n];
        while(count<n){
            if(perfect(num)){
                System.out.println(num);
                count++;
            }
            num++;
        }
    }
    public static boolean perfect(int num){
        int Sum=0;
        for(int i=1;i<num;i++){
            if(num%i==0){
                Sum=Sum+i;
            }
        }
        if(Sum==num){
            return true;
        }
        return false;
    }
}
```

OUTPUT:



```
6
28
496
```

8. Write a program to enter the marks of a student in four subjects. Then calculate the total and aggregate, display the grade obtained by the student. If the student scores an aggregate greater than 75%, then the grade is Distinction. If aggregate is 60>= and <75, then the grade is First Division. If aggregate is 50 >= and <60, then the grade is Second Division. If aggregate is 40>= and <50, then the grade is Third Division. Else the grade is Fail.

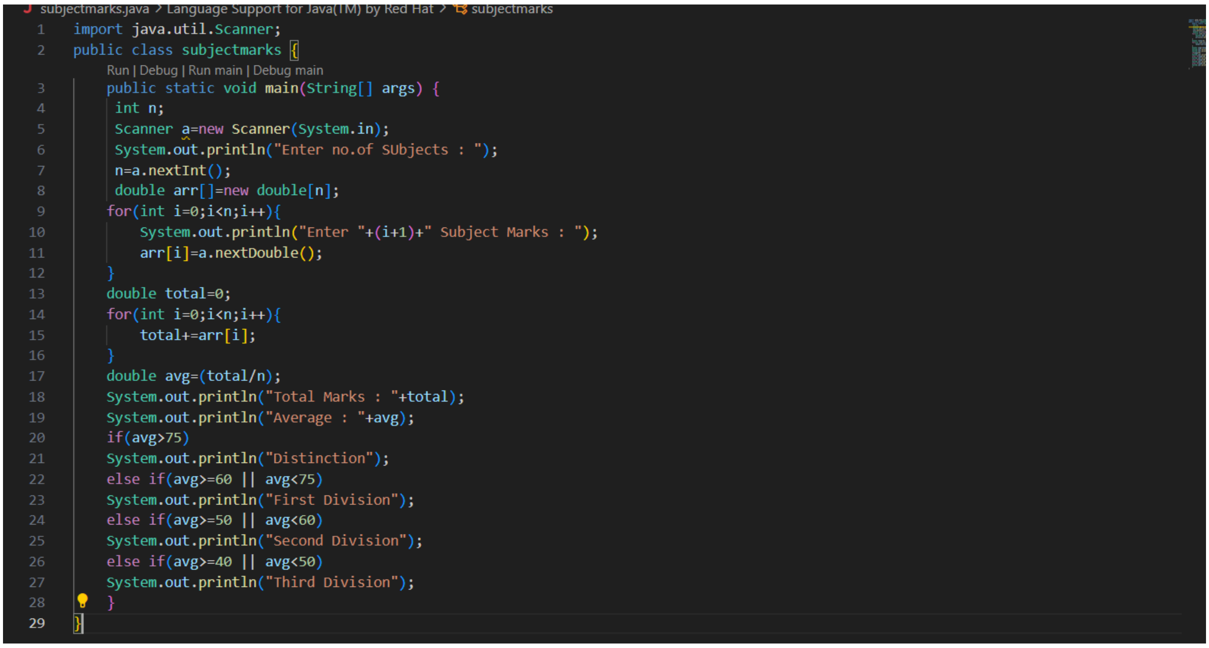
   Sample Input & Output:
   Enter the marks in python: 90
   Enter the marks in c programming: 91

Enter the marks in Mathematics: 92
Enter the marks in Physics: 93
Total= 366
Aggregate = 91.5
DISTINCTION

Test cases:

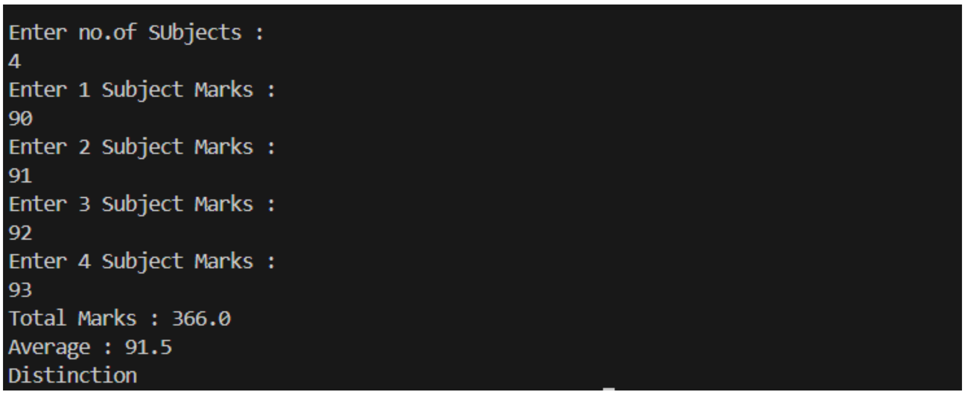
a) 18, 76,93,65
b) 73,78,79,75
c) 98,106,120,95
d) 96,73, -85,95
e) 78,59.8,76,79

CODE:


subjectmarks.java > Language Support for Java(TM) by Red Hat > subjectmarks
Run | Debug | Run main | Debug main

```java
import java.util.Scanner;
public class subjectmarks {
    public static void main(String[] args) {
     int n;
     Scanner a=new Scanner(System.in);
     System.out.println("Enter no.of SUbjects : ");
     n=a.nextInt();
     double arr[]=new double[n];
    for(int i=0;i<n;i++){
        System.out.println("Enter "+(i+1)+" Subject Marks : ");
        arr[i]=a.nextDouble();
    }
    double total=0;
    for(int i=0;i<n;i++){
        total+=arr[i];
    }
    double avg=(total/n);
    System.out.println("Total Marks : "+total);
    System.out.println("Average : "+avg);
    if(avg>75)
    System.out.println("Distinction");
    else if(avg>=60 || avg<75)
    System.out.println("First Division");
    else if(avg>=50 || avg<60)
    System.out.println("Second Division");
    else if(avg>=40 || avg<50)
    System.out.println("Third Division");
    }
}
```

OUTPUT:



```
Enter no.of SUbjects :
4
Enter 1 Subject Marks :
90
Enter 2 Subject Marks :
91
Enter 3 Subject Marks :
92
Enter 4 Subject Marks :
93
Total Marks : 366.0
Average : 91.5
Distinction
```

9. Write a program to read the numbers until -1 is encountered. Find the average of positive numbers and negative numbers entered by user.

Sample Input:

Enter -1 to exit…
Enter the number: 7
Enter the number: -2
Enter the number: 9
Enter the number: -8

Enter the number: -6
Enter the number: -4
Enter the number: 10
Enter the number: -1

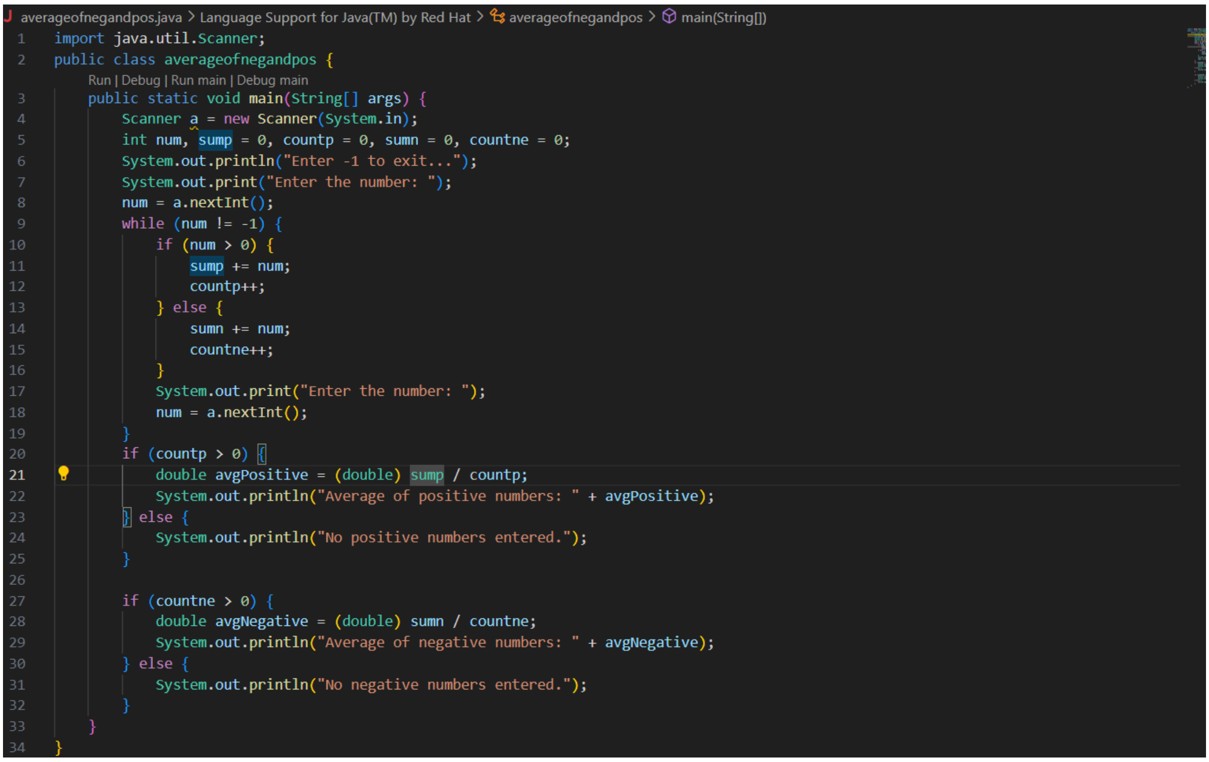
Sample Output:

The average of negative numbers is: -5.0
The average of positive numbers is : 8.66666667

Test cases:

1. -1,43, -87, -29, 1, -9
2. 73, 7-6,2,10,28,-1
3. -5, -9, -46,2,5,0
4. 9, 11, -5, 6, 0,-1
5. -1,-1,-1,-1,-1

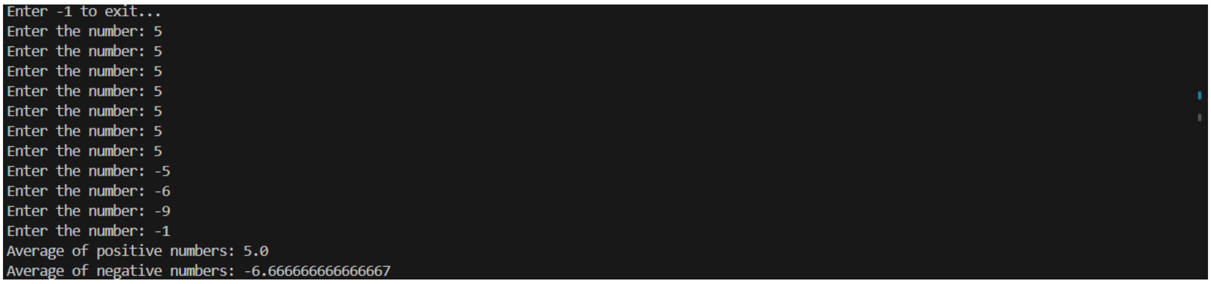
CODE:


averageofnegandpos.java > Language Support for Java(TM) by Red Hat > averageofnegandpos > main(String[])
Run | Debug | Run main | Debug main

```java
import java.util.Scanner;
public class averageofnegandpos {
    public static void main(String[] args) {
        Scanner a = new Scanner(System.in);
        int num, sump = 0, countp = 0, sumn = 0, countne = 0;
        System.out.println("Enter -1 to exit...");
        System.out.print("Enter the number: ");
        num = a.nextInt();
        while (num != -1) {
            if (num > 0) {
                sump += num;
                countp++;
            } else {
                sumn += num;
                countne++;
            }
            System.out.print("Enter the number: ");
            num = a.nextInt();
        }
        if (countp > 0) {
            double avgPositive = (double) sump / countp;
            System.out.println("Average of positive numbers: " + avgPositive);
        } else {
            System.out.println("No positive numbers entered.");
        }

        if (countne > 0) {
            double avgNegative = (double) sumn / countne;
            System.out.println("Average of negative numbers: " + avgNegative);
        } else {
            System.out.println("No negative numbers entered.");
        }
    }
}
```

OUTPUT:



```
Enter -1 to exit...
Enter the number: 5
Enter the number: 5
Enter the number: 5
Enter the number: 5
Enter the number: 5
Enter the number: 5
Enter the number: 5
Enter the number: -5
Enter the number: -6
Enter the number: -9
Enter the number: -1
Average of positive numbers: 5.0
Average of negative numbers: -6.666666666666667
```

10. Write a program to read a character until a * is encountered. Also count the number of uppercase, lowercase, and numbers entered by the users.

Sample Input:

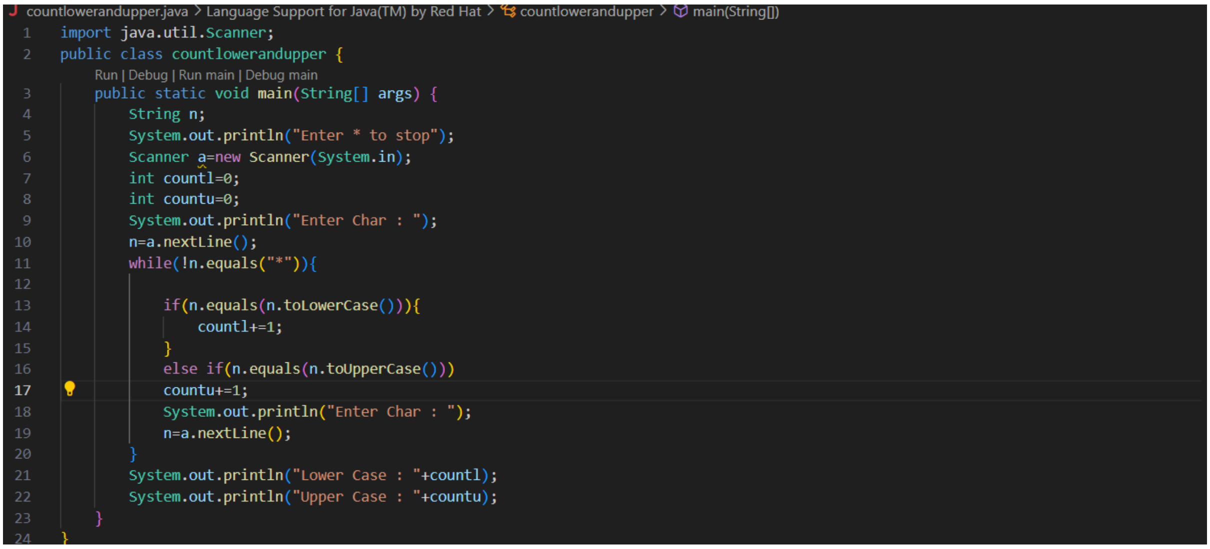
Enter * to exit…

Enter any character: W

Enter any character: d
Enter any character: A
Enter any character: G
Enter any character: g
Enter any character: H
Enter any character: *
Sample Output:
Total count of lower case:2
Total count of upper case:4
Total count of numbers =0
Test cases:

1. 1,7,6,9,5
2. S, Q, l, K,7, j, M
3. M, j, L, &, @, G
4. D, K, I, 6, L, *
5. *, K, A, e, 1, 8, %, *

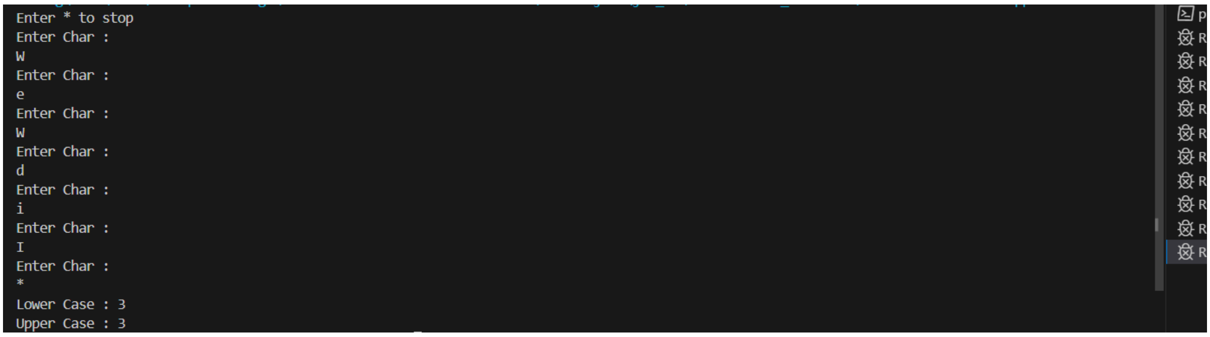
CODE:


countlowerandupper.java > Language Support for Java(TM) by Red Hat > countlowerandupper > main(String[])
Run | Debug | Run main | Debug main

```java
import java.util.Scanner;
public class countlowerandupper {
    public static void main(String[] args) {
        String n;
        System.out.println("Enter * to stop");
        Scanner a=new Scanner(System.in);
        int countl=0;
        int countu=0;
        System.out.println("Enter Char : ");
        n=a.nextLine();
        while(!n.equals("*")){

            if(n.equals(n.toLowerCase())){
                countl+=1;
            }
            else if(n.equals(n.toUpperCase()))
            countu+=1;
            System.out.println("Enter Char : ");
            n=a.nextLine();
        }
        System.out.println("Lower Case : "+countl);
        System.out.println("Upper Case : "+countu);
    }
}
```

OUPUT:



```
Enter * to stop
Enter Char :
W
Enter Char :
e
Enter Char :
W
Enter Char :
d
Enter Char :
i
Enter Char :
I
Enter Char :
*
Lower Case : 3
Upper Case : 3
```