Danny has recently got his job offer as an Event Concept Creator at Sparsh Event Services. The Company has sent him a detailed salary structure with details of his basic salary, HRA and DA. The Company has promised to pay him as under:

If his basic salary is less than Rs. 15000, then HRA = 15% of basic salary and DA = 90% of basic salary.

If his basic salary is either equal to or above Rs. 15000, then HRA = Rs. 5000 and DA = 98% of basic salary.

If the Danny's salary is given as input, write a program to find his gross salary, using method

Note : Gross Salary = Basic Salary+HRA+DA*

Input Format

First line of the input is an integer that corresponds to the basic salary of Danny.

Constraints

No Constraints

Output Format

Output should display the double value that refers to the gross salary of Danny. Display the output correct to 2 decimal places.

Sample Input 0

12000

Sample Output 0

24600.00

ANSWER:

```
mport java.util.Scanner;
 public class SalaryComputation {
 public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int basicSalary = scanner.nextInt();
```

```
        double hra, da;

        if (basicSalary < 15000) {
            hra = 0.15 * basicSalary;
            da = 0.90 * basicSalary;
        } else {
            hra = 5000;
            da = 0.98 * basicSalary;
        }

        double grossSalary = basicSalary + hra + da;

        // Print gross salary formatted to 2 decimal places
        System.out.printf("%.2f\n", grossSalary);

        scanner.close();
    }
}
```

College Management wants to separate the eligible students for their placement.so find the eligible students for the placement.

Notes:-

- If the students has 1 arrear and the cpga is above 70 - They are eligible for Placement.
- If the students has 1 or 2 arrear and the cpga is above 75 -They are eligible for Placement.
- Remaining students aren't eligible for Placement.

Input Format

input consists of one String and two integer.

Constraints

No constraints

Output Format

print the statement "Eligible for Placement" or "Not Eligible for Placement".

Sample Input 0

```
John
1
76
```

Sample Output 0

```
Name                    of                    the                    Student:John
John             is             Eligible             for             Placement
```

Sample Input 1

```
John
2
70
```

Sample Output 1

```
Name                    of                    the                    Student:John
John is Not Eligible for Placement
```

```
import java.util.Scanner;

public class PlacementEligibility {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Taking inputs
        String name = sc.nextLine();
        int arrears = sc.nextInt();
        int cgpa = sc.nextInt();
```

```
        // Print student name
        System.out.println("Name of the Student:" + name);

        // Check eligibility
        if ((arrears == 1 && cgpa > 70) ||
            ((arrears == 1 || arrears == 2) && cgpa > 75)) {
            System.out.println(name + " is Eligible for Placement");
        } else {
            System.out.println(name + " is Not Eligible for Placement");
        }

        sc.close();
    }
}
```

Maya wants to know how to find whether the alphabet is Vowel or Consonant.could you please help her to find the alphabet is Vowel or Consonant.

Input Format

input consists of one Character.

Constraints

No Constraints

Output Format

print whether the character is Vowel or Consonant or Invalid Input.

Sample Input 0

```
A
```

Sample Output 0

```
The Character A is Vowel
```

Sample Input 1

```
T
```

Sample Output 1

```
The Character T is Consonant
```

Sample Input 2

```
u
```

Sample Output 2

```
The Character u is Vowel
```

```
import java.util.Scanner;
public class VowelConsonant {
    public static void main(String[] args) {
     Scanner sc = new Scanner(System.in);
 char ch = sc.next().charAt(0); // Read first character         if ((ch >= 'A' && ch <= 'Z') || (ch >= 'a' && ch <= 'z')) {
      if (ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U' ||
              ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
              System.out.println("The Character " + ch + " is Vowel");
           } else {
              System.out.println("The Character " + ch + " is Consonant");
           }
        } else {
           System.out.println("Invalid Input");
        }
      sc.close();
```

```
    }
}
```

Write a program that takes a number from the user and generates an integer between 1 and 7. It displays the weekday name.

Input Format

Input consists of one integer.

Constraints

number 1 to 7

Output Format

Execute the day name of the given day number.If the given number is beyond the limit execute the statement"Enter a valid Number".

Sample Input 0

```
1
```

Sample Output 0

```
Monday
```

Sample Input 1

```
7
```

Sample Output 1

```
Sunday
```

Sample Input 2

```
14
```

Sample Output 2

```
Enter a valid Input
```

```
import java.util.Scanner;
```

```
public class Weekday {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number (1-7): ");
        int dayNumber = sc.nextInt();

        switch(dayNumber) {
            case 1:
                System.out.println("Monday");
                break;
            case 2:
                System.out.println("Tuesday");
                break;
            case 3:
                System.out.println("Wednesday");
                break;
            case 4:
                System.out.println("Thursday");
                break;
            case 5:
                System.out.println("Friday");
                break;
            case 6:
                System.out.println("Saturday");
                break;
            case 7:
```

```
            System.out.println("Sunday");
            break;
        default:
            System.out.println("Enter a valid Input");
    }

    sc.close();
  }
}
```

Write a program to Check if given number is 3 -digit and find it is palindrome

Input Format

Get a input as a integer value

Constraints

only 3 digit number

Output Format

print palindrome or not or invalid input

Sample Input 0

```
121
```

Sample Output 0

```
palindrome
```

Sample Input 1

```
234
```

Sample Output 1

```
not                                                                palindrome
```

Sample Input 2

```
72
```

Sample Output 2

```
Invalid Input
```

```
import java.util.Scanner;

public class PalindromeCheck {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int num = sc.nextInt();

        // Check if it's a 3-digit number
        if (num >= 100 && num <= 999) {
            int original = num;
            int reversed = 0;

            while (num > 0) {
                int digit = num % 10;
                reversed = reversed * 10 + digit;
                num = num / 10;
            }

            if (original == reversed) {
```

```
            System.out.println("palindrome");
        } else {
            System.out.println("not palindrome");
        }
    } else {
        System.out.println("Invalid Input");
    }
  }
}
```

Write a program to find whether the given number is a Harshad number or not. Note that Harshard number is an integer that is divisible by the sum of its digits.

Input Format

Input consists of 1 integer.

Constraints

No Constraints

Output Format

If the given number is a Harshad Number, display “Harshad Number” or display “Not Harshad Number”.

Sample Input 0

```
11
```

Sample Output 0

```
Not Harshad Number
```

Sample Input 1

```
1729
```

Sample Output 1

Harshad Number

Contest ends in 2 days

Submissions: 228
Max Score: 10
Difficulty: Medium
Rate This Challenge:

More

```
import java.util.Scanner;


public class HarshadNumber {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);


        // Input
        int n = sc.nextInt();
        sc.close();


        int temp = n;
        int digitSum = 0;


        // Calculate sum of digits
        while (temp > 0) {
            digitSum += temp % 10;
            temp /= 10;
        }
```

```
        // Check divisibility
        if (n % digitSum == 0) {
            System.out.println("Harshad Number");
        } else {
            System.out.println("Not Harshad Number");
        }
    }
}
```

Write the program to calculate the sum of the Fibonacci series between the range of the two values. The n is the start of the range value and m is the end of the range value. Finally, return the sum of the range value of the Fibonacci Series.

Input Format

Input consists of two integers.

Constraints

N is greater than equal to one and lesser than equal to 20

M is greater than equal to one and lesser than equal to 20

Output Format

The sum of the range value.

If the constraints are more than the input print the statement is "Invalid Input".

Sample Input 0

```
3
5
```

Sample Output 0

```
The Sum of Fibonacci value is 6.0
```

Sample Input 1

```
22
23
```

Sample Output 1

```
Invalid Input
```

```
import java.util.Scanner;

public class FibonacciRangeSum {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Input
        int n = sc.nextInt();
        int m = sc.nextInt();
        sc.close();

        // Check constraints
        if (n < 1 || n > 20 || m < 1 || m > 20 || n > m) {
            System.out.println("Invalid Input");
            return;
        }

        int a = 0, b = 1;
        int sum = 0;

        // Generate Fibonacci numbers and sum those in the range
        for (int i = 1; i <= m; i++) {
            int fib;
```

```
            if (i == 1) {
                fib = a;
            } else if (i == 2) {
                fib = b;
            } else {
                fib = a + b;
                a = b;
                b = fib;
            }

            if (i >= n) {
                sum += fib;
            }
        }

        System.out.println("The Sum of Fibonacci value is " + (double)sum);
    }
}
```

write the program to print the Multiplication table for nth table and till n count.

Input Format

consdier one integer input

Constraints

1=>n<=9

Output Format

Multiplication table

Sample Input 0

```
5
```

Sample Output 0

```
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
```

Sample Input 1

```
11
```

Sample Output 1

```
Invalid Input
```

```
import java.util.Scanner;

public class MultiplicationTable {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        sc.close();
        if (n < 1 || n > 9) {
            System.out.println("Invalid Input");
            return;
        }
        for (int i = 1; i <= n; i++) {
            System.out.println(n + " x " + i + " = " + (n * i));
        }
```

```
    }
}
```

Write a program to find whether the given number is an Abundant number or not. Note: An abundant number is a number for which the sum of its proper divisors is greater than the number itself. For example, integer 12 is an abundant number. The divisors of 12 are 1, 2, 3, 4 and 6. The sum of divisors of 12 is 16. As 12 < 16, it is an abundant number.

Input Format

Input consists of 1 integer.

Constraints

No Constraint

Output Format

If it is an Abundant number display “Abundant Number” or display “Not Abundant Number”.

Sample Input 0

```
6
```

Sample Output 0

```
Not Abundant Number
```

Sample Input 1

```
12
```

Sample Output 1

```
Abundant Number
```

```
import java.util.Scanner;

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

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

        // Input
        int n = sc.nextInt();
        sc.close();

        int sum = 0;

        // Find proper divisors and calculate their sum
        for (int i = 1; i <= n / 2; i++) {
            if (n % i == 0) {
                sum += i;
            }
        }

        // Check condition for Abundant Number
        if (sum > n) {
            System.out.println("Abundant Number");
        } else {
            System.out.println("Not Abundant Number");
        }
    }
}
```

write a program to find the given 3- digit number is Armstrong number or not

Input Format

Input corresponds to the 3-digit Integer Number

Constraints

N>=100 <=999

Output Format

Yes or No

Sample Input 0

```
153
```

Sample Output 0

```
Yes
```

Sample Input 1

```
1245
```

Sample Output 1

```
No
```

import java.util.Scanner;

public class ArmstrongNumber { public static void main(String[] args) { Scanner sc = new Scanner(System.in);

```
//                                                  Input
 int                n                =                sc.nextInt();
 sc.close();

 //    Check    constraints:    must    be    a    3-digit    number
 if       (n       <       100       ||       n       >       999)       {
     System.out.println("No");
     return;
 }

 int                   temp                   =                   n;
 int                   sum                    =                   0;

 //      Calculate      sum      of      cubes      of      digits
```

```
    while (temp > 0) {
        int digit = temp % 10;
        sum += digit * digit * digit;
        temp /= 10;
    }

    // Check if sum equals the original number
    if (sum == n) {
        System.out.println("Yes");
    } else {
        System.out.println("No");
    }
}


}
```

The rules for generating Collatz Sequence are:

If n is even: n = n / 2 If n is odd: n = 3n + 1 For example, if the starting number is 5 the sequence is:

5 -> 16 -> 8 -> 4 -> 2 -> 1 It has been proved that for almost all integers, the repeated application of the above rule will result in a sequence that ends at 1.

Given a positive integer, write a program to print this sequence and the number of times this rule needs to be applied in order to reach 1.

Input Format

Input consists of a positive integer.

Constraints

No constraints

Output Format

Print the numbers in the sequence, one per line and finally print the number of times the rule has to be applied in order to reach 1.

Sample Input 0

```
5
```

Sample Output 0

```
Enter a number
5
16
8
4
2
1
count:5
```

Sample Input 1

```
1
```

Sample Output 1

```
Enter a number
1
count:0
```

```java
public class CollatzSequence {

   public static void main(String[] args) {

       Scanner sc = new Scanner(System.in);


       System.out.println("Enter a number");

       int n = sc.nextInt();

       sc.close();


       if (n <= 0) {

           System.out.println("Invalid Input");
```

```
            return;
        }

        int count = 0;
        int temp = n;

        while (temp != 1) {
            System.out.println(temp);
            if (temp % 2 == 0) {
                temp = temp / 2;
            } else {
                temp = 3 * temp + 1;
            }
            count++;
        }

        // Print the last number 1
        System.out.println(temp);
        System.out.println("count:" + count);
    }
}
```

sum of even number 1

Problem
Submissions
Leaderboard
Discussions

Find the sum of even number with the range

Input Format

Consider two integer input n and m

Constraints

n and m values greater than 0 and less than 30

Output Format

sum of even values

Sample Input 0

```
5
15
```

Sample Output 0

```
50
```

Sample Input 1

```
22
10
```

Sample Output 1

```
Invalid Input
```

```
import java.util.Scanner;
```

```
public class SumEvenRange {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

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

        // check constraints
        if (n <= 0 || m <= 0 || n >= 30 || m >= 30 || n > m) {
            System.out.println("Invalid Input");
        } else {
            int sum = 0;
            for (int i = n; i <= m; i++) {
                if (i % 2 == 0) {
                    sum += i;
                }
            }
            System.out.println(sum);
        }
    }
}
```

Reverse a Number Using Loop 1

Problem
Submissions
Leaderboard
Discussions

write the program to Reverse a given number Using Looping statement

Input Format

get a user input as an integer value

Constraints

0< n >1000000000

Output Format

Reverse order of a number

Sample Input 0

```
1423
```

Sample Output 0

```
3241
```

Sample Input 1

```
-128
```

Sample Output 1

```
Invalid                                                                    Input
```

Sample Input 2

```
987654321
```

Sample Output 2

```
123456789
```

```
import java.util.Scanner;

public class ReverseNumber {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        long n = sc.nextLong();  // Use long to handle large numbers
```

```
    sc.close();

    // Check constraints: number must be positive and <= 1,000,000,000
    if (n <= 0 || n > 1000000000) {
        System.out.println("Invalid Input");
        return;
    }

    long reversed = 0;
    long temp = n;

    // Reverse the number using loop
    while (temp > 0) {
        long digit = temp % 10;
        reversed = reversed * 10 + digit;
        temp /= 10;
    }

    System.out.println(reversed);
}
```