**1. Type Conversion Challenge**

**Problem:**

**Write a Java program that accepts an integer, a float, and a character from the user.**

**Perform the following operations:**

** Convert the integer to a float and add it to the float input.**

** Convert the character to its ASCII value and add it to the integer.**

** Display the results with proper data type usage.**

import java.util.\*;

class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

int a=sc.nextInt();

float b= sc.nextFloat();

char c=sc.next().charAt(0);

int d= (int)c+a;

float e=((float)a)+b;

System.out.println(d);

System.out.println(e);

}

}

**2. Arithmetic Expression Evaluator**

**Problem:**

**Write a program that takes three numbers from the user: two integers and one double.**

**Perform and display the results of the following:**

** Addition, Subtraction, Multiplication, and Division between the integers.**

** Multiply the result of the addition with the double value.**

** Ensure proper type casting is used wherever necessary.**

import java.util.\*;

class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

int a=sc.nextInt();

int b=sc.nextInt();

double c=sc.nextDouble();

int d=a+b, e=a-b, f=a\*b, g=a/b;

System.out.println(d);

System.out.println(e);

System.out.println(f);

System.out.println(g);

double h=d\*c;

System.out.println(h);

}

}

**3. Bitwise Operator Experiment**

**Problem:**

**Create a program that reads two integer numbers from the user. Perform the following**

**bitwise operations and print the results:**

** AND**

** OR**

** XOR**

** Left Shift (both numbers by 2 bits)**

** Right Shift (both numbers by 2 bits)**

import java.util.\*;

class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

int a=sc.nextInt();

int b=sc.nextInt();

int c= a&b, d=a|b, e=a^b, f= a<<2, g=b<<2, h=a>>2, i=b>>2;

System.out.println("AND: "+c);

System.out.println("OR: "+d);

System.out.println("XOR: "+e);

System.out.println("a << 2: "+f);

System.out.println("b << 2: "+g);

System.out.println("a >> 2: "+h);

System.out.println("b >> 2: "+i);

}

}

**1. Check Whether a Character is a Vowel or Consonant**

**Input: A single alphabet character**

**Output: Whether it is a vowel or a consonant**

**Example: 'a' → Vowel, 'z' → Consonant**

import java.util.\*;

class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

char c=sc.next().charAt(0);

if(c=='a'||c=='e'||c=='i'||c=='o'||c=='u'||c=='A'||c=='E'||c=='I'||c=='O'||c=='U')

{

System.out.print(c+ " is a Vowel");

}

else{

System.out.print(c+ " is a Consonant");

}

}

}

**2. Print the Grade Based on Marks**

**Input: Marks (0 to 100)**

**Use if-else ladder to print:**

**♣ 90–100 → Grade A**

**♣ 75–89 → Grade B**

**♣ 60–74 → Grade C**

**♣ 40–59 → Grade D**

**♣ Below 40 → Fail**

import java.util.\*;

class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

int marks=sc.nextInt();

if(marks>=90&&marks<=100){

System.out.print("Grade A");

}

else if(marks>=75&&marks<=89){

System.out.print("Grade B");

}

else if(marks>=60&&marks<=74){

System.out.print("Grade C");

}

else if(marks>=40&&marks<=59){

System.out.print("Grade D");

}

else if(marks<40){

System.out.print("Fail");

}

else{

System.out.print("Invalid Marks");

}

}

}

**3. Simple Interest or Compound Interest Calculator**

**Input: User chooses 1 for Simple Interest, 2 for Compound Interest**

**Take input for P (principal), R (rate), T (time)**

**Output: Display the calculated interest**

import java.util.\*;

class Main {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

int p=sc.nextInt();

int r=sc.nextInt();

int t=sc.nextInt();

int SI=(p\*r\*t)/100;

double a=p\*Math.pow((1+(r/100.0)),t);

double CI=a-p;

System.out.println(SI);

System.out.println(CI);

}

}

**4. Print All Prime Numbers from 1 to N**

**Input: A number N**

**Output: All prime numbers between 1 and N using for loop and if conditions**

**Assignment 3 - Arrays**

**1. Search for an Element**

**Problem Statement:**

**Write a program to search for an element in the array. If found, print its index (0-based);**

**otherwise, print &quot;Not found&quot;.**

**Input:**

** First line: Integer n (size of the array)**

** Second line: n space-separated integers (array elements)**

** Third line: Integer x (element to search)**

**Output:**

** Index of the first occurrence of x in the array, or &quot;Not found&quot;**

**Constraints:**

** 1 &lt;= n &lt;= 100**

** -10^4 &lt;= arr[i], x &lt;= 10^4**

**Sample Input:**

**5**

**3 8 2 9 6**

**9**

**Sample Output:**

**3**

package Sakthi;

import java.util.\*;

public class Hello {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int[] arr=new int[n];

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

arr[i]=sc.nextInt();

}

int a=sc.nextInt();

boolean found = false;

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

if(arr[i]==a) {

System.out.print(i);

found=true;

break;

}

}

if(!found) {

System.out.print("Not Found");

}

}

}

**2.Duplicate Elements**

**Problem Statement:**

**Write a program to identify and print all duplicate elements in a 1D array. If no duplicates**

**are found, print &quot;No duplicates&quot;.**

**Input:**

** First line: Integer n (number of elements)**

** Second line: n space-separated integers**

**Output:**

** All duplicate elements (in any order)**

** Or &quot;No duplicates&quot; if all elements are unique**

**Constraints:**

** 1 &lt;= n &lt;= 100**

** -10^4 &lt;= arr[i] &lt;= 10^4**

**Sample Input:**

**7**

**5 3 8 5 6 3 2**

**Sample Output:**

**5 3**

package Sakthi;

import java.util.Scanner;

public class Hello {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int[] arr=new int[n];

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

arr[i]=sc.nextInt();

}

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

boolean found=false;

for(int j=0;j<i;j++) {

if(arr[i]==arr[j]) {

found=true;

break;

}

}

if(!found) {

for(int k=i+1;k<n;k++) {

if(arr[i]==arr[k]) {

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

break;

}

}

}

}

}

}

**3.Left Rotation by K Position**

**Problem Statement:**

**Write a program to perform left rotation of a 1D array by k position.**

**Input:**

** irst line: Integer n — the size of the array**

** Second line: n space-separated integers — the elements of the array**

** Third line: Integer k — number of positions to rotate the array to the left**

**Output:**

** A single line containing the rotated array elements after k left rotations.**

**Constraints:**

** 1 &lt;= n &lt;= 100**

** -10^4 &lt;= arr[i] &lt;= 10^4**

** 0 &lt;= k &lt;= 100**

**Sample Input:**

**6**

**1 2 3 4 5 6**

**2**

**Sample Output:**

**3 4 5 6 1 2**

package Sakthi;

import java.util.Scanner;

public class Hello {

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int[] arr=new int[n];

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

arr[i]=sc.nextInt();

}

int k=sc.nextInt();

for(int i=k;i<n;i++) {

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

}

for(int i=0;i<k;i++) {

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

}

}

}