JAVA PROGRAM

```
package airthmatic;
import java.util.Scanner;
public class Airthmaticopperation {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Scanner s=new Scanner(System.in);

		System.out.println("Enter the first integer:");
		int num1=s.nextInt();
		System.out.println("Enter second integer:");
		int num2=s.nextInt();

		int sum=num1+num2;
		int difference=num1-num2;
		int product=num1*num2;

		System.out.println("Sum:"+sum);
		System.out.println("Difference:"+difference);
		System.out.println("Product:"+product);
		s.close();
		}
}
```

QUADRATIC

```
package airthmatic;
import java.util.Scanner;
public class qudratic {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Scanner scanner = new Scanner(System.in);

	  System.out.println("Quadratic Equation Solver");
	  System.out.print("Enter the coefficient 'a': ");
	  double a = scanner.nextDouble();

	  System.out.print("Enter the coefficient 'b': ");
	  double b = scanner.nextDouble();

	  System.out.print("Enter the coefficient 'c': ");
	  double c = scanner.nextDouble();

	  double discriminant = b * b - 4 * a * c;

	  if (discriminant > 0) {
	      double root1 = (-b + Math.sqrt(discriminant)) / (2 * a);
	      double root2 = (-b - Math.sqrt(discriminant)) / (2 * a);

	      System.out.println("Two real solutions:");
	      System.out.println("Root 1: " + root1);
	      System.out.println("Root 2: " + root2);
	  } else if (discriminant == 0) {
```

```
            double root = -b / (2 * a);
            System.out.println("One real solution:");
            System.out.println("Root: " + root);
        } else {
            System.out.println("No real solutions. The discriminant is
negative.");
        }

        scanner.close();


    }

}
```

SI AND CI

```
package airthmatic;
import java.util.Scanner;
public class SiandCi {

    public static void main(String[] args) {
            // TODO Auto-generated method stub
            Scanner s=new Scanner(System.in);
            System.out.println("----INNTREST CALCULATOR----");
            System.out.println("Enter the principal amount(initial
investment):");
            double P=s.nextDouble();
            System.out.println("Enter the annualintrest rate(in
percentage");
            double R=s.nextDouble();
            System.out.println("Enter the Time period in year:");
            double T=s.nextDouble();
            System.out.println("Enter the number of intrest compounded in
the year:");
            double F=s.nextDouble();

            double SI=(P*R*T)/100.0;
            double CI=P*Math.pow(1+(R/(100*F)),F*T)-P;
            System.out.println("Simple Interest: " + SI);
        System.out.println("Compound Interest: " + CI);
        s.close();

    }

}
```

2.SWAP

```
package airthmatic;
import java.util.Scanner;
public class swapnum {

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

```
		// TODO Auto-generated method stub
		Scanner sc= new Scanner(System.in);
		System.out.println("enter the 1st integer:");
		int num1=sc.nextInt();
		System.out.println("enter the 2nd integer:");
		int num2=sc.nextInt();
		swapwithtemp(num1,num2);
		swapwithouttemp(num1,num2);
		sc.close();

	}
	private static void swapwithtemp(int num1,int num2) {
		System.out.println("swapping with temp variable");
		System.out.println("before swap : num1="+num1+",num2="+num2);
		int temp=num1;
		num1=num2;
		num2=temp;
		System.out.println("after swap : num1="+num1+",num2="+num2);
		System.out.println();
	}

	private static void swapwithouttemp(int num1,int num2) {
		System.out.println("swapping without temp variable");
		System.out.println("before swap : num1="+num1+",num2="+num2);
	    num1=num1+num2;
	    num2=num1-num2;
	    num1=num1-num2;
	    System.out.println("after swap : num1="+num1+",num2="+num2);
	    System.out.println();
		}



}
```

PRIMENUMMBER

```
package airthmatic;
import java.util.Scanner;
public class primenumbers {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Scanner s=new Scanner(System.in);
		System.out.println("Enter max number: ");

	  String input = s.nextLine();
	  int maxNumber = Integer.parseInt( input );

	  System.out.println("List of the prime number between 1 - " +
maxNumber);

	  for (int num = 2; num <= maxNumber; num++)
	  {
	      boolean isPrime = true;
	      for (int i=2; i <= num/2; i++)
	      {
	          if ( num % i == 0)
	          {
```

```
                    isPrime = false;
                    break;
                }
            }

            if ( isPrime == true )
                System.out.println(num);
            s.close();
        }
    }
}
```

FACTORIAL

```
package airthmatic;
import java.util.Scanner;
public class factorial {

	public static void main(String[] args) {
		// TODO Auto-generated method stub

	 Scanner sc = new Scanner(System.in);

	System.out.print("Enter a positive integer: ");
	int number = sc.nextInt();

	if (number < 0)
	{
	    System.out.println("Factorial is not defined for negative
numbers.");
	}
	else
	{
	    int factorial = 1;
	    for (int i = 1; i <= number; i++) {
	        factorial=factorial*i;
	    }
	    System.out.println("Factorial of " + number + " is " +
factorial);
	}

	sc.close();


	}

}
```

BINARYSEARCH

package airthmatic;

import java.util.Scanner;

```
import java.util.Arrays;
public class exp3binary {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Scanner scanner = new Scanner(System.in);

    // Get the number of elements
    System.out.print("Enter the number of elements: ");
    int n = scanner.nextInt();

    // Create an array to store the elements
    int[] arr = new int[n];

    // Get the elements from the user
    System.out.println("Enter the elements (sorted for binary search):");
    for (int i = 0; i < n; i++) {
       arr[i] = scanner.nextInt();
    }

    // Get the element to search
    System.out.print("Enter the element to search: ");
    int searchElement = scanner.nextInt();



    // Perform linear search
    int linearSearchIndex = linearSearch(arr, searchElement);

    if (linearSearchIndex != -1) {
       System.out.println("Linear Search: Element found at index " + linearSearchIndex);
```

```
    } else {
        System.out.println("Linear Search: Element not found");
    }


    // Perform binary search (requires a sorted array)
    Arrays.sort(arr); // Sorting for binary search
    System.out.println("Sorted Array:");
    for(int i=0;i<arr.length;i++) {
         System.out.print(arr[i]+" ");
    }
    int binarySearchIndex = binarySearch(arr, searchElement);

    if (binarySearchIndex != -1) {
        System.out.println("\nBinary Search: Element found at index " + binarySearchIndex);
    } else {
        System.out.println("\nBinary Search: Element not found");
    }

    scanner.close();
}

// Linear Search method
private static int linearSearch(int[] arr, int searchElement) {
    for (int i = 0; i < arr.length; i++) {
        if (arr[i] == searchElement) {
            return i; // Element found, return index
        }
    }
    return -1; // Element not found
}
```

```
    // Binary Search method (requires a sorted array)
    private static int binarySearch(int[] arr, int searchElement) {
        int left = 0;
        int right = arr.length - 1;

        while (left <= right) {
            int mid = left + (right - left) / 2;

            if (arr[mid] == searchElement) {
                return mid; // Element found, return index
            } else if (arr[mid] < searchElement) {
                left = mid + 1; // Search the right half
            } else {
                right = mid - 1; // Search the left half
            }
        }

        return -1; // Element not found


          }

}
```

BUBBLESORT

```
package airthmatic;
import java.util.Scanner;
public class exp3bbuublesort {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Scanner scanner = new Scanner(System.in);
```

```
        // Get the number of elements
        System.out.print("Enter the number of elements: ");
        int n = scanner.nextInt();

        // Create an array to store the elements
        int[] arr = new int[n];

        // Get the elements from the user
        System.out.println("Enter the elements:");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }

        // Sort the array in ascending order
        bubbleSortAscending(arr);

        // Display the sorted array in ascending order
        System.out.println("Sorted array in ascending order:");
        printArray(arr);

        // Sort the array in descending order
        bubbleSortDescending(arr);

        // Display the sorted array in descending order
        System.out.println("Sorted array in descending order:");
        printArray(arr);

        scanner.close();
    }

    // Bubble Sort in Ascending Order
    private static void bubbleSortAscending(int[] arr) {
        int n = arr.length;
        for (int i = 0; i < n - 1; i++) {
            for (int j = 0; j < n - 1 - i; j++) {
                if (arr[j] > arr[j + 1]) {
                    // Swap arr[j] and arr[j + 1]
                    int temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                }
            }
        }
    }

    // Bubble Sort in Descending Order
    private static void bubbleSortDescending(int[] arr) {
        int n = arr.length;
        for (int i = 0; i < n - 1; i++) {
            for (int j = 0; j < n - 1 - i; j++) {
                if (arr[j] < arr[j + 1]) {
                    // Swap arr[j] and arr[j + 1]
                    int temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                }
            }
        }
    }
```

```
    // Utility method to print an array
    private static void printArray(int[] arr) {
        for (int value : arr) {
            System.out.print(value + " ");
        }
        System.out.println();


      }

}
```

LARGESTAND SMALLEST

```
package airthmatic;
import java.util.Scanner;
public class exp3clargestandsmallestinaray {

      public static void main(String[] args) {
            // TODO Auto-generated method stub
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of elements in the array: ");
        int n = scanner.nextInt();
        int[] arr = new int[n];

        System.out.println("Enter " + n + " elements:");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }

        int largest = arr[0];
        int smallest = arr[0];

        for (int i =1; i < n; i++) {
            if (arr[i] > largest) {
                largest = arr[i];
            }
            if (arr[i] < smallest) {
                smallest = arr[i];
            }
        }

        System.out.println("Largest element in the array: " + largest);
        System.out.println("Smallest element in the array: " + smallest);

        scanner.close();
    }


      }
```

MATRIX

```
package airthmatic;
import java.util.Scanner;
public class Exp4matrix {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
	  Scanner sc = new Scanner(System.in);

	  // Input the dimensions of matrices A and B
	  System.out.println("Enter the number of rows for matrices A and
B:");
	  int numRows = sc.nextInt();
	  System.out.println("Enter the number of columns for matrices A and
B: ");
	  int numCols = sc.nextInt();

	  int[][] matrixA = new int[numRows][numCols];
	  int[][] matrixB = new int[numRows][numCols];

	  // Input elements for matrix A
	  System.out.println("Enter elements for matrix A:");
	  for (int i = 0; i < numRows; i++) {
	      for (int j = 0; j < numCols; j++) {
	          matrixA[i][j] = sc.nextInt();
	      }
	  }

	  // Input elements for matrix B
	  System.out.println("Enter elements for matrix B:");
	  for (int i = 0; i < numRows; i++) {
	      for (int j = 0; j < numCols; j++) {
	          matrixB[i][j] = sc.nextInt();
	      }
	  }

	  // Perform matrix addition
	  int[][] sumMatrix = new int[numRows][numCols];
	  for (int i = 0; i < numRows; i++) {
	      for (int j = 0; j < numCols; j++) {
	          sumMatrix[i][j] = matrixA[i][j] + matrixB[i][j];
	      }
	  }

	  // Perform matrix multiplication
	  int[][] productMatrix = new int[numRows][numCols];
	  for (int i = 0; i < numRows; i++) {
	      for (int j = 0; j < numCols; j++) {
	          for (int k = 0; k < numCols; k++) {
	              productMatrix[i][j] += matrixA[i][k] * matrixB[k][j];
	          }
	      }
	  }

	  // Display the result of matrix addition
	  System.out.println("Matrix A + Matrix B:");
	  for (int i = 0; i < numRows; i++) {
	      for (int j = 0; j < numCols; j++) {
	          System.out.print(sumMatrix[i][j] + " ");
	      }
	      System.out.println();
```

```
        }

        // Display the result of matrix multiplication
        System.out.println("Matrix A * Matrix B:");
        for (int i = 0; i < numRows; i++) {
            for (int j = 0; j < numCols; j++) {
                System.out.print(productMatrix[i][j] + " ");
            }
            System.out.println();
        }

        // Calculate and display the determinant of matrix A
        int determinantA = calculateDeterminant(matrixA);
        System.out.println("Determinant of Matrix A: " + determinantA);
        int determinantB = calculateDeterminant(matrixB);
        System.out.println("Determinant of Matrix B: " + determinantB);
        sc.close();
    }

    // Function to calculate the determinant of a 2x2 matrix
    public static int calculateDeterminant(int[][] matrix) {
        if (matrix.length != 2 || matrix[0].length != 2 || matrix[1].length
!= 2) {
            return 0; // Determinant is only defined for 2x2 matrices
        }
        return matrix[0][0] * matrix[1][1] - matrix[0][1] * matrix[1][0];


      }

}
```

Reversce string

```
package airthmatic;
import java.util.Scanner;
public class Exp5reversestrigpalidrome {

      public static void main(String[] args) {
            // TODO Auto-generated method stub
        Scanner sc = new Scanner(System.in);

        // Input a string
        System.out.print("Enter a string: ");
        String inputString = sc.nextLine();

        // Reverse the string
        String reversedString = reverseString(inputString);
        System.out.println("Reversed String: " + reversedString);

        // Check if the string is a palindrome
        boolean isPalindrome = isPalindrome(inputString);
        if (isPalindrome) {
            System.out.println("The string is a palindrome.");
        } else {
            System.out.println("The string is not a palindrome.");
```

```
        }

        // Compare two strings
        System.out.print("Enter another string for comparison: ");
        String secondString = sc.nextLine();
        boolean areEqual = compareStrings(inputString, secondString);
        if (areEqual) {
            System.out.println("The two strings are equal.");
        } else {
            System.out.println("The two strings are not equal.");
        }

        sc.close();
    }

    // Function to reverse a string
    public static String reverseString(String str) {
       String reverse = new StringBuffer(str).reverse().toString();
        return reverse;
    }

    // Function to check if a string is Palindrome
    public static boolean isPalindrome(String str) {
        String reversed = reverseString(str);
        return str.equals(reversed);
    }

    // Function to compare two strings for equality
    public static boolean compareStrings(String str1, String str2) {
        return str1.equals(str2);


      }

}
```

//student details

```
package internal2;
import java.util.Scanner;

class Student {
    String USN;
    String name;
    String branch;
    String phone;

    public Student(String USN, String name, String branch, String phone) {
        this.USN = USN;
        this.name = name;
        this.branch = branch;
        this.phone = phone;
    }

    public void displaydetails() {
        System.out.println("USN: " + USN);
        System.out.println("Name: " + name);
        System.out.println("Branch: " + branch);
```

```
        System.out.println("Phone: " + phone);
        System.out.println();
    }
}

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

        System.out.print("Enter the number of students (n): ");
        int n = sc.nextInt();

        Student[] students = new Student[n];

        for (int i = 0; i < n; i++) {
            sc.nextLine(); // Consume the newline character
            System.out.println("Enter details for Student " + (i + 1) +
":");
            System.out.print("USN: ");
            String USN = sc.nextLine();
            System.out.print("Name: ");
            String name = sc.nextLine();
            System.out.print("Branch: ");
            String branch = sc.nextLine();
            System.out.print("Phone: ");
            String phone = sc.nextLine();

            students[i] = new Student(USN, name, branch, phone);
        }

        System.out.println("\n2Student Details:");
        for (int i = 0; i < n; i++) {
            System.out.println("Student " + (i + 1) + " Details:");
            students[i].displaydetails();
        }

        sc.close();
    }
}
```

Staff

```
package internal2;
import java.util.Scanner;
class Staff{
      protected String StaffID;
      protected String Name;
      protected String Phone;
      protected String Salary;
      public Staff (String StaffID,String Name,String Phone,String Salary)
{
            this.StaffID=StaffID;
            this.Name=Name;
            this.Phone=Phone;
            this.Salary=Salary;
      }
      public void DisplayDetails() {
            System.out.println("StaffID:"+StaffID);
            System.out.println("Name:"+Name);
            System.out.println("Phone:"+Phone);
            System.out.println("Salary:"+Salary);
```

```
	}

}
class Teaching extends Staff{
	protected String Domain;
	protected String Publications;
	public Teaching (String StaffID,String Name,String Phone,String
Salary,String Domain,String Publications) {
		super (StaffID, Name, Phone, Salary);
		this.Domain=Domain;
		this.Publications=Publications;
	}
	@Override
	public void DisplayDetails() {
		super.DisplayDetails();
		System.out.println("Domain:"+Domain);
		System.out.println("Publications:"+Publications);
	}
}
class Contract extends Staff{
	protected String Period;
	public Contract(String StaffID,String Name,String Phone,String
Salary,String Period) {
		super (StaffID, Name, Phone, Salary);
		this.Period =Period;
	}
	@Override
	public void DisplayDetails() {
		super.DisplayDetails();
		System.out.println("Period:"+Period);
	}
}
class Technical extends Staff{
	private String Skills;
	public Technical(String StaffID,String Name,String Phone,String
Salary,String Skills) {
		super (StaffID, Name, Phone, Salary);
		this.Skills =Skills;
	}
	@Override
	public void DisplayDetails() {
		super.DisplayDetails();
		System.out.println("Skills:"+Skills);
	}
}

public class StafID{
	public static void main(String[] args) {
		Scanner sc=new Scanner(System.in);
		Teaching teach  = new Teaching("103"," Rae",
"38294774927","4110","ECE","jaihind");
		Contract cont = new Contract("113", "Rakshith",
"38294774927","45000","2 years");
		Technical tech=new Technical("123", "ram",
"38294774927","45000","Python");
		System.out.println("\nTeaching staff Details:");
		teach.DisplayDetails();
		System.out.println("\nContract  staffDetails;");
		cont.DisplayDetails();
		System.out.println("\nTechnical staff detail:");
		tech.DisplayDetails();
```

```
            sc.close();

      }
}
```

BankAccount

```
package internal2;

import java.util.Scanner;

class BankAccount {
      private double balance;

      public BankAccount(double initialBalance) {
            balance = initialBalance;
      }

      public void deposit(double amount) {

            if (amount > 0) {
                  balance += amount;
                  System.out.println("Deposited ₹" + amount);
            } else {
                  System.out.println("Invalid deposit amount.");
            }
      }

      public void withdraw(double amount) {
            if (amount > 0 && amount <= balance) {
                  balance -= amount;
                  System.out.println("Withdrawn ₹" + amount);
            } else {
                  System.out.println("Invalid withdrawal amount or
insufficient balance.");
            }
      }

      public double getBalance() {
            return balance;
      }
}

class SBAccount extends BankAccount {
      public SBAccount(double initialBalance) {
            super(initialBalance);
      }

      @Override
      public void withdraw(double amount) {
            if (getBalance() - amount >= 100) {
                  super.withdraw(amount);
            } else {
                  System.out.println("Withdrawal not allowed: Minimum
balance of ₹100 must be maintained.");
            }
      }
}

class Bankaccount_overloading {
```

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

		System.out.print("Enter the initial balance for the SBAccount: ");
		double initialBalance = sc.nextDouble();
		SBAccount sbAccount = new SBAccount(initialBalance);

		while (true) {
			System.out.println("\n1. Deposit\n2. Withdraw\n3. Check Balance\n4. Exit");
			System.out.print("Select an option (1/2/3/4): ");
			int choice = sc.nextInt();

			switch (choice) {
			case 1:
				System.out.print("Enter the deposit amount: ₹");
				double depositAmount = sc.nextDouble();
				sbAccount.deposit(depositAmount);
				break;
			case 2:
				System.out.print("Enter the withdrawal amount: ₹");
				double withdrawAmount = sc.nextDouble();
				sbAccount.withdraw(withdrawAmount);
				break;
			case 3:
				System.out.println("Current balance: ₹" + sbAccount.getBalance());
				break;
			case 4:
				System.out.println("Exiting the program.");
				sc.close();
				System.exit(0);
			default:
				System.out.println("Invalid choice. Please select a valid option.");
			}
		}
	}
}
```

//abstract class Bankacnt

```
package internal2;

import java.util.Scanner;
abstract class Bankacnt {

	protected double balance;
	public Bankacnt(double initialBalance) {

		balance = initialBalance;
	}

	public abstract void deposit(double amount);
	public abstract void withdraw(double amount);
	public double getBalance() {
		return balance;
```

```
	}
}

class SavingsAccount extends Bankacnt {

	public SavingsAccount(double initialBalance) {

		super(initialBalance);
	}
	@Override
	public void deposit(double amount) {
		balance += amount;
	}

	@Override
	public void withdraw(double amount) {
		if (balance - amount >= 0) {
			balance -= amount;
		} else {
			System.out.println("Insufficient Balance for
withdrawal.");
		}
	}
}

class CurrentAccount extends Bankacnt{
	public CurrentAccount(double initialBalance) {
		super(initialBalance);
	}
	@Override
	public void deposit(double amount) {
		balance += amount;
	}
	@Override
	public void withdraw(double amount) {
		balance -= amount;
	}
}
class AbstractBankaccount {

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

	System.out.print("Enter initial balance for Savings Account: ");
	double initialBalanceSavings = scanner.nextDouble();
	SavingsAccount savingsAccount = new
SavingsAccount(initialBalanceSavings);

	System.out.print("Enter initial balance for Current Account: ");
	double initialBalanceCurrent = scanner.nextDouble();
	CurrentAccount currentAccount = new
CurrentAccount(initialBalanceCurrent);

	System.out.print("Enter the deposit amount for Savings Account: ");
	double depositAmountSavings = scanner.nextDouble();
	savingsAccount.deposit(depositAmountSavings);

	System.out.print("Enter the withdrawal amount for Savings Account:
");
	double withdrawAmountSavings = scanner.nextDouble();
	savingsAccount.withdraw(withdrawAmountSavings);
```

```
        System.out.print("Enter the deposit amount for Current Account: ");
        double depositAmountCurrent = scanner.nextDouble();
        currentAccount.deposit(depositAmountCurrent);

        System.out.print("Enter the withdrawal amount for Current Account:
");
        double withdrawAmountCurrent = scanner.nextDouble();
        currentAccount.withdraw(withdrawAmountCurrent);

        System.out.println("Savings Account Balance: " +
savingsAccount.getBalance());
        System.out.println("Current Account Balance: " +
currentAccount.getBalance());
        scanner.close();
    }
}
```

overloading

```
package internal2;
class cnst
{
    int i,j;
    cnst()
    {
     i=10;
     j=20;
    }

    cnst(int i,int j)
    {

       this.i=i;
       this.j=j;
    }

    cnst(int x)
    {
       i=x;
       j=20;
    }

    void sum()
    {
       int sum=i+j;
       System.out.println("Numbers are  "+i+" and "+j+" \nSum is ="+sum);
    }

    void sum(int x )
    {
       int sum=x+j;
       System.out.println("Numbers are  "+x+" and "+j+"\nSum is ="+sum);
    }

    void sum(int x ,int y)
    {
       int sum=x+y;
```

```
        System.out.println("Numbers are "+x+" and "+y+" \nSum is = "+sum);

    }
}

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

            cnst ob=new cnst();
            cnst ob1=new cnst(50);
            cnst ob2=new cnst(30,40);
            ob.sum();
            ob1.sum(50);
            ob2.sum(30,40);
      }
}
```

Exception

```
package internal2;

import java.util.Scanner;

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

            // Part 1: Handling ArithmeticException for division by zero
            try {
                  System.out.print("Enter the value of a: ");
                  int a = sc.nextInt();
                  System.out.print("Enter the value of b: ");
                  int b = sc.nextInt();

                  if (b == 0) {
                        throw new ArithmeticException("Division by zero is
not allowed.");
                  }

                  double result = (double) a / b;
                  System.out.println("Result of a/b: " + result);
            } catch (ArithmeticException e) {
                  System.out.println("ArithmeticException: " +
e.getMessage());
            }

            // Part 2: Demonstrating ArrayIndexOutOfBoundsException
            try {
                  int[] numbers = { 1, 2, 3, 4, 5 };
                  System.out.print("Enter an index to access the array (0-
4): ");
                  int index = sc.nextInt();
                  int value = numbers[index];
                  System.out.println("Value at index " + index + ": " +
value);
            } catch (ArrayIndexOutOfBoundsException e) {
                  System.out.println("ArrayIndexOutOfBoundsException: " +
e.getMessage());
```

```
		}

		sc.close();
	}
}
```

Oddnumberchecker

```
package internal2;

public class Oddnumberchecker {

	public static void main(String[] args) {
		// TODO Auto-generated method stub

		try {
			checkEvenNumber(4);
			checkEvenNumber(7);
		} catch (IllegalArgumentException e) {
			System.out.println("IllegalArgumentException :" +
e.getMessage());
		}
	}

	public static void checkEvenNumber(int number) {
		if (number % 2 != 0) {
			throw new IllegalArgumentException("Number Must be
Even");
		}else {
			System.out.println(number+" is an Even number");
		}
	}
}
```

```java
package P1;

public class A {
    private int privateVarA = 10;
    public int publicVarA = 20;
    protected int protectedVarA = 30;
    int defaultVarA = 40;

    public void display() {
        System.out.println("\nClass A - Access Modifiers");
        System.out.println("privateVarA: " + privateVarA);
        System.out.println("publicVarA: " + publicVarA);
        System.out.println("protectedVarA: " + protectedVarA);
        System.out.println("defaultVarA: " + defaultVarA);
    }
}
```

```java
package P1;
public class B extends A {
    public void display() {
        System.out.println("\nClass B - Inherited from A");
        // You can access inherited members
        // privateVarA is not accessible in this class
        System.out.println("publicVarA: " + publicVarA);
        System.out.println("protectedVarA: " + protectedVarA);
        System.out.println("defaultVarA: " + defaultVarA);
    }
}
```

```java
package P1;
public class C {
    public void display() {
        System.out.println("\nClass C - No Inheritance from A");
        A objA = new A();
        // You can access public, protected, and default members of A from
another class in the same package
        System.out.println("publicVarA: " + objA.publicVarA);
        System.out.println("protectedVarA: " + objA.protectedVarA);
        System.out.println("defaultVarA: " + objA.defaultVarA);
    }
}
```

```java
package P2;

import P1.A;

public class D extends A {
    public void display() {
        System.out.println("\nClass D - Inherited from A in P1");
        // You can access inherited members from a different package
        // privateVarA is not accessible in this class
        System.out.println("publicVarA: " + publicVarA);
        System.out.println("protectedVarA: " + protectedVarA);
```

```
        // defaultVarA is not accessible because it's in a different
package
    }
}
```

```
package P2;

import P1.A;

public class E {
    public void display() {
        System.out.println("\nClass E - No Inheritance from A in P1");
        A objA = new A();
        // You can access public members of A in P1 from a different
package
        System.out.println("publicVarA: " + objA.publicVarA);
        // defaultVarA and protectedVarA are not accessible because it's in
a different package
    }
}
```

```
package P1;

class Main {
    public static void main(String[] args) {
        P1.A objA = new P1.A();
        objA.display();

        P1.B objB = new P1.B();
        objB.display();

        P1.C objC = new P1.C();
        objC.display();

        P2.D objD = new P2.D();
        objD.display();

        P2.E objE = new P2.E();
        objE.display();
    }
}
```