# Day 4

**Task1: Array Sorting and Searching**

**1A)** Implement a function called BruteForceSort that sorts an array using the brute force approach. Use this function to sort an array created with Initialize Array.

**Program:**

package Assignments.Day4;

import java.util.Scanner;

public class Task1a {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the number of Elements in the Array: ");

int n = scanner.nextInt();

int [] arr = new int[n];

System.out.println("Enter the Elements in the Array: ");

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

arr[i] = scanner.nextInt();

}

bruteForceSort(arr, n);

}

private static void bruteForceSort(int[] arr, int n) {

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

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

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

int temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

}

System.out.println("Elements After sorted: ");

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

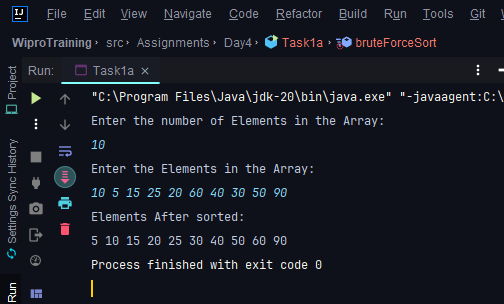
System.out.print(arr[i] + " ");

}

}

}

**Output:**



**1B)** Write a function named PerformLinearSearch that searches for a specific element in an array and returns the index of the element if found or -1 if not found.

**Program:**

package Assignments.Day4;

import java.util.Scanner;

public class Task1b {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter number of elements in an Array: ");

int n = scanner.nextInt();

int [] arr = new int[n];

System.out.println("Enter Array elements: ");

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

arr[i] = scanner.nextInt();

}

System.out.println("Enter the search Element: ");

int searchElement = scanner.nextInt();

System.out.println("The element is at index: "+performLinearSearch(arr, searchElement, n));

}

private static int performLinearSearch(int [] arr, int searchElement, int n){

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

if(arr[i] == searchElement){

return i;

}

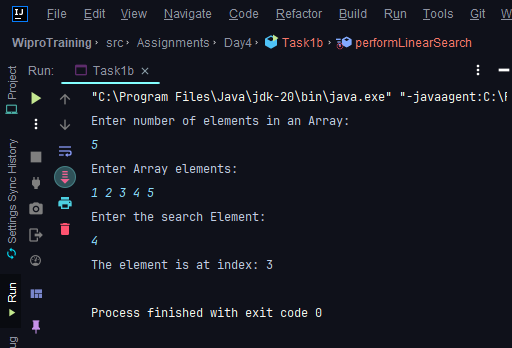
}

return -1;

}

}

**Output:**



**Task2: Two-Sum Problem**

Given an array of integers, write a program that finds if there are two numbers that add up to a specific target. You may assume that each input would have exactly one solution, and you may not use the same element twice. Optimize the solution for time complexity.

**Program:**

package Assignments.Day4;

import java.util.HashMap;

import java.util.Scanner;

public class Task2 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the number of Elements: ");

int n = scanner.nextInt();

int [] arr = new int[n];

System.out.println("Enter the Array Elements: ");

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

arr[i] = scanner.nextInt();

}

int target = scanner.nextInt();

twoSum(arr, target, n);

}

private static void twoSum(int[] arr, int target, int n){

HashMap<Integer,Integer> map = new HashMap<>();

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

int cur = arr[i];

if(map.containsKey(target-cur)){

System.out.println(target + " is the sum of " + map.get(target-cur) + ", "+ i+ " indexes");

break;

}else {

map.put(arr[i],i);

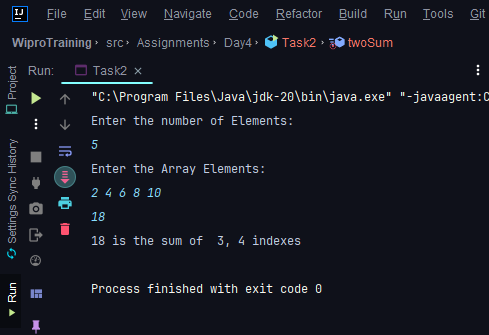
}

}

}

}

**Output:**



**Task 3: Understanding Functions through Arrays**

Write a recursive function named SumArray that calculates and returns the sum of elements in an array, demonstrate with example.

**Program:**

package Assignments.Day4;

//a) Write a recursive function named SumArray that calculates and returns the sum of elements in an array, demonstarte with example.

public class Task3 {

public static void main(String[] args) {

int [] arr = {40, 30, 39, 5, 28 };

int n = arr.length;

System.out.println("Sum Of All Given Array Elements : "+SumArray(arr, 0, n-1));

}

private static int SumArray(int[] arr,int sum, int n){

if(n == -1) return 0; // for empty array

if(n == 0)return sum + arr[n]; // for 1st index

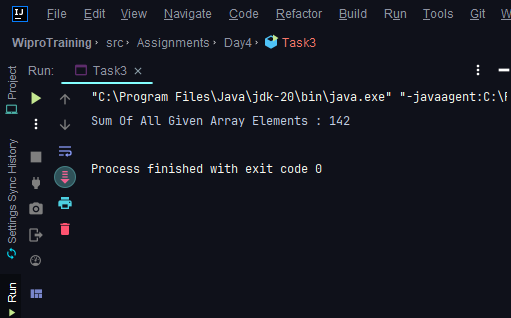
sum += arr[n];

return SumArray(arr, sum, n-1);

}

}

**Output:**



**Task 4: Advanced Array Operations**

**4A)** Implement a method Slice Array that takes an array, a starting index, and an end index, then returns a new array containing the elements from the start to the end index.

**Program:**

package Assignments.Day4;

import java.util.Scanner;

public class Task4a {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the starting Number: ");

int start = scanner.nextInt(); // 5

System.out.println("Enter the end Number: ");

int end = scanner.nextInt(); // 10

int n = end-start;

int [] arr = new int[n+1];

SliceArray(arr, start, end, n);

}

// 5 6 7 8 9 10

private static void SliceArray(int[] arr, int start, int end, int n) {

int i = 0;

while(start <= end){

arr[i] = start;

start++;

i++;

}

for(int j : arr){

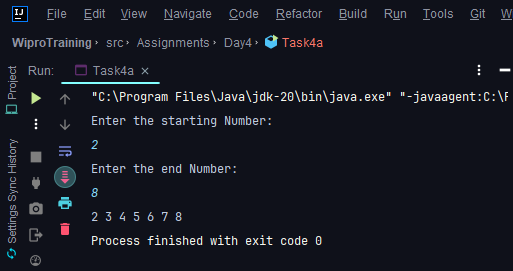
System.out.print(j + " ");

}

}

}

**Output:**



**4B)** Create a recursive function to find the nth element of a Fibonacci sequence and store the first n elements in an array.

**Program:**

package Assignments.Day4;

import java.util.Scanner;

/\*

fibonacci : 0 1 1 2 3 5 8.....

\*/

public class Task4b {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the nth value: ");

int n = scanner.nextInt();

System.out.println(fibonacci(n, 0, 1, 1));

}

private static int fibonacci(int n, int a, int b, int count) {

if(count == n) return a;

int sum = a+b;

a = b;

b = sum;

count++;

return fibonacci(n, a,b, count);

}

}

**Output:**

