

**Task 1:**

**Aim:** To check whether a given integer number is even or odd and return

**Algorithm:**

- 1 Start
- 2 Read an integer n
- 3 Check if  $n \% 2 == 0$ 
  - If true, return 1 (Even)
  - Else, return 0 (Odd)

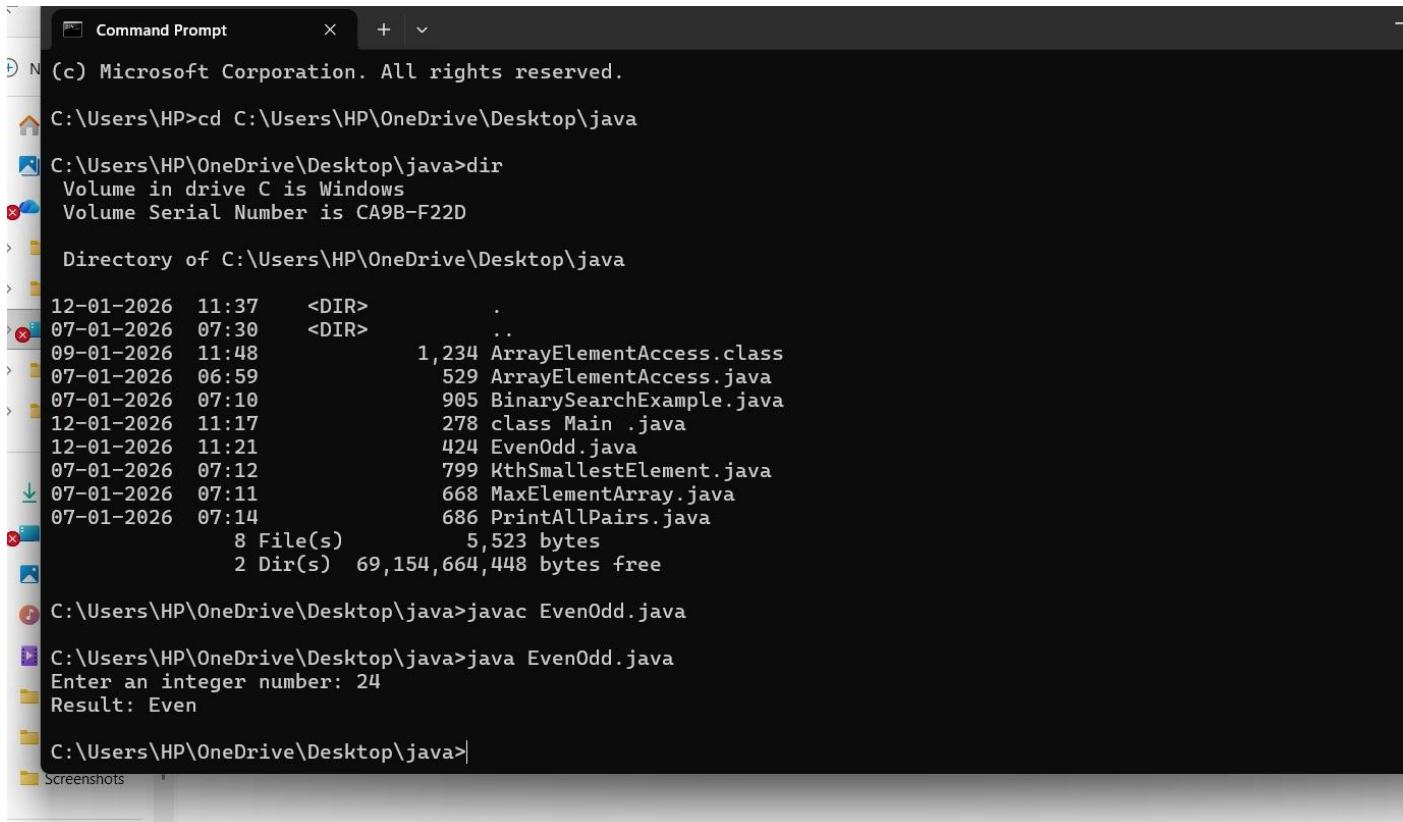
- 4 Print the returned value

- 5 Stop

**Program:**

```
import java.util.*;  
public class Main {  
    static int isEven(int n){  
        if (n % 2 == 0)  
            return 1;  
        else  
            return 0;  
    }  
  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        int n = sc.nextInt();  
        System.out.print(isEven(n));  
        sc.close();  
    }  
}
```

## **Output:**



The screenshot shows a Windows Command Prompt window titled "Command Prompt". The user has navigated to the directory "C:\Users\HP\Desktop\java". They run a "dir" command to list files, which includes "ArrayElementAccess.java", "BinarySearchExample.java", "Main.java", "EvenOdd.java", "KthSmallestElement.java", "MaxElementArray.java", and "PrintAllPairs.java". The "javac EvenOdd.java" command is run, followed by "java EvenOdd.java". The user enters "24" when prompted for an integer number, and the output "Result: Even" is displayed. The window also shows icons for Home, File Explorer, Task View, and Start.

```
C:\Users\HP>cd C:\Users\HP\OneDrive\Desktop\java
C:\Users\HP\OneDrive\Desktop\java>dir
Volume in drive C is Windows
Volume Serial Number is CA9B-F22D
Directory of C:\Users\HP\OneDrive\Desktop\java

12-01-2026  11:37    <DIR>          .
07-01-2026  07:30    <DIR>          ..
09-01-2026  11:48      1,234 ArrayElementAccess.class
07-01-2026  06:59      529 ArrayElementAccess.java
07-01-2026  07:10      905 BinarySearchExample.java
12-01-2026  11:17      278 class Main .java
12-01-2026  11:21      424 EvenOdd.java
07-01-2026  07:12      799 KthSmallestElement.java
07-01-2026  07:11      668 MaxElementArray.java
07-01-2026  07:14      686 PrintAllPairs.java
               8 File(s)       5,523 bytes
               2 Dir(s)  69,154,664,448 bytes free

C:\Users\HP\OneDrive\Desktop\java>javac EvenOdd.java
C:\Users\HP\OneDrive\Desktop\java>java EvenOdd.java
Enter an integer number: 24
Result: Even
C:\Users\HP\OneDrive\Desktop\java>
```

**Result:** Thus finding the Even or odd is shown successfully

**Task2:**

**Aim:** To access and print the element present at a given index in an array.

**Algorithm:**

- 1 Start
- 2 Read the size of the array n
- 3 Read n elements into the array
- 4 Read the index value index
- 5 Check whether index is valid
  - If  $\text{index} \geq 0$  and  $\text{index} < n$ , print the element at that index
  - Else, print an error message (invalid index)
- 6 Stop

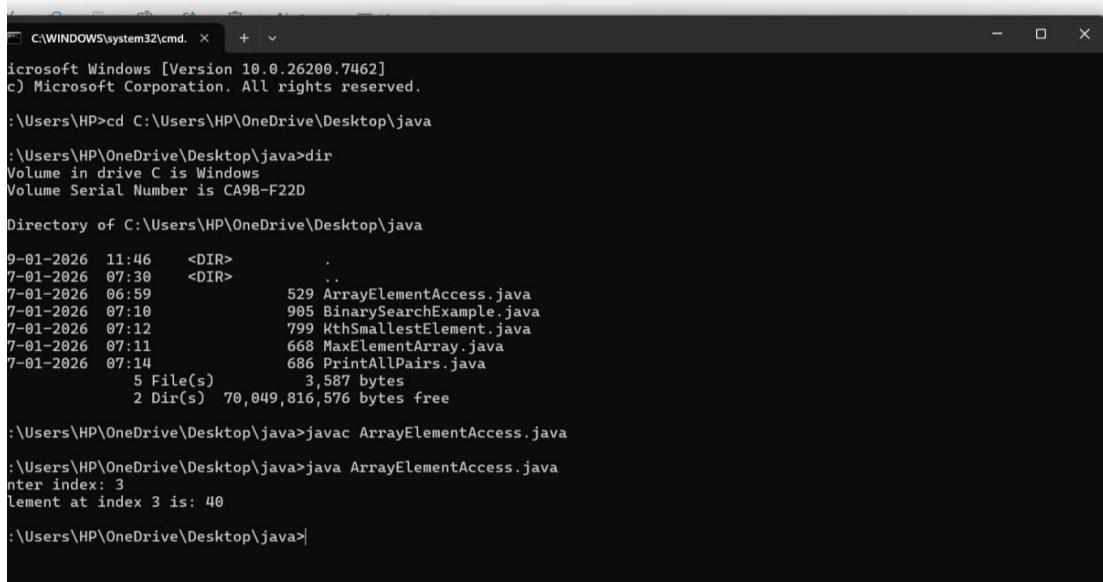
**Program:**

```
import java.util.Scanner;

public class ArrayIndexAccess {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[] arr = {10, 20, 30, 40, 50};
        System.out.print("Enter index: ");
        int index = sc.nextInt();
        if (index >= 0 && index < arr.length) {
            System.out.println("Element at index " + index + " is: " + arr[index]);
        } else {
            System.out.println("Invalid index!");
        }
        sc.close();
    }
}
```

## **Output:**



```
C:\WINDOWS\system32\cmd. x + v
Microsoft Windows [Version 10.0.26200.7462]
c) Microsoft Corporation. All rights reserved.

:C:\Users\HP>cd C:\Users\HP\OneDrive\Desktop\java

:C:\Users\HP\OneDrive\Desktop\java>dir
Volume in drive C is Windows
Volume Serial Number is CA9B-F22D

Directory of C:\Users\HP\OneDrive\Desktop\java

9-01-2026 11:46    <DIR>      .
7-01-2026 07:30    <DIR>      ..
7-01-2026 06:59           529 ArrayElementAccess.java
7-01-2026 07:10           905 BinarySearchExample.java
7-01-2026 07:12           799 KthSmallestElement.java
7-01-2026 07:11           668 MaxElementArray.java
7-01-2026 07:14           686 PrintAllPairs.java
                  5 File(s)   3,587 bytes
                  2 Dir(s)  70,049,816,576 bytes free

:C:\Users\HP\OneDrive\Desktop\java>javac ArrayElementAccess.java
:C:\Users\HP\OneDrive\Desktop\java>java ArrayElementAccess.java
Enter index: 3
lement at index 3 is: 40

:C:\Users\HP\OneDrive\Desktop\java>
```

**Result:** Thus finding the element at index was shown sucessfully

**Task3:**

**Aim:** To search for a given element in a sorted array using Binary Search technique.

**Algorithm:**

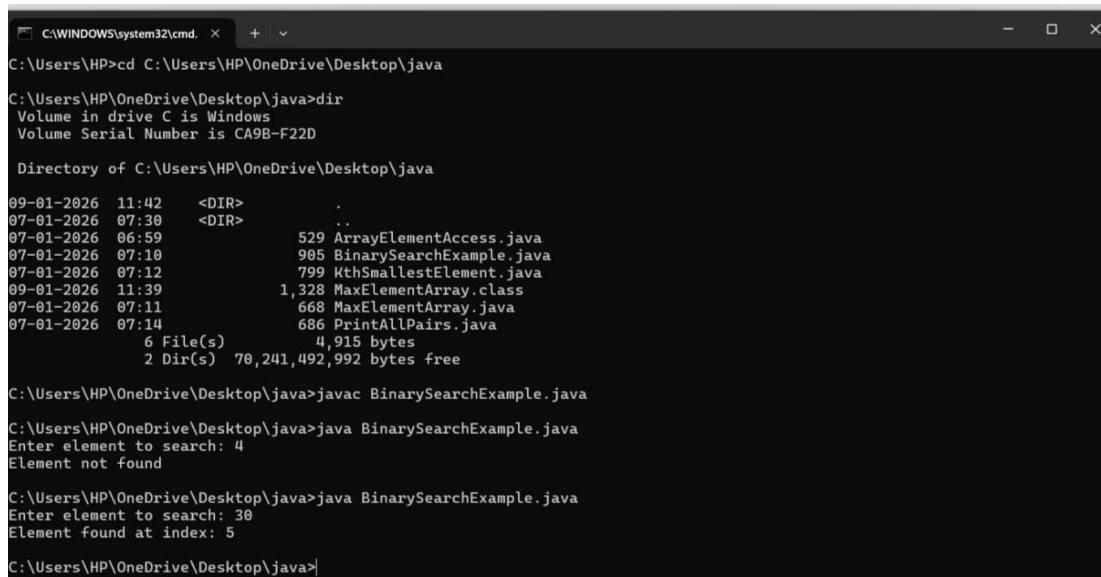
- 1 Start
- 2 Read the number of elements n
- 3 Read the n elements of the sorted array
- 4 Read the element to be searched key
- 5 Initialize low = 0 and high = n – 1
- 6 Repeat while low ≤ high
  1. Calculate mid = (low + high) / 2
  2. If array[mid] == key, print the position and stop
  3. If key < array[mid], set high = mid – 1
  4. Else, set low = mid + 1
- 7 If the element is not found, print “Element not found”
- 8 Stop

**Program:**

```
import java.util.Scanner;  
  
public class BinarySearchExample {  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
  
        int[] arr = {10, 20, 30, 40, 50, 60, 70};  
  
        System.out.print("Enter element to search: ");  
  
        int key = sc.nextInt();  
  
        int low = 0;  
  
        int high = arr.length - 1;  
  
        boolean found = false;
```

```
while (low <= high) {  
    int mid = (low + high) / 2;  
  
    if (arr[mid] == key) {  
        System.out.println("Element found at index: " + mid);  
        found = true;  
        break;  
    } else if (arr[mid] < key) {  
        low = mid + 1;  
    } else {  
        high = mid - 1;  
    }  
}  
  
if (!found) {  
    System.out.println("Element not found");  
}  
sc.close();  
}
```

## **Output:**



```
C:\WINDOWS\system32\cmd. x + v
C:\Users\HP>cd C:\Users\HP\OneDrive\Desktop\java
C:\Users\HP\OneDrive\Desktop\java>dir
 Volume in drive C is Windows
 Volume Serial Number is CA9B-F22D

 Directory of C:\Users\HP\OneDrive\Desktop\java

09-01-2026  11:42    <DIR>          .
07-01-2026  07:30    <DIR>          ..
07-01-2026  06:59           529 ArrayElementAccess.java
07-01-2026  07:10           905 BinarySearchExample.java
07-01-2026  07:12           799 KthSmallestElement.java
09-01-2026  11:39           1,328 MaxElementArray.class
07-01-2026  07:11           668 MaxElementArray.java
07-01-2026  07:14           686 PrintAllPairs.java
07-01-2026      6 File(s)   4,915 bytes
                           2 Dir(s)  70,241,492,992 bytes free

C:\Users\HP\OneDrive\Desktop\java>javac BinarySearchExample.java
C:\Users\HP\OneDrive\Desktop\java>java BinarySearchExample.java
Enter element to search: 4
Element not found

C:\Users\HP\OneDrive\Desktop\java>java BinarySearchExample.java
Enter element to search: 30
Element found at index: 5

C:\Users\HP\OneDrive\Desktop\java>
```

**Result:** Thus finding the element in sorted array using binary search is shown successfully

## **Task4:**

**Aim:** To find the maximum element in an array of n integers.

### **Algorithm:**

- 1 Start
- 2 Read the number of elements n
- 3 Read n integers into the array
- 4 Assume the first element of the array as maximum
  - max = array[0]
- 5 Compare max with each remaining element of the array
  - If any element is greater than max, update max
- 6 After completing the loop, max contains the largest element
- 7 Print the value of max
- 8 Stop

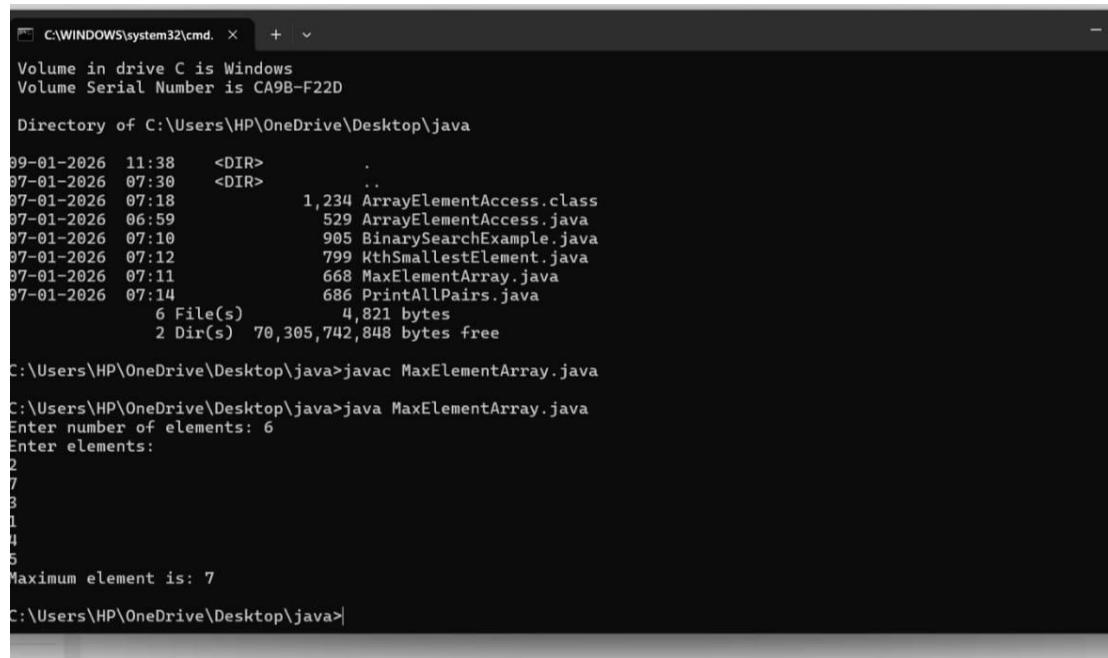
### **Program:**

```
import java.util.Scanner;

public class MaxElementArray {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number of elements: ");
        int n = sc.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter array elements:");
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }
        int max = arr[0]; // assume first element is maximum
        for (int i = 1; i < n; i++) {
            if (arr[i] > max) {
```

**Output:**



```
C:\WINDOWS\system32\cmd. × + ▾ -  
Volume in drive C is Windows  
Volume Serial Number is CA9B-F22D  
Directory of C:\Users\HP\OneDrive\Desktop\java  
09-01-2026 11:38 <DIR> .  
07-01-2026 07:30 <DIR> ..  
07-01-2026 07:18 1,234 ArrayElementAccess.class  
07-01-2026 06:59 529 ArrayElementAccess.java  
07-01-2026 07:10 905 BinarySearchExample.java  
07-01-2026 07:12 799 KthSmallestElement.java  
07-01-2026 07:11 668 MaxElementArray.java  
07-01-2026 07:14 686 PrintAllPairs.java  
   6 File(s)      4,821 bytes  
   2 Dir(s)  70,305,742,848 bytes free  
C:\Users\HP\OneDrive\Desktop\java>javac MaxElementArray.java  
C:\Users\HP\OneDrive\Desktop\java>java MaxElementArray.java  
Enter number of elements: 6  
Enter elements:  
2  
7  
3  
1  
4  
5  
Maximum element is: 7  
C:\Users\HP\OneDrive\Desktop\java>
```

Result: Thus finding the max element was shown successfully

### **Task5:**

**Aim:** To find the Kth smallest element in a given array of integers.

### **Algorithm:**

- 1 Start
- 2 Read the number of elements n
- 3 Read n integers into the array
- 4 Read the positive integer k
- 5 Sort the array in **ascending order**
- 6 The element at index  $k - 1$  is the **Kth smallest element**
- 7 Print the Kth smallest element
- 8 Stop

### **Program:**

```
import java.util.Arrays;  
import java.util.Scanner;  
public class KthSmallest {  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        System.out.print("Enter number of elements: ");  
        int n = sc.nextInt();  
        int[] arr = new int[n];  
        System.out.println("Enter array elements:");  
        for (int i = 0; i < n; i++) {  
            arr[i] = sc.nextInt();  
        }  
        Arrays.sort(arr);  
        System.out.println("Kth smallest element is: " + arr[n-1]);  
    }  
}
```

```

System.out.print("Enter K: ");
int k = sc.nextInt();

Arrays.sort(arr)

if (k > 0 && k <= n) {

    System.out.println("Kth smallest element is: " + arr[k - 1]);

} else {

    System.out.println("Invalid K value");

}

sc.close();

}

}

```

### Output:

```

C:\WINDOWS\system32\cmd. x + v
07-01-2026 07:14      686 PrintAllPairs.java
6 File(s)        4,821 bytes
2 Dir(s)  70,136,410,112 bytes free

C:\Users\HP\OneDrive\Desktop\java> javac KthSmallestElement.java
C:\Users\HP\OneDrive\Desktop\java> java KthSmallestElement.java
Enter number of elements: 5
Enter elements:
4
4
5
6
2
Enter value of K: 9
Invalid value of K
C:\Users\HP\OneDrive\Desktop\java> java KthSmallestElement.java
Enter number of elements: 6
Enter elements:
7
10
4
3
28
15
Enter value of K: 3
Kth smallest element is: 7
C:\Users\HP\OneDrive\Desktop\java>

```

**Result:** Thus Finding the kth smallest element was shown successfully

**Task6:**

**Aim:** To print all possible pairs of elements from an array of size n.

**Algorithm:**

- 1 Start
- 2 Read the number of elements n
- 3 Read n elements into the array
- 4 Use two nested loops to generate pairs
  - Outer loop from i = 0 to n – 1
  - Inner loop from j = i + 1 to n – 1
- 5 Print the pair (array[i], array[j])
- 6 Repeat until all possible pairs are printed
- 7 Stop

**Program:**

```
import java.util.Scanner;  
  
public class PrintAllPairs {  
  
    public static void main(String[] args) {  
  
        Scanner sc = new Scanner(System.in);  
  
        System.out.print("Enter number of elements: ");  
  
        int n = sc.nextInt();  
  
        int[] arr = new int[n];  
  
        System.out.println("Enter array elements:");  
  
        for (int i = 0; i < n; i++) {  
  
            arr[i] = sc.nextInt();  
  
        }  
  
        System.out.println("All possible pairs are:");
```

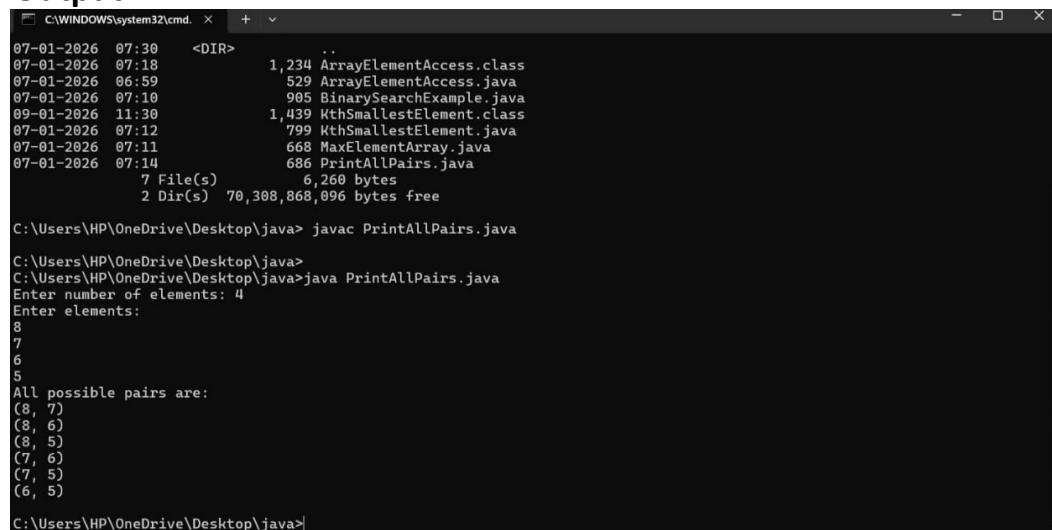
```

for (int i = 0; i < n; i++) {
    for (int j = i + 1; j < n; j++) {
        System.out.println("(" + arr[i] + ", " + arr[j] + ")");
    }
}

sc.close();
}
}

```

**Output:**



The screenshot shows a Windows Command Prompt window titled 'C:\WINDOWS\system32\cmd'. The command line shows the following sequence of operations:

- The directory listing shows several Java files and their corresponding class files.
- The command `javac PrintAllPairs.java` is run to compile the Java source file.
- The command `java PrintAllPairs` is run to execute the compiled Java program.
- The user is prompted to 'Enter number of elements: 4'.
- The user enters the elements: 8, 7, 6, 5.
- The output displays all possible pairs of the entered elements: (8, 7), (8, 6), (8, 5), (7, 6), (7, 5), (6, 5).

**Result: Thus Finding the All pairs of elements was shown Sucessfully**

### **Task7:**

**Aim:** To calculate the sum of even digits or odd digits of a given number based on the user's choice.

### **Algorithm:**

- 1 Start
- 2 Read an integer input1 and a string input2
- 3 Initialize sum = 0
- 4 Repeat while input1 > 0 to
  1. Extract the last digit using digit = input1 % 10
  2. If input2 is "even" and digit is even, add digit to sum
  3. If input2 is "odd" and digit is odd, add digit to sum
  4. Remove the last digit using input1 = input1 / 10
- 5 Return sum
- 6 Stop

### **Program:**

```
class UserMainCode
{
    public int evenodddigitSum(int input1, String input2)
    {
        int sum = 0;
        while (input1 > 0)
        {
            int digit = input1 % 10;

            if (input2.equalsIgnoreCase("even") && digit % 2 == 0)
            {
                sum += digit;
            }
        }
    }
}
```

```

    }

    else if (input2.equalsIgnoreCase("odd") && digit % 2 != 0)

    {

        sum += digit;

    }

input1 = input1 / 10;

}

return sum;

}

```

### Output:

The screenshot shows a Windows Command Prompt window titled "Command Prompt". The command entered was "java SumEvenOddDigits.java". The output shows the program prompting for a number (7), asking if it wants the sum of even or odd digits, and then displaying the result (The sum of odd digits in 7 is: 7).

```

C:\Users\HP\OneDrive\Desktop>java SumEvenOddDigits.java
C:\Users\HP\OneDrive\Desktop>java SumEvenOddDigits.java
ReEnter a number: 7
Do you want the sum of even digits or odd digits? (Enter 'even' or 'odd'): odd
C:\Users\HP\OneDrive\Desktop>The sum of odd digits in 7 is: 7

```

**Result: Thus to calculate sum of even digits was shown sucessfully**

**Task8:**

**Aim:** To find the N<sup>th</sup> Fibonacci number using an iterative approach.

**Algorithm:**

1. Start
2. Read the integer value input1 (position of Fibonacci series)
3. If input1 is equal to 1, return 0
4. If input1 is equal to 2, return 1
5. Initialize two variables:
  - o a = 0 (first Fibonacci number)
  - o b = 1 (second Fibonacci number)
6. Repeat the following steps from i = 3 to input1:  
    Compute c = a + b  
    Assign a = b  
    Assign b = c
7. After the loop ends, return the value of b as the N<sup>th</sup> Fibonacci number
8. Stop

**Program:**

```
class UserMainCode
{
    public int nthFibonacci(int input1)
    {
        if (input1 == 1)
            return 0;
        if (input1 == 2)
            return 1;
```

```

int a = 0, b = 1, c;

for (int i = 3; i <= input1; i++)

{
    c = a + b;
    a = b;
    b = c;
}

return b;
}
}

```

## Output:

The screenshot shows a Windows Command Prompt window titled "Command Prompt". The command line shows the following sequence of commands and output:

```

C:\> Volume in drive C is Windows
C:\> Volume Serial Number is CA9B-F22D
C:\> Directory of C:\Users\HP\OneDrive\Desktop\java
C:\> 12-01-2026 11:51 <DIR> .
C:\> 07-01-2026 07:30 <DIR> ..
C:\> 09-01-2026 11:48 1,234 ArrayElementAccess.class
C:\> 07-01-2026 06:59 529 ArrayElementAccess.java
C:\> 07-01-2026 07:10 905 BinarySearchExample.java
C:\> 12-01-2026 11:17 278 class Main .java
C:\> 12-01-2026 11:38 730 EvenOdd.class
C:\> 12-01-2026 11:21 424 EvenOdd.java
C:\> 12-01-2026 11:51 764 FibonacciIterative.java
C:\> 07-01-2026 07:12 799 KthSmallestElement.java
C:\> 07-01-2026 07:11 668 MaxElementArray.java
C:\> 07-01-2026 07:14 686 PrintAllPairs.java
C:\> 12-01-2026 11:47 1,578 SumEvenOddDigits.class
C:\> 12-01-2026 11:46 1,037 SumEvenOddDigits.java
C:\> 12-01-2026 11:44 1,037 SumEvenOddDigits.txt
C:\> 13 File(s) 10,669 bytes
C:\> 2 Dir(s) 69,138,251,776 bytes free
C:\> C:\Users\HP\OneDrive\Desktop\java> javac FibonacciIterative.java
C:\> C:\Users\HP\OneDrive\Desktop\java> java FibonacciIterative.java
C:\> Enter N: 4
C:\> Fibonacci number at position 4 is 3
C:\> C:\Users\HP\OneDrive\Desktop\java>

```

**Result: Thus finding the Nth Fibonacci was shown sucessfully**

**Task9:**

**AIM:** To check whether a given number is a palindrome number and return

**Algorithm:**

1. Start
2. Read the integer value input1
3. Store the value of input1 in a variable original
4. Initialize reverse = 0
5. Repeat the following steps while input1 > 0:
  1. Extract the last digit using digit = input1 % 10
  2. Form the reversed number using reverse = reverse \* 10 + digit
  3. Remove the last digit using input1 = input1 / 10
6. Compare original with reverse
  - o If both are equal, return 2 (palindrome)
  - o Otherwise, return 1 (not palindrome)
7. Stop

**Program:**

```
class UserMainCode
{
    public int isPalinNum(int input1)
    {
        int original = input1;
        int reverse = 0;

        while (input1 > 0)
        {
            int digit = input1 % 10;
```

```

        reverse = reverse * 10 + digit;

        input1 = input1 / 10;

    }

}

if (original == reverse)

    return 2; // palindrome

else

    return 1; // not palindrome

}

```

### Output:

```

Directory of C:\Users\HP\OneDrive\Desktop\java

12-01-2026 11:56    <DIR>      .
07-01-2026 07:30    <DIR>      ..
09-01-2026 11:48          1,234 ArrayElementAccess.class
07-01-2026 06:59          529 ArrayElementAccess.java
07-01-2026 07:10          905 BinarySearchExample.java
12-01-2026 11:17          278 class Main.java
12-01-2026 11:38          730 EvenOdd.class
12-01-2026 11:21          424 EvenOdd.java
12-01-2026 11:52          1,320 FibonacciIterative.class
12-01-2026 11:51          764 FibonacciIterative.java
07-01-2026 07:12          799 KthSmallestElement.java
07-01-2026 07:11          668 MaxElementArray.java
12-01-2026 12:00          797 PalindromeNumber.java
07-01-2026 07:14          686 PrintAllPairs.java
12-01-2026 11:47          1,578 SumEvenOddDigits.class
12-01-2026 11:46          1,037 SumEvenOddDigits.java
12-01-2026 11:44          1,037 SumEvenOddDigits.txt
15 File(s)           12,786 bytes
2 Dir(s)   68,697,903,104 bytes free

C:\Users\HP\OneDrive\Desktop\java> javac PalindromeNumber.java
C:\Users\HP\OneDrive\Desktop\java> java PalindromeNumber.java
Enter a number: 6
6 is a palindrome number.

C:\Users\HP\OneDrive\Desktop\java>

```

**Result: Thus checking the palindrome was shown successfully**

**Task10:**

**AIM:** To find the sum of the last digits of two given integers.

**Algorithm:**

1. Start
2. Read two integer values input1 and input2
3. Convert both numbers to their absolute values to handle negative inputs
4. Find the last digit of the first number using  $\text{last1} = \text{input1} \% 10$
5. Find the last digit of the second number using  $\text{last2} = \text{input2} \% 10$
6. Calculate the sum of the last digits
  - o  $\text{sum} = \text{last1} + \text{last2}$
7. Return the value of sum
8. Stop

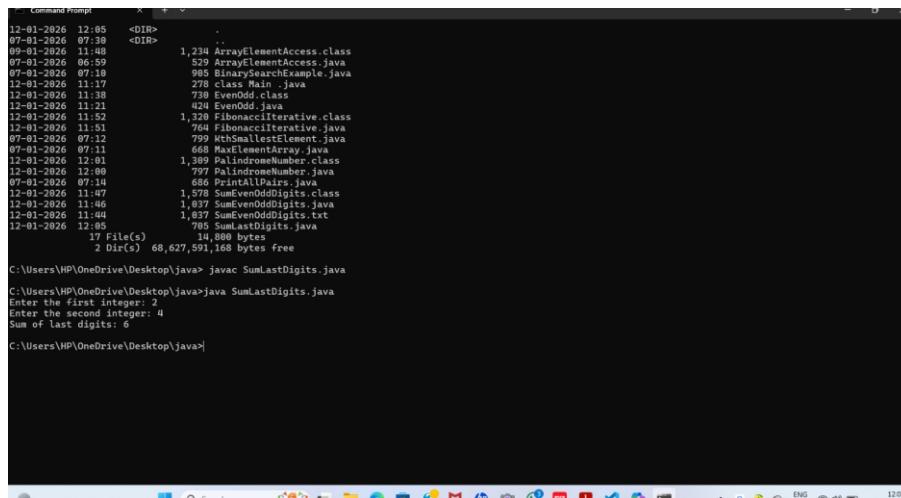
**Program:**

```
class UserMainCode
{
    public int addLastDigits(int input1, int input2)
    {
        input1 = Math.abs(input1);
        input2 = Math.abs(input2);

        int last1 = input1 % 10;
        int last2 = input2 % 10;

        return last1 + last2;
    }
}
```

## OUTPUT:



The screenshot shows a Windows Command Prompt window. The title bar says "Command Prompt". The content of the window is as follows:

```
12-01-2026 12:05  <DIR> .
07-01-2026 07:30  <DIR> .
09-01-2026 11:48      1,276 ArrayElementAccess.class
07-01-2026 06:59      929 ArrayElementAccess.java
07-01-2026 11:10      985 BinarySearchExample.java
12-01-2026 11:17      278 class Main.java
12-01-2026 11:38      730 EvenOdd.class
12-01-2026 11:21      424 EvenOdd.java
12-01-2026 11:52      1,132 EvenOddIterative.class
12-01-2026 11:51      764 FibonacciIterative.java
07-01-2026 07:12      799 KthSmallestElement.java
07-01-2026 07:11      668 MaxElementArray.java
12-01-2026 12:01      1,399 PalindromeNumber.class
12-01-2026 12:00      797 PalindromeNumber.java
07-01-2026 07:14      686 PrintAllPairs.java
12-01-2026 11:47      1,578 SumEvenOddDigits.class
12-01-2026 11:46      1,837 SumEvenOddDigits.java
12-01-2026 11:44      1,671 SumEvenOddDigits.txt
12-01-2026 12:05      785 SumLastDigits.java
17 File(s)   14,888 bytes free
2 Dir(s)   68,627,591,168 bytes free

C:\Users\HP\Desktop>java SumLastDigits.java
C:\Users\HP\OneDrive\Desktop>java SumLastDigits.java
Enter the first integer: 2
Enter the second integer: 4
Sum of last digits: 6

C:\Users\HP\OneDrive\Desktop>
```

**Result:** Thus sum of the last two digits was shown sucessfully