

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:

```
(c) Microsoft Corporation. All rights reserved.

C:\Users\jampa>cd C:\Users\jampa\OneDrive\Documents\java tasks

C:\Users\jampa\OneDrive\Documents\java tasks>dir
 Volume in drive C is Windows
 Volume Serial Number is 3AFB-8EC9

 Directory of C:\Users\jampa\OneDrive\Documents\java tasks

09-01-2026  10:01      <DIR>          .
07-01-2026  12:00      <DIR>          ..
06-01-2026  20:51            1,230 ArrayIndexAccess.class
06-01-2026  20:49            632 ArrayIndexAccess.java
06-01-2026  20:53            1,413 BinarySearchExample.class
06-01-2026  20:52            934 BinarySearchExample.java
06-01-2026  20:57            1,419 KthSmallest.class
06-01-2026  20:56            825 KthSmallest.java
06-01-2026  20:55            1,334 MaxElementArray.class
06-01-2026  20:54            711 MaxElementArray.java
06-01-2026  21:05            1,362 PrintAllPairs.class
06-01-2026  21:04            696 PrintAllPairs.java
09-01-2026  10:01            363 task1.java
                           11 File(s)       10,919 bytes
                           2 Dir(s)   42,964,537,344 bytes free

C:\Users\jampa\OneDrive\Documents\java tasks>javac task1.java

C:\Users\jampa\OneDrive\Documents\java tasks>java task1
6
1
C:\Users\jampa\OneDrive\Documents\java tasks>|
```

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:\Users\jampa\OneDrive\Documents\java tasks>dir
 Volume in drive C is Windows
 Volume Serial Number is 3AFB-8EC9

 Directory of C:\Users\jampa\OneDrive\Documents\java tasks

09-01-2026  10:02    <DIR>          .
07-01-2026  12:00    <DIR>          ..
06-01-2026  20:51          1,230 ArrayIndexAccess.class
06-01-2026  20:49          632 ArrayIndexAccess.java
06-01-2026  20:53          1,413 BinarySearchExample.class
06-01-2026  20:52          934 BinarySearchExample.java
06-01-2026  20:57          1,419 KthSmallest.class
06-01-2026  20:56          825 KthSmallest.java
06-01-2026  20:55          1,334 MaxElementArray.class
06-01-2026  20:54          711 MaxElementArray.java
06-01-2026  21:05          1,362 PrintAllPairs.class
06-01-2026  21:04          696 PrintAllPairs.java
09-01-2026  10:02          663 task1.class
09-01-2026  10:01          363 task1.java
               12 File(s)      11,582 bytes
                2 Dir(s)  42,958,155,776 bytes free

C:\Users\jampa\OneDrive\Documents\java tasks>javac ArrayIndexAccess.java
C:\Users\jampa\OneDrive\Documents\java tasks>java ArrayIndexAccess
Enter index: 1
Element at index 1 is: 20

C:\Users\jampa\OneDrive\Documents\java tasks>
```

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:\Users\jampa\OneDrive\Documents\java tasks>dir
 Volume in drive C is Windows
 Volume Serial Number is 3AFB-8EC9

 Directory of C:\Users\jampa\OneDrive\Documents\java tasks

09-01-2026  10:02    <DIR>        .
07-01-2026  12:00    <DIR>        ..
09-01-2026  10:12            1,230 ArrayIndexAccess.class
06-01-2026  20:49            632 ArrayIndexAccess.java
06-01-2026  20:53            1,413 BinarySearchExample.class
06-01-2026  20:52            934 BinarySearchExample.java
06-01-2026  20:57            1,419 KthSmallest.class
06-01-2026  20:56            825 KthSmallest.java
06-01-2026  20:55            1,334 MaxElementArray.class
06-01-2026  20:54            711 MaxElementArray.java
06-01-2026  21:05            1,362 PrintAllPairs.class
06-01-2026  21:04            696 PrintAllPairs.java
09-01-2026  10:17            663 task1.class
09-01-2026  10:01            363 task1.java
               12 File(s)      11,582 bytes
                2 Dir(s)  42,945,163,264 bytes free

C:\Users\jampa\OneDrive\Documents\java tasks>javac BinarySearchExample.java

C:\Users\jampa\OneDrive\Documents\java tasks>java BinarySearchExample
Enter element to search: 3
Element not found
```

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:

```
Volume Serial Number is 3AFB-8EC9

Directory of C:\Users\jampa\OneDrive\Documents\java tasks

09-01-2026  10:02      <DIR>          .
07-01-2026  12:00      <DIR>          ..
09-01-2026  10:12          1,230 ArrayIndexAccess.class
06-01-2026  20:49          632 ArrayIndexAccess.java
09-01-2026  10:18          1,413 BinarySearchExample.class
06-01-2026  20:52          934 BinarySearchExample.java
06-01-2026  20:57          1,419 KthSmallest.class
06-01-2026  20:56          825 KthSmallest.java
06-01-2026  20:55          1,334 MaxElementArray.class
06-01-2026  20:54          711 MaxElementArray.java
06-01-2026  21:05          1,362 PrintAllPairs.class
06-01-2026  21:04          696 PrintAllPairs.java
09-01-2026  10:17          663 task1.class
09-01-2026  10:01          363 task1.java
               12 File(s)       11,582 bytes
               2 Dir(s)   43,026,292,736 bytes free

C:\Users\jampa\OneDrive\Documents\java tasks>javac MaxElementArray.java

C:\Users\jampa\OneDrive\Documents\java tasks>java MaxElementArray
Enter number of elements: 2
Enter array elements:
10 20 30
Maximum element is: 20
```

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();  
  
        }  
    }  
}
```

```

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:

```

Volume Serial Number is 3AFB-8EC9

Directory of C:\Users\jampa\OneDrive\Documents\java tasks

09-01-2026  10:02    <DIR>      .
07-01-2026  12:00    <DIR>      ..
09-01-2026  10:12          1,230 ArrayIndexAccess.class
06-01-2026  20:49          632 ArrayIndexAccess.java
09-01-2026  10:18          1,413 BinarySearchExample.class
06-01-2026  20:52          934 BinarySearchExample.java
06-01-2026  20:57          1,419 KthSmallest.class
06-01-2026  20:56          825 KthSmallest.java
09-01-2026  10:20          1,334 MaxElementArray.class
06-01-2026  20:54          711 MaxElementArray.java
06-01-2026  21:05          1,362 PrintAllPairs.class
06-01-2026  21:04          696 PrintAllPairs.java
09-01-2026  10:17          663 task1.class
09-01-2026  10:01          363 task1.java
               12 File(s)      11,582 bytes
               2 Dir(s)   43,017,932,800 bytes free

C:\Users\jampa\OneDrive\Documents\java tasks>javac KthSmallest.java

C:\Users\jampa\OneDrive\Documents\java tasks>java KthSmallest
Enter number of elements: 2
Enter array elements:
10 20 30
Enter K: Invalid K value

```

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:

```

07-01-2026 12:00      <DIR>          ..
09-01-2026 10:12          1,230 ArrayIndexAccess.class
06-01-2026 20:49          632 ArrayIndexAccess.java
09-01-2026 10:18          1,413 BinarySearchExample.class
06-01-2026 20:52          934 BinarySearchExample.java
09-01-2026 10:22          1,419 KthSmallest.class
06-01-2026 20:56          825 KthSmallest.java
09-01-2026 10:20          1,334 MaxElementArray.class
06-01-2026 20:54          711 MaxElementArray.java
06-01-2026 21:05          1,362 PrintAllPairs.class
06-01-2026 21:04          696 PrintAllPairs.java
09-01-2026 10:17          663 task1.class
09-01-2026 10:01          363 task1.java
              12 File(s)     11,582 bytes
              2 Dir(s)   43,018,153,984 bytes free

C:\Users\jampa\OneDrive\Documents\java tasks>javac PrintAllPairs.java

C:\Users\jampa\OneDrive\Documents\java tasks>java PrintAllPairs
Error: Could not find or load main class PrintAllPairs
Caused by: java.lang.ClassNotFoundException: PrintAllPairs

C:\Users\jampa\OneDrive\Documents\java tasks>java PrintAllPairs
Enter number of elements: 2
Enter array elements:
10 20 30
All possible pairs are:
(10, 20)

C:\Users\jampa\OneDrive\Documents\java tasks>

```

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:

```

Directory of C:\Users\jampa\OneDrive\Documents\java tasks

10-01-2026  17:32      <DIR>          .
10-01-2026  16:34      <DIR>          ..
09-01-2026  10:12          1,230 ArrayIndexAccess.class
06-01-2026  20:49          632 ArrayIndexAccess.java
09-01-2026  10:18          1,413 BinarySearchExample.class
06-01-2026  20:52          934 BinarySearchExample.java
09-01-2026  10:30          533 evenodddigitSum.java
09-01-2026  10:22          1,419 KthSmallest.class
06-01-2026  20:56          825 KthSmallest.java
09-01-2026  10:20          1,334 MaxElementArray.class
06-01-2026  20:54          711 MaxElementArray.java
09-01-2026  10:34          409 nthFibonacci.java
09-01-2026  10:24          1,362 PrintAllPairs.class
06-01-2026  21:04          696 PrintAllPairs.java
09-01-2026  10:17          663 task1.class
09-01-2026  10:01          363 task1.java
10-01-2026  17:37          772 UserMainCode.class
10-01-2026  17:42          946 UserMainCode.java
                           16 File(s)       14,242 bytes
                           2 Dir(s)   41,513,451,520 bytes free

C:\Users\jampa\OneDrive\Documents\java tasks>javac UserMainCode.java
C:\Users\jampa\OneDrive\Documents\java tasks>java UserMainCode
12345 even
6

```

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

Task8:

Aim: To find the Nth 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 Nth 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:

```

Directory of C:\Users\jampa\OneDrive\Documents\java tasks

10-01-2026  17:32      <DIR>          .
10-01-2026  16:34      <DIR>          ..
09-01-2026  10:12          1,230 ArrayIndexAccess.class
06-01-2026  20:49          632 ArrayIndexAccess.java
09-01-2026  10:18          1,413 BinarySearchExample.class
06-01-2026  20:52          934 BinarySearchExample.java
09-01-2026  10:30          533 evenodddigitSum.java
09-01-2026  10:22          1,419 KthSmallest.class
06-01-2026  20:56          825 KthSmallest.java
09-01-2026  10:20          1,334 MaxElementArray.class
06-01-2026  20:54          711 MaxElementArray.java
09-01-2026  10:34          409 nthFibonacci.java
09-01-2026  10:24          1,362 PrintAllPairs.class
06-01-2026  21:04          696 PrintAllPairs.java
09-01-2026  10:17          663 task1.class
09-01-2026  10:01          363 task1.java
10-01-2026  17:43          974 UserMainCode.class
10-01-2026  17:46          672 UserMainCode.java
               16 File(s)       14,170 bytes
               2 Dir(s)   41,511,624,704 bytes free

C:\Users\jampa\OneDrive\Documents\java tasks>javac UserMainCode.java
C:\Users\jampa\OneDrive\Documents\java tasks>java UserMainCode
7
8

```

Result:Thus finding the Nth Fibonaci 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:

```

Volume in drive C is Windows
Volume Serial Number is 3AFB-8EC9

Directory of C:\Users\jampa\OneDrive\Documents\java tasks

10-01-2026  17:32    <DIR>          .
10-01-2026  16:34    <DIR>          ..
09-01-2026  10:12          1,230 ArrayIndexAccess.class
06-01-2026  20:49          632 ArrayIndexAccess.java
09-01-2026  10:18          1,413 BinarySearchExample.class
06-01-2026  20:52          934 BinarySearchExample.java
09-01-2026  10:30          533 evenodddigitSum.java
09-01-2026  10:22          1,419 KthSmallest.class
06-01-2026  20:56          825 KthSmallest.java
09-01-2026  10:20          1,334 MaxElementArray.class
06-01-2026  20:54          711 MaxElementArray.java
09-01-2026  10:34          409 nthFibonacci.java
09-01-2026  10:24          1,362 PrintAllPairs.class
06-01-2026  21:04          696 PrintAllPairs.java
09-01-2026  10:17          663 task1.class
09-01-2026  10:01          363 task1.java
10-01-2026  17:33          582 UserMainCode.class
10-01-2026  17:36          750 UserMainCode.java
               16 File(s)   13,856 bytes
               2 Dir(s)  41,527,918,592 bytes free

C:\Users\jampa\OneDrive\Documents\java tasks>javac UserMainCode.java
C:\Users\jampa\OneDrive\Documents\java tasks>java UserMainCode
121
2

```

Result:Thus checking the palindrome was shown sucessfully

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:

```
Directory of C:\Users\jampa\OneDrive\Documents\java tasks

10-01-2026  17:32    <DIR>          .
10-01-2026  16:34    <DIR>          ..
09-01-2026  10:12          1,230 ArrayIndexAccess.class
06-01-2026  20:49          632 ArrayIndexAccess.java
09-01-2026  10:18          1,413 BinarySearchExample.class
06-01-2026  20:52          934 BinarySearchExample.java
09-01-2026  10:30          533 evenodddigitSum.java
09-01-2026  10:22          1,419 KthSmallest.class
06-01-2026  20:56          825 KthSmallest.java
09-01-2026  10:20          1,334 MaxElementArray.class
06-01-2026  20:54          711 MaxElementArray.java
09-01-2026  10:34          409 nthFibonacci.java
09-01-2026  10:24          1,362 PrintAllPairs.class
06-01-2026  21:04          696 PrintAllPairs.java
09-01-2026  10:17          663 task1.class
09-01-2026  10:01          363 task1.java
10-01-2026  17:31          362 UserMainCode.class
10-01-2026  17:32          465 UserMainCode.java
                           16 File(s)      13,351 bytes
                           2 Dir(s)   41,533,677,568 bytes free

C:\Users\jampa\OneDrive\Documents\java tasks>javac UserMainCode.java

C:\Users\jampa\OneDrive\Documents\java tasks>java UserMainCode
9

C:\Users\jampa\OneDrive\Documents\java tasks>
```

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