Competitive Programming Lab - 11

Academic year: 2020-2021 **Semester:** Long Sem

Faculty Name: Dr. D Sumathi mam **Date:** 16/ 7/ 2022

Student name: Taran Mamidala **Reg. no.:** 19BCE7346

TWO SUM PROBLEM :

Q1.) This is a classic algorithmic interview question. There are many different solution routes, each of which involves a different technique. This handout details the problem and gives a few different solution routes.

Problem Statement:

You are given an array of n integers and a number k . Determine whether there is a pair of elements in the array that sums to exactly k . For example, given the array [1, 3, 7] and
k = 8, the answer is "yes," but given k = 6 the answer is "no."

Possible Follow-Up Questions:

•Can you modify the array? Yes, that's fine.

•Do we know something about the range of the numbers in the array? No, they can be arbitrary integers.

•Are the array elements necessarily positive? No, they can be positive, negative, or zero.

•Do we know anything about the value of k relative to n or the numbers in the array? No, it can be arbitrary.

•Can we consider pairs of an element and itself? No, the pair should consist of two different array elements.

•Can the array contain duplicates? Sure, that's a possibility.

•Is the array necessarily in sorted order? No, that's not guaranteed.

•What about integer overflow? For simplicity, don't worry about this. What to Watch For

•Make sure their solution works on arrays of zero or one element (it should always return false.)

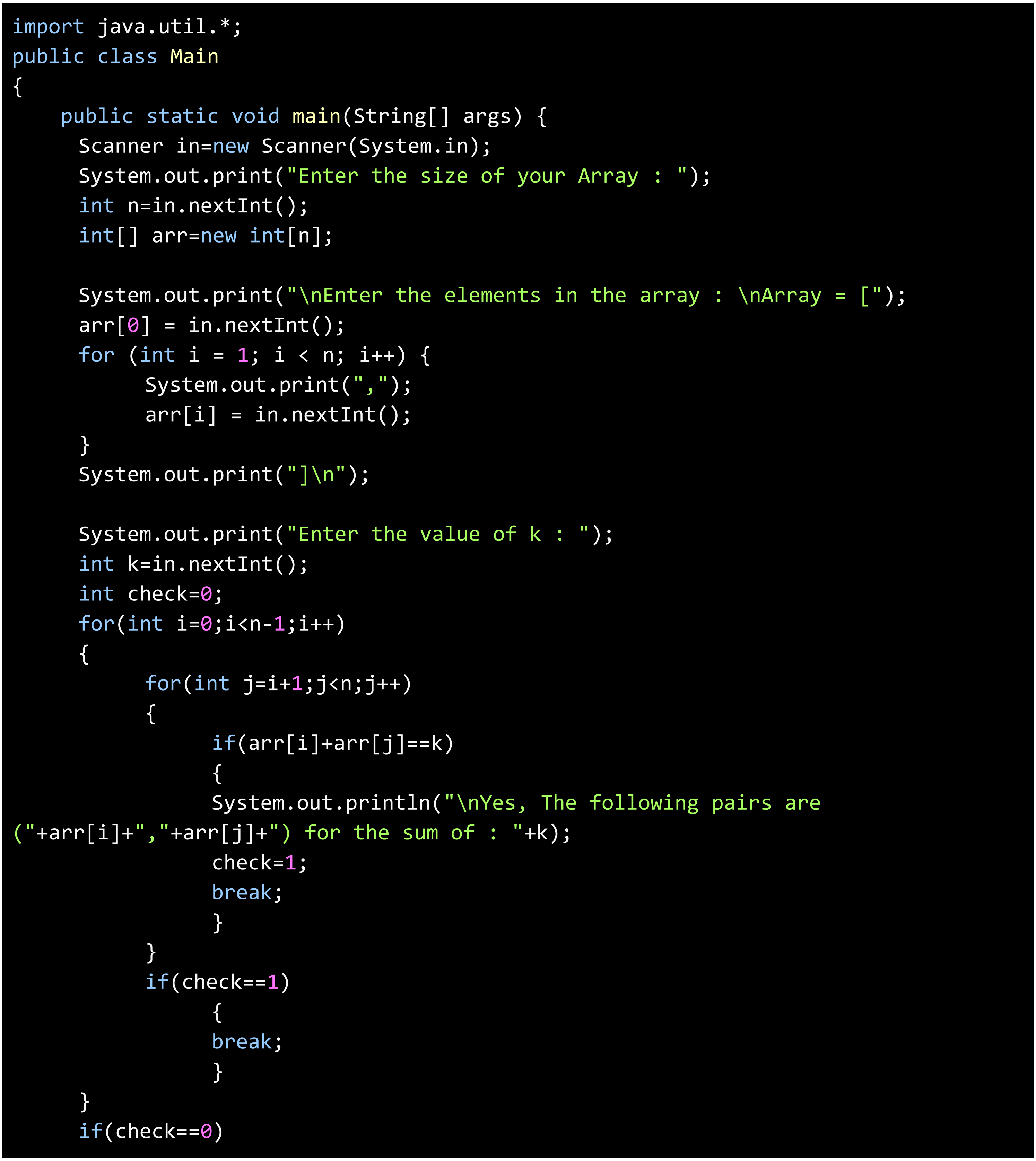
•Make sure their solution always returns a value.

•Make sure their solution works with both positive and negative numbers.

•Make sure their solution works if the only way to make the sum is to use an element that appears twice in the array (for example, [3, 3] with k=6)

Make sure their solution works if the answer is "no" but k/2 is present in the array (for example, [1, 3] with k = 6) should return false.

CODE :



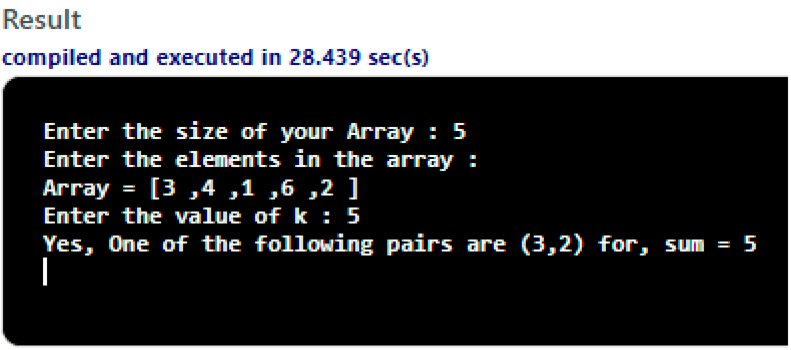
```
import java.util.*;
public class Main
{
    public static void main(String[] args) {
      Scanner in=new Scanner(System.in);
      System.out.print("Enter the size of your Array : ");
      int n=in.nextInt();
      int[] arr=new int[n];

      System.out.print("\nEnter the elements in the array : \nArray = [");
      arr[0] = in.nextInt();
      for (int i = 1; i < n; i++) {
           System.out.print(",");
           arr[i] = in.nextInt();
      }
      System.out.print("]\n");

      System.out.print("Enter the value of k : ");
      int k=in.nextInt();
      int check=0;
      for(int i=0;i<n-1;i++)
      {
           for(int j=i+1;j<n;j++)
           {
                if(arr[i]+arr[j]==k)
                {
                System.out.println("\nYes, The following pairs are
("+arr[i]+","+arr[j]+") for the sum of : "+k);
                check=1;
                break;
                }
           }
           if(check==1)
                {
                break;
                }
      }
      if(check==0)
```

```
        {
            System.out.println("\nNO pairs found for the given sum.");
        }
      }
}
```

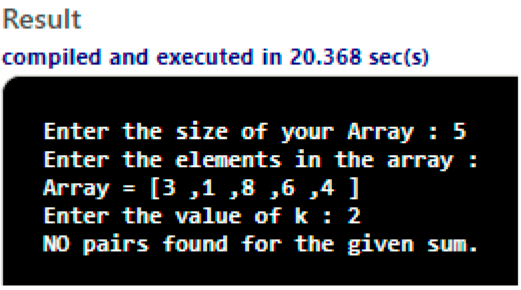
OUTPUT:



Result

compiled and executed in 28.439 sec(s)

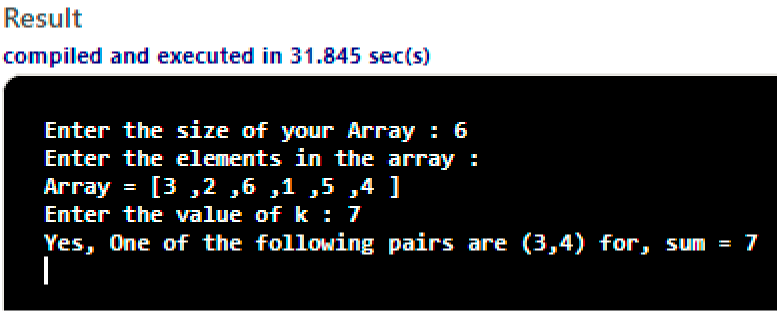
```
Enter the size of your Array : 5
Enter the elements in the array :
Array = [3 ,4 ,1 ,6 ,2 ]
Enter the value of k : 5
Yes, One of the following pairs are (3,2) for, sum = 5
```



Result

compiled and executed in 20.368 sec(s)

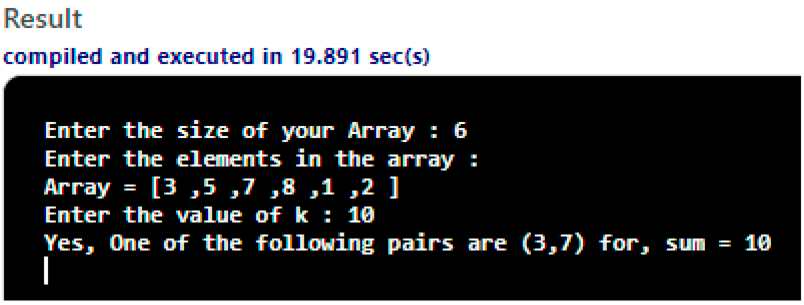
```
Enter the size of your Array : 5
Enter the elements in the array :
Array = [3 ,1 ,8 ,6 ,4 ]
Enter the value of k : 2
NO pairs found for the given sum.
```



Result

compiled and executed in 31.845 sec(s)

```
Enter the size of your Array : 6
Enter the elements in the array :
Array = [3 ,2 ,6 ,1 ,5 ,4 ]
Enter the value of k : 7
Yes, One of the following pairs are (3,4) for, sum = 7
```



Result

compiled and executed in 19.891 sec(s)

```
Enter the size of your Array : 6
Enter the elements in the array :
Array = [3 ,5 ,7 ,8 ,1 ,2 ]
Enter the value of k : 10
Yes, One of the following pairs are (3,7) for, sum = 10
```

N PAIRS OF BALANCED PARENTHESES

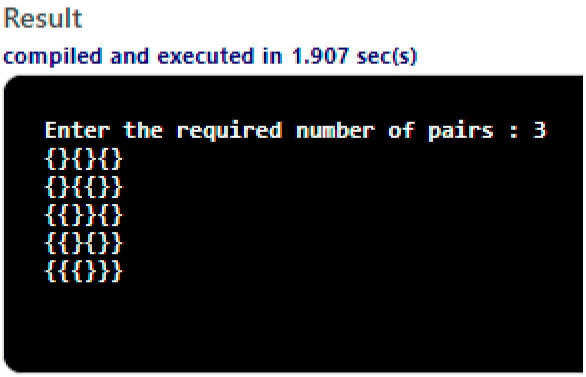
Q2.) Generate all strings of n pairs of balanced parentheses. For example, if n = 3, you'd generate the strings ((())), (()()), (())(), ()(()), ()()().

CODE:

```
import java.io.*;
import java.util.*;
public class Main {
    static void printParen(char str[], int pos, int n, int openParen, int
closeParen)
    {
    if (closeParen == n)
    {
        for (int i = 0; i < str.length; i++)
            System.out.print(str[i]);
        System.out.println();
        return;
    }
    else {
        if (openParen > closeParen) {
            str[pos] = '}';
            printParen(str, pos + 1, n, openParen,
                        closeParen + 1);
        }
        if (openParen < n) {
            str[pos] = '{';
            printParen(str, pos + 1, n, openParen + 1,
                        closeParen);
        }
    }
    }
    static void getParenthesis(char str[], int n)
    {
    if (n > 0)
        printParen(str, 0, n, 0, 0);
    return;
    }
    public static void main(String[] args)
    {
    Scanner in=new Scanner(System.in);
    System.out.println("Enter the required number of pairs : ");
    int n = in.nextInt();
    char[] str = new char[2 * n];
    getParenthesis(str, n);
```

```
        }
}
```

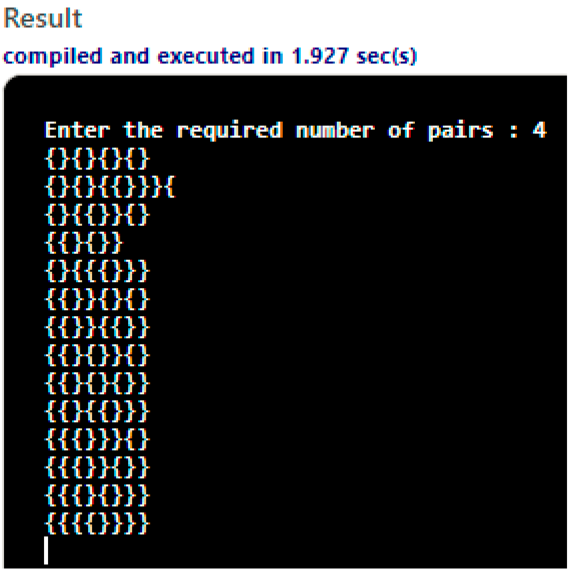
OUTPUT:



Result

compiled and executed in 1.907 sec(s)

```
Enter the required number of pairs : 3
{}{}{}
{}{{}}
{{}}{}
{{}{}}
{{{}}}
```



Result

compiled and executed in 1.927 sec(s)

```
Enter the required number of pairs : 4
{}{}{}{}
{}{}{{}}}{
{}{{}}{}
{{}{}}
{}{{{}}}
{{}}{}{}
{{}}{{}}
{{}{}}{}
{{}{}{}}
{{}{{}}}
{{{}}}{}
{{{}}{}}
{{{}{}}}
{{{{}}}}
```

MAX PATH IN TRIANGLE

Q3.) You are given a pyramid of numbers like the one shown here:

137

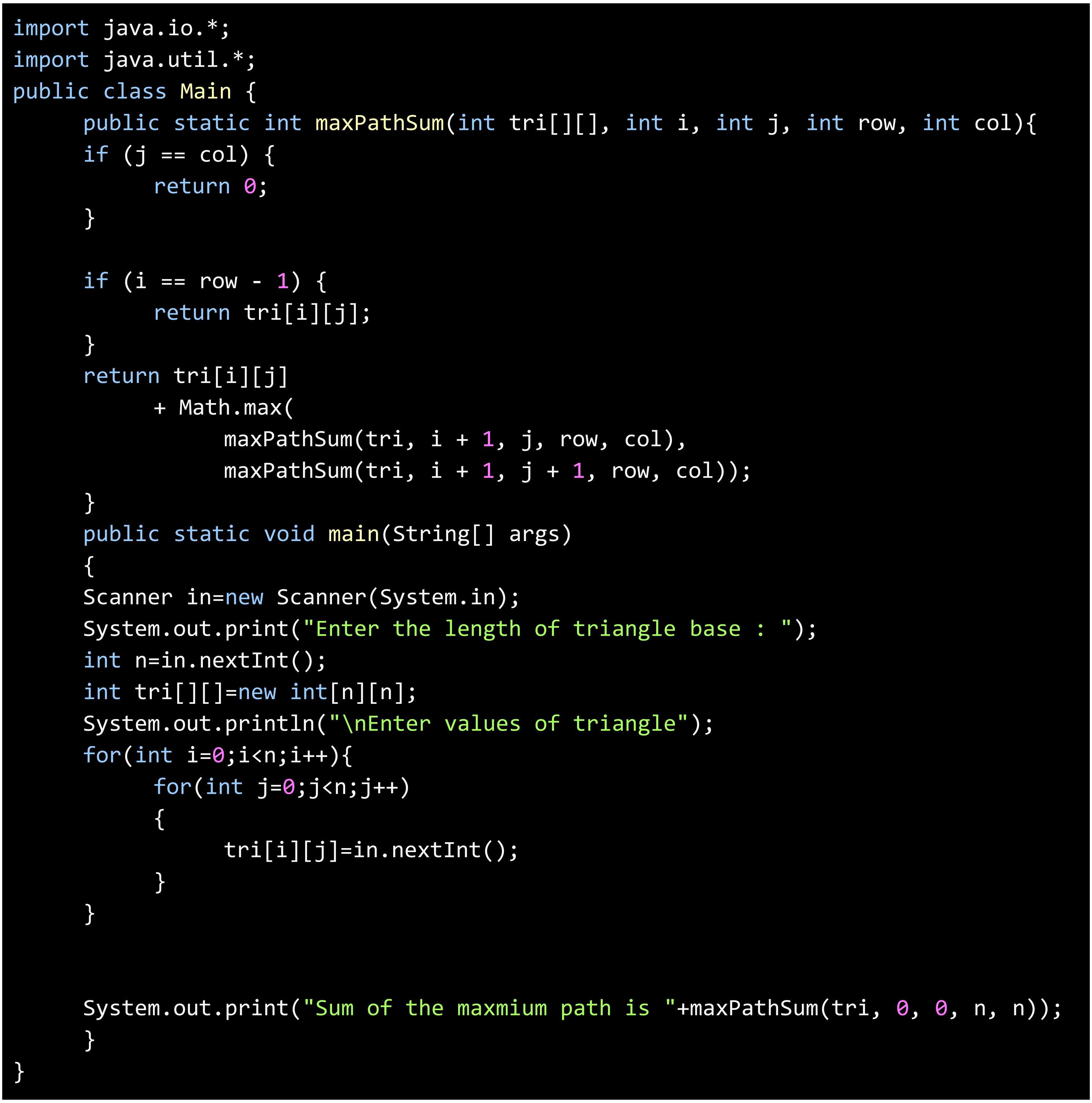
42 -15

-4 13 45

21 14 -92 33

Values in the pyramid can be both positive or negative. A path from the top of the pyramid to the bottom consists of starting at the top of the pyramid and taking steps diagonally left or diagonally right down to the bottom of the pyramid. The cost of a path is the sum of all the values in the pyramid. Find the path from the top of the pyramid to the bottom with the highest total cost.

CODE:

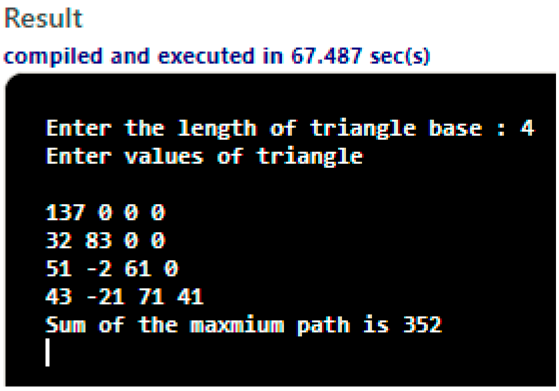


```java
import java.io.*;
import java.util.*;
public class Main {
    public static int maxPathSum(int tri[][], int i, int j, int row, int col){
    if (j == col) {
        return 0;
    }

    if (i == row - 1) {
        return tri[i][j];
    }
    return tri[i][j]
        + Math.max(
            maxPathSum(tri, i + 1, j, row, col),
            maxPathSum(tri, i + 1, j + 1, row, col));
    }
    public static void main(String[] args)
    {
    Scanner in=new Scanner(System.in);
    System.out.print("Enter the length of triangle base : ");
    int n=in.nextInt();
    int tri[][]=new int[n][n];
    System.out.println("\nEnter values of triangle");
    for(int i=0;i<n;i++){
        for(int j=0;j<n;j++)
        {
            tri[i][j]=in.nextInt();
        }
    }


    System.out.print("Sum of the maxmium path is "+maxPathSum(tri, 0, 0, n, n));
    }
}
```

OUTPUT:



Result

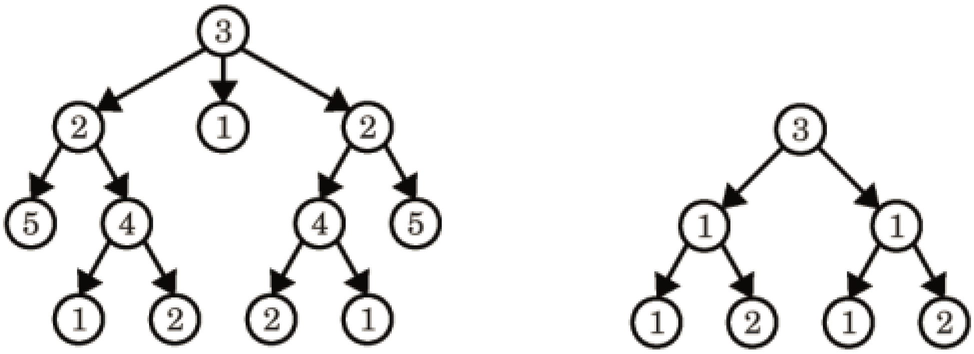
compiled and executed in 67.487 sec(s)

```
Enter the length of triangle base : 4
Enter values of triangle

137 0 0 0
32 83 0 0
51 -2 61 0
43 -21 71 41
Sum of the maxmium path is 352
```

CHECK IF TREE IS PALINDROMIC OR NOT

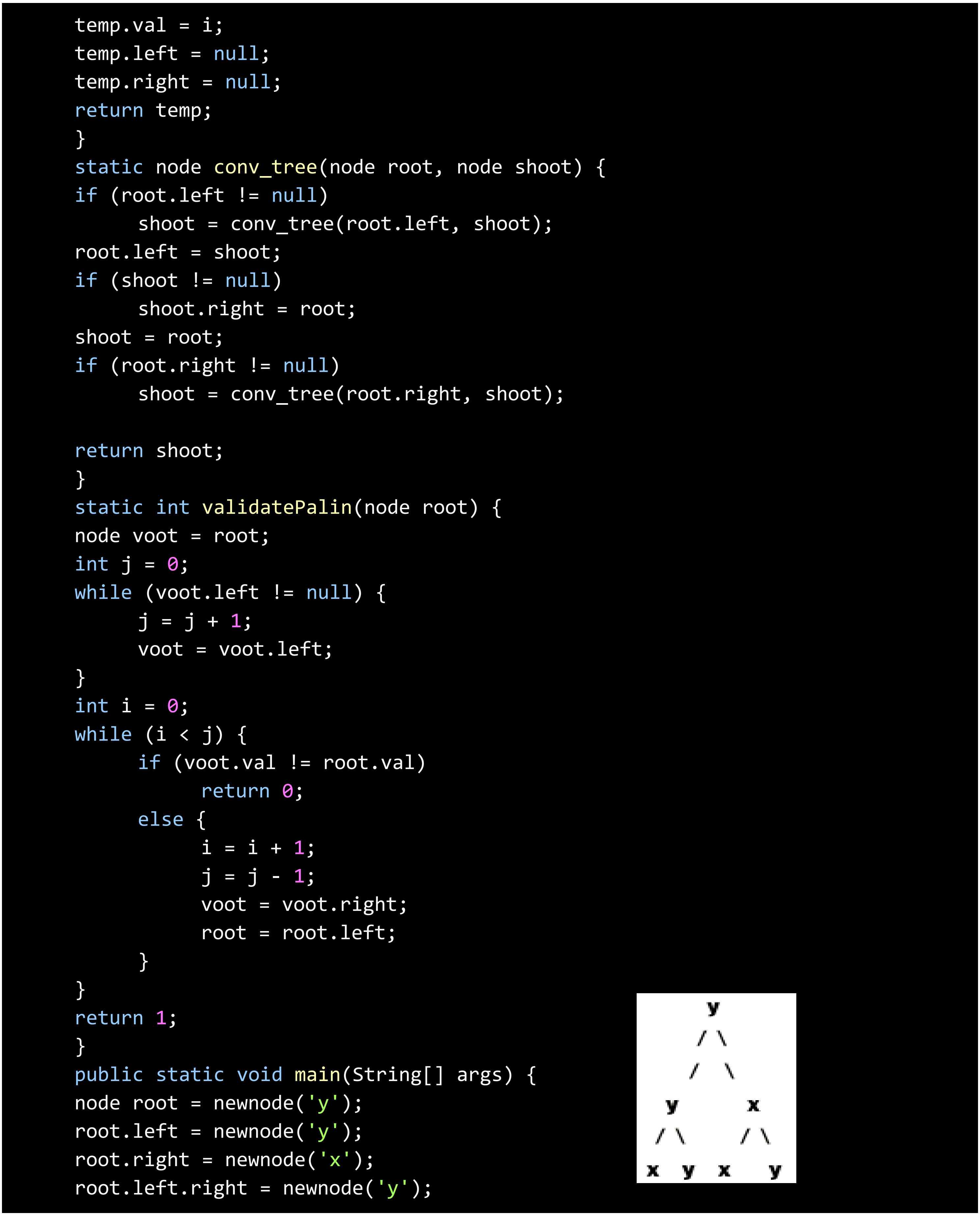
Q4.) A Palindromic tree is a tree that is the same when it's mirrored around the root. For example, the left tree below is a palindromic tree and the right tree below is not:



Given a tree, determine whether it is a palindromic tree.

CODE:

```
import java.util.*;
public class Main {
      static class node {
      char val;
      node left;
      node right;
      };
      static node newnode(char i) {
      node temp = null;
      temp = new node();
```



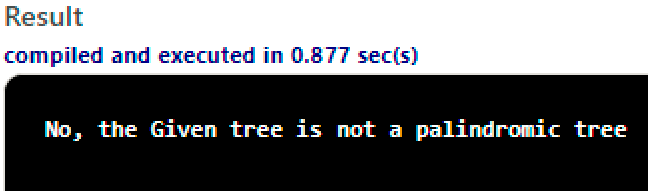
```
temp.val = i;
temp.left = null;
temp.right = null;
return temp;
}
static node conv_tree(node root, node shoot) {
if (root.left != null)
      shoot = conv_tree(root.left, shoot);
root.left = shoot;
if (shoot != null)
      shoot.right = root;
shoot = root;
if (root.right != null)
      shoot = conv_tree(root.right, shoot);

return shoot;
}
static int validatePalin(node root) {
node voot = root;
int j = 0;
while (voot.left != null) {
      j = j + 1;
      voot = voot.left;
}
int i = 0;
while (i < j) {
      if (voot.val != root.val)
            return 0;
      else {
            i = i + 1;
            j = j - 1;
            voot = voot.right;
            root = root.left;
      }
}
return 1;
}
public static void main(String[] args) {
node root = newnode('y');
root.left = newnode('y');
root.right = newnode('x');
root.left.right = newnode('y');
```

```
    root.left.left = newnode('x');
    root.right.right = newnode('y');
    root.right.left = newnode('x');
    node shoot = conv_tree(root, null);

    if (validatePalin(shoot) == 1)
        System.out.print("Yes, the Given tree is a palindromic tree");
    else
        System.out.print("No, the Given tree is not a palindromic tree");
    }
}
```

OUTPUT:



Result

compiled and executed in 0.877 sec(s)

```
No, the Given tree is not a palindromic tree
```

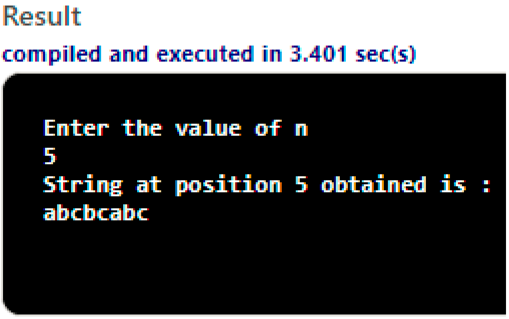
Nth ELEMENT IN FIBONACCI STRINGS

Q5.) The Fibonacci strings are a series of recursively-defined strings. Fa is the string a, Fi is the string be, and Fn+2 is the concatenation of Fn and Fn+1. For example, F2 is abc, F3 is bcabc, F4 is abcbcabc, etc. Given a number n and an index k , return the kth character of the string Fn

CODE:

```
import java.util.*;
public class Main
{
    static String getString(int N)
    {
    String sum = "";
    String A= "a",B= "bc";
    for (int i = 2; i < N; i++)
    {
         sum = A + B;
         A = B;
         B = sum;
    }
    return sum;
    }
    public static void main(String[] args)
    {
    Scanner in=new Scanner(System.in);
    System.out.println("Enter the value of n");
    int N = in.nextInt();
    System.out.println("String at position "+N+" obtained is :\n"+getString(N));
    }
}
```

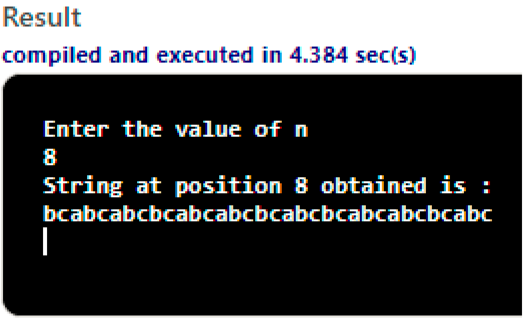
OUTPUT:



Result

compiled and executed in 3.401 sec(s)

```
Enter the value of n
5
String at position 5 obtained is :
abcbcabc
```



Result

compiled and executed in 4.384 sec(s)

```
Enter the value of n
8
String at position 8 obtained is :
bcabcabcbcabcabcbcabcbcabcabcbcabc
```