Programs Concepts

Programs

Basic Programs

Number Programs

Array Programs

Matrix Programs

Pattern Programs

String Programs

Tree Programs

Singly Linked List

Circular Linked List

Doubly Linked List

Programs List | Programming Examples

A list of programs or programming examples on C, C++, Java, C#, Python and PHP are given below.

- Basic Programs
- Number Programs
- Array Programs
- Matrix Programs
- Pattern Programs
- String Programs
- Tree Programs
- Singly Linked List Programs
- Circular Linked List Programs
- Doubly Linked List Programs
- Miscellaneous

Basic Programs

1) Program to calculate the area of rectangle

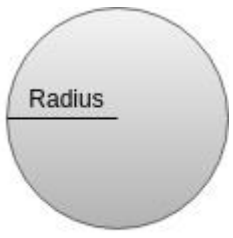
Input:

1. Width = 10, Height = 5

Output:

```
Area of Rectangle = Width * Height
                  = 10 * 5
                  = 50
```

2) Program to calculate the volume of sphere



Sphere

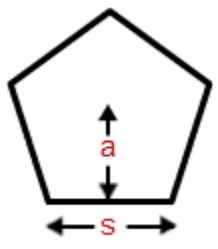
Input:

1. Radius = 48, Pie = 3.14

Output:

```
Volume = (4.0/3.0) * pie * (radius * radius * radius);
       = (4.0/3.0) * 3.14 * 48 * 48 * 48
       = 463433.132812
```

3) Program to find the area of the pentagon



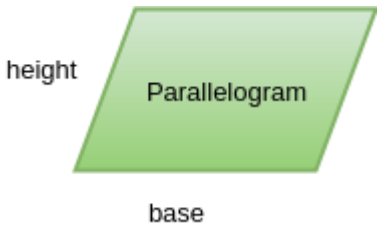
Input:

1. s = 13
2. a = 5;

Output:

```
Area of Pentagon = (5.0/2.0) * s * a
                 = (5.0/2.0) * 13 * 5
                 = 162.5
```

4) Program to find the area of parallelogram



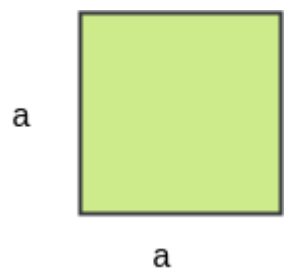
Input:

1. base = 4;
2. height = 18;

Output:

```
Area of Parallelogram = base * height;
                      = 4 * 18
                      = 72
```

5) Program to find the area of square



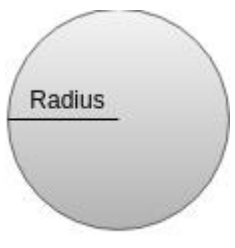
Input:

1. a = 13

Output:

```
Area of Square = a²
               = 13²
               = 169
```

6) Program to find the surface area of sphere



Sphere

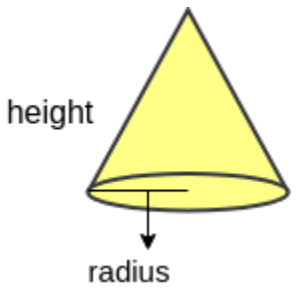
Input:

1. Radius = 37, Pie = 3.14

Output:

```
Volume = 4 * pie * (radius * radius);
       = 4 * 3.14 * 37 * 37
       = 17210.285714
```

7) Program to find the volume of cone



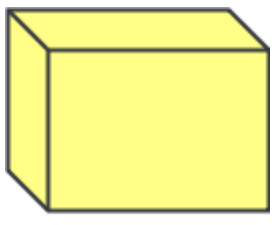
Input:

1. Radius = 38, Height = 35, Pie = 3.14

Output:

```
Volume = pie * radius * radius * height/3;
       = 3.14 * 38 * 38 * 35/3
       = 48766.666667
```

8) Program to find the volume of the cube



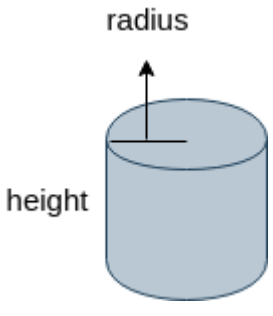
Input:

1. side = 4

Output:

```
Volume of cube = side³
               = 4³
               = 64
```

9) Program to find the volume of cylinder



Input:

1. radius (r) = 38 , height (h) = 35

Output:

```
Volume of the cylinder = pie * radius² * height
                       = 3.14 * 38* 38 * 35
                       = 146300.000000
```

10) Program to calculate the CGPA percentage

CGPA percentage is = (float)(9.5 * (CGPA));

Input:

1. CGPA = (Grades in all Subjects) / (Total Number of Subjects).
2. English = 9.1;
3. Hindi = 8.5;
4. Maths = 9.5;
5. Science =9.6;
6. SocialStudy = 8.6;
7. CGPA = (9.1+8.5+9.5+9.6+8.6)/(5.0);

Output:

```
CGPA percentage is = 86.070000
```

11) Program to convert Celsius into Fahrenheit

Temperature in Fahrenheit = ((celsius * 9) / 5) + 32

Input:

1. celsius= 12

Output:

```
Temperature in Fahrenheit = 53.6
```

12) Program to convert days into years

Input:

1. days= 799;

Output:

```
Number of years = days / 365;
                = 799 / 365
                = 2
```

13) Program to convert Fahrenheit into Celsius

Temperature in Celsius = ((Fahrenheit-32)*5)/9

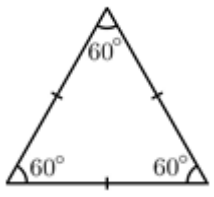
Input:

1. Fahrenheit = 54

Output:

```
Temperature in Celsius= ((54-32)*5)/9 = 12.22222
```

14) Program to find the area of an equilateral triangle



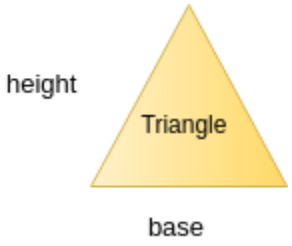
Input:

1. side (a) = 5

Output:

```
Area of Equilateral Triangle = ( 1.73 * a * a) / 4
                             = ( 1.73 * 5 * 5) / 4
                             = 10.81250
```

15) Program to find the area of a triangle



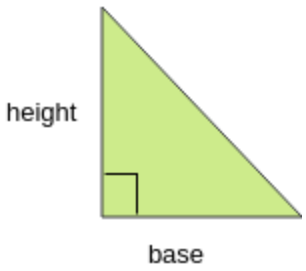
Input:

1. b = 5
2. h = 12

Output:

```
Area of Triangle = (b * h) / 2
                 = (5 * 12) / 2
                 = 30.0
```

16) Program to find the area of the right angle triangle



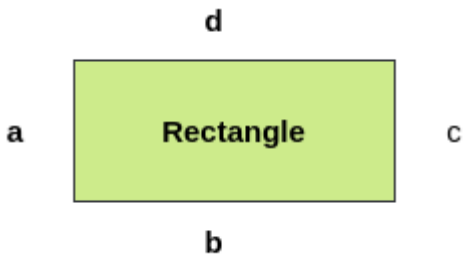
Input:

1. b = 5
2. h = 8

Output:

```
Area of Triangle = (b * h) / 2
                 = (5 * 8) / 2
                 = 20.0
```

17) Program to find the perimeter of the rectangle



Input:

1. a = c = 5
2. b = d = 4

Output:

```
Perimeter of Rectangle = 2 * ( a + b);
                       = 2 * (5 + 4)
                       = 18.00000
```

18) Program to find the simple interest

Simple Interest = (P × R × T) / 100

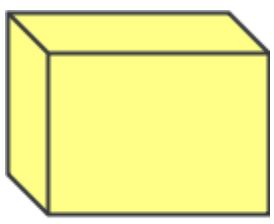
Input:

1. P = 34000, R = 30,T = 5
2. where P = Principal Amount, R = Rate per Annum, T = Time (years)

Output:

```
Simple Interest = 51000.000
```

19) Program to find the surface area of a cube



Surface Area Of Cube = 6 (a * a)

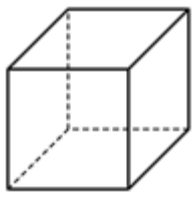
Input:

1. b = 5, h = 5
2. a (side) = length = breadth = height

Output:

```
Surface Area Of Cube = 6 * 5 * 5=150.00000
```

20) Program to find the surface area of cuboid



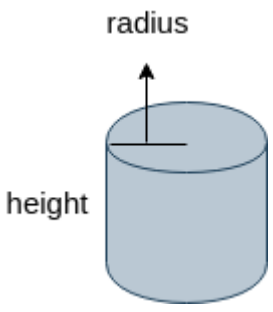
Input:

1. l= 2, w = 3, h = 5;
2. where l = length, w = width and h = height.

Output:

```
Surface Area OfCuboid = 2 * (l * w+ w * h + h * l)
                      = 2 * (2 * 3 + 3 * 5 + 5 * 2)
                      = 62.00000
```

21) Program to find the surface area of the cylinder



Surface Area of Cylinder = 2 Π (r + h)

Input:

1. r = 2.0, h = 5.0

Output:

```
Surface Area of Cylinder = 2 Π (r + h)
Here, r = radius, h = height, and Π ( pie ) = 3.14
        = 2 * 3.14 * ( 2.0 + 5.0)
        = 44.00000
```

Number Programs

1) Program to Check Disarium number

Input:

1. num = 175

Output:

```
1¹ + 7² + 5³ = 1 + 49 + 125 = 175
175 is a disarium number
```

2) Program to Check Happy number

Input:

1. num = 82

Output:

```
8² + 2² = 68
6² + 8² = 100
1² + 0² + 0² = 1
82 is a happy number number
```

3) Program to Check Harshad number

A number is said to be the Harshad number if it is divisible by the sum of its digit.

Input:

1. num = 156

Output:

```
156 is a Harshad number
```

4) Program to print all Disarium numbers between 1 to 100

Input:

1. range(1, 101)

Output:

```
Disarium numbers between 1 and 100 are: 1 2 3 4 5 6 7 8 9 89
```

5) Program to print all Happy numbers between 1 to 100

Input:

1. range(1, 101)

Output:

```
List of happy numbers between 1 and 100: 1 7 10 13 19 23 28 31 32 44 49 68 70 79 82 86 91 94 97 100
```

6) Program to print all Pronic numbers between 1 to 100

A number is said to be pronic number if it is a product of two consecutive numbers.

For examples:

6 = 2 x 3
72 = 8 x 9

Input:

1. range(1, 101)

Output:

```
Pronic numbers between 1 and 100: 2 6 12 20 30 42 56 72 90
```

7) Program to determine whether a given number is a Deficient number

8) Program to determine whether a given number is an abundant number

9) Program to determine whether a given number is a twisted prime number

10) Program to print all abundant numbers between 1 to 100

11) Program to print all Kaprekar numbers between 1 to 100

12) Program to print all prime numbers between 1 to 100

13) Program to print the average of n numbers

14) Program to print the combination (nCr) of the given number

15) Program to print the first 10 prime numbers

16) Program to print the permutation (nPr) of the given number

17) Program to print the sum of digits without using modulus

18) Program to swap two numbers

19) Program to swap two numbers without using the third variable

---

Array Programs

1) Program to copy all the elements of one array into another array

Input:

```
1. arr1 = [1, 2, 3, 4, 5];
2. arr2 = [None] * len(arr1);
```

Output:

```
Elements of original array: 1 2 3 4 5
Elements of new array: 1 2 3 4 5
```

2) Program to find the frequency of each element of an array

Input:

```
1. arr = [1, 2, 8, 3, 2, 2, 2, 5, 1]
```

Output:

```
Element | Frequency
       1|2
       2|4
       8|1
       3|1
       5|1
```

3) Program to left rotate the elements of an array

Input:

```
1. arr = [1, 2, 3, 4, 5]
2. Here, n determine the number of times an array should be rotated
3. n = 3
```

Output:

```
Original array: 1 2 3 4 5
Array after left rotation: 4 5 1 2 3
```

4) Program to print the duplicate elements of an array

Input:

```
1. arr = [1, 2, 3, 4, 2, 7, 8, 8, 3];
```

Output:

```
Duplicate elements in given array:
2
3
```

```
8
```

5) Program to print the elements of an array

Input:

1. arr = [1, 2, 3, 4, 5]

Output:

```
Elements of given array: 1 2 3 4 5
```

6) Program to print the elements of an array in reverse order

Input:

1. arr = [1, 2, 3, 4, 5]

Output:

```
Original array: 1 2 3 4 5
Array in reverse order: 5 4 3 2 1
```

7) Program to print the elements of an array present on even position

Input:

1. arr = [1, 2, 3, 4, 5]

Output:

```
Elements of given array present on even position:
2
4
```

8) Program to print the elements of an array present on odd position

Input:

1. arr = [1, 2, 3, 4, 5]

Output:

```
Elements of given array present on odd position:
1
3
5
```

9) Program to print the largest element present in an array

Input:

1. arr = [25, 11, 7, 75, 56]

Output:

```
Largest element present in given array: 75
```

10) Program to print the number of elements present in an array

Input:

1. arr = [1, 2, 3, 4, 5]

Output:

```
Number of elements present in given array: 5
```

11) Program to print the smallest element present in an array

Input:

1. arr = [25, 11, 7, 75, 56]

Output:

```
Smallest element present in given array: 7
```

12) Program to print the sum of all the elements of an array

Input:

1. arr = [1, 2, 3, 4, 5]
2. sum = 0

Output:

```
Sum of all the elements of an array: 15
```

13) Program to right rotate the elements of an array

Input:

1. arr = [1, 2, 3, 4, 5]
2. Here, n determine the number of times an array should be rotated
3. n = 3

Output:

```
Original array: 1 2 3 4 5
Array after right rotation: 3 4 5 1 2
```

14) Program to sort the elements of an array in ascending order

Input:

1. arr = [5, 2, 8, 7, 1]

Output:

```
Elements of original array: 5 2 8 7 1
Elements of array sorted in ascending order: 1 2 5 7 8
```

15) Program to sort the elements of an array in descending order

Input:

1. arr = [5, 2, 8, 7, 1]

Output:

```
Elements of original array: 5 2 8 7 1
Elements of array sorted in descending order: 8 7 5 2 1
```

Matrix Programs

1) Program to calculate the addition of 2 matrices

Input:

1. Matrix a = [1, 0, 1]
2. [4, 5, 6]
3. [1, 2, 3]
4.
5. matrix b = [1, 1, 1]
6. [2, 3, 1]
7. [1, 5, 1]

Output:

```
Addition of two matrices: [2 1 2]
                          [6 8 7]
```

```
[2 7 4]
```

2) Program to calculate the subtraction of 2 matrices

Input:

1. Matrix a = [4, 5, 6]
2. [3, 4, 1]
3. [1, 2, 3]
4.
5. Matrix b = [2, 0, 3]
6. [2, 3, 1]
7. [1, 1, 1]

Output:

```
Subtraction of two matrices: [2 5 3]
                             [1 1 0]
                             [0 1 2]
```

3) Program to determine whether a given matrix is an identity matrix

Input:

1. Matrix a =[1, 0, 0]
2. [0, 1, 0]
3. [0, 0, 1]

Output:

```
Given matrix is an identity matrix
```

4) Program to determine whether a given matrix is a sparse matrix

Input:

1. Matrix a = [4, 0, 0]
2. [0, 5, 0]
3. [0, 0, 6]

Output:

```
Given matrix is a sparse matrix
```

5) Program to determine whether two matrices are equal

Input:

1. Matrix a = [1, 2, 3]
2. [8, 4, 6]
3. [4, 5, 7]
4.
5. matrix b = [1, 2, 3]
6. [8, 4, 6]
7. [4, 5, 7]

Output:

```
Matrices are equal
```

6) Program to display the lower triangular matrix

Input:

1. Matrix a = [1, 2, 3]
2. [8, 6, 4]
3. [4, 5, 6]

Output:

```
Lower triangular matrix: [1 0 0]
                         [8 6 0]
                         [4 5 6]
```

7) Program to display the upper triangular matrix

Input:

1. Matrix a = [1, 2, 3]
2. [8, 6, 4]
3. [4, 5, 6]

Output:

```
Upper triangular matrix: [1 2 3]
                         [0 6 4]
                         [0 0 6]
```

8) Program to find the frequency of odd & even numbers in the given Matrix

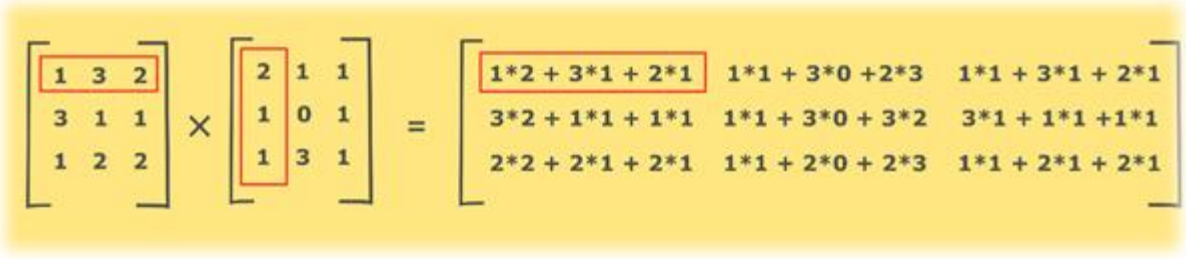
Input:

1. Matrix a = [4, 1, 3]
2. [3, 5, 7]
3. [8, 2, 6]

Output:

```
Frequency of odd numbers: 5
Frequency of even numbers: 4
```

9) Program to find the product of two matrices



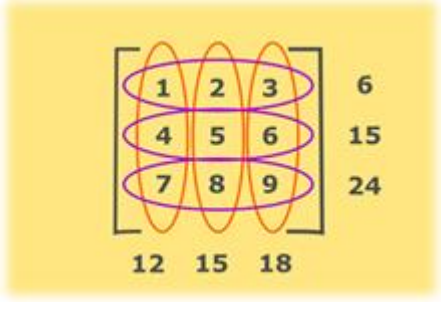
Input:

1. Matrix a = [1, 3, 2]
2. [3, 1, 1]
3. [1, 2, 2]
4.
5. matrix b = [2, 1, 1]
6. [1, 0, 1]
7. [1, 3, 1]

Output:

```
Product of two matrices: [7 7 6]
                         [8 6 5]
                         [6 7 5]
```

10) Program to find the sum of each row and each column of a matrix



Input:

1. Matrix a = [1, 2, 3]
2. [4, 5, 6]
3. [7, 8, 9]

Output:

```
Sum of 1 row: 6
Sum of 2 row: 15
Sum of 3 row: 24
Sum of 1 column: 12
Sum of 2 column: 15
Sum of 3 column: 18
```

11) Program to find the transpose of a given matrix

Input:

1. Matrix a = [1, 2, 3]
2. [4, 5, 6]
3. [7, 8, 9]

Output:

```
Transpose of given matrix: [1 4 7]
                           [2 5 8]
                           [3 6 9]
```

String Programs

1) Program to count the total number of punctuation characters exists in a string

Input:

1. char str [] = "Good Morning! Mr. James Potter. Had your breakfast?"

Output:

If any character in the string is matched with ('!', "," ,"\'" ,";" ,"\"", ".", "-" ,"?"), increment the count by 1.

```
Total number of punctuation characters exists in string: 4
```

2) Program to count the total number of vowels and consonants in a string

Input:

1. str = "This is a really simple sentence"

Output:

```
vowels = a, e, i, o, u
consonants = b, c, d, f, g, h, j, k, l, m, n, p, q, r, s, t, v, w, x, y, z
```

3) Program to determine whether two strings are the anagram

Input:

Two Strings are called the anagram if they contain the same characters. However, the order or sequence of the characters can be different.

```
str1 = "Grab";
str2 = "Brag";
```

Output:

```
Both the strings are anagram.
```

4) Program to divide a string in 'N' equal parts

Input:

1. str = "aaaabbbbcccc"

Output:

```
Equal parts of given string are
aaaa
bbbb
cccc
```

5) Program to find all the permutations of a string

Input:

1. char str[] = "ABC"

Output:

```
All the permutations of the string are:
ABC
ACB
BAC
BCA
CBA
CAB
```

6) Program to find all possible subsets of a string

Input:

1. str = "ABC"

Output:

```
All subsets for given string are:
A
AB
ABC
B
BC
C
```

7) Program to find the longest repeating sequence in a string

Input:

1. str = "acbdfghybdf"

Output:

```
Longest repeating sequence: bdf
```

8) Program to remove all the white spaces from a string

Input:

```
1. str1 = "Remove white spaces"
```

Output:

```
String after removing all the white spaces : Removewhitespaces
```

9) Program to replace lower-case characters with upper-case and vice versa

Input:

```
1. str1 = "Great Power"
```

Output:

```
String after case conversion : gREAT pOWER
```

10) Program to replace the spaces of a string with a specific character

Input:

```
1. char string[] = "Once in a blue moon"
2. char ch = '-'
```

Output:

```
String after replacing spaces with given character: Once-in-a-blue-moon
```

11) Program to Count the Total Number of Characters in a String

Input:

```
1. string = "The best of both worlds"
```

Output:

```
Total number of characters in a string: 19
```

12) Program to Count the Total Number of Words in a String

Input:

1. sentence = "Beauty lies in the eyes of beholder"

Output:

```
Total number of words in the given string: 7
```

13) Program to Determine Whether a Given String is Palindrome

Input:

1. string = "Kayak"

Output:

```
Given string is palindrome.
```

14) Program to Determine Whether One String is a Rotation of Another

Input:

1. str1 = "abcde"
2. str2 = "deabc"

Output:

```
Second string is a rotation of first string.
```

15) Program to Find Maximum and Minimum Occurring Character in a String

Input:

1. string = "grass is greener on the other side"

Output:

```
Minimum occurring character: a
Maximum occurring character: e
```

16) Program to Find Reverse of a String

Input:

1. string = "Dream big"

Output:

```
Original string: Dream big
Reverse of given string: gib maerD
```

17) Program to Find the Duplicate Characters in a String

Input:

1. string = "Great responsibility"

Output:

```
Duplicate characters in a given string:
r
e
t
s
i
```

18) Program to Find the Duplicate Words in a String

Input:

1. string = "big black bug bit a big black dog on his big black nose"

Output:

```
Duplicate words in a given string:
big
black
```

19) Program to Find the Frequency of Characters

Input:

1. string = "picture perfect"

Output:

```
Characters and their corresponding frequencies
p-2
i-1
c-2
t-2
u-1
r-2
e-3
f-1
```

20) Program to Find the Largest and Smallest Word in a String

Input:

```
1. string = "Hardships often prepare ordinary people for an extraordinary destiny"
```

Output:

```
Smallest word: an
Largest word: extraordinary
```

21) Program to Find the Most Repeated Word in a Text File

Input:

```
1. file = open("data.txt", "r")
```

data.txt file content:

The term "computer" is derived from Latin word "computare" which means to calculate. Computer is a programmable electronic device. Computer accepts raw data as input and processes it with set of instructions to produce result as output. The history of computer begins with the birth of abacus which is believed to be the first computer.

Output:

```
Most repeated word: computer
```

22) Program to Find the Number of Words in the Given Text File

Input:

```
1. file = open("data.txt", "r")
```

data.txt file content:

The term "computer" is derived from Latin word "computare" which means to calculate. Computer is a programmable electronic device. Computer accepts raw data as input and processes it with set of instructions to produce result as output. The history of computer begins with the birth of abacus which is believed to be the first computer.

Output:

```
Number of words present in given file: 117
```

23) Program to Print Smallest and Biggest Possible Palindrome Word in a Given String

Input:

```
1. string = "Wow you own kayak"
```

Output:

```
Smallest palindromic word: wow
Biggest palindromic word: kayak
```

24) Program to Separate the Individual Characters from a String

Input:

```
1. string = "characters"
```

Output:

```
Individual characters from given string: characters
```

25) Program to Swap two String Variables Without Using Third or Temp Variable

Input:

```
1. str1 = "Good";
2. str2 = "morning";
```

Output:

```
Strings before swapping: Good morning
Strings after swapping: morning Good
```

Pattern Programs

1) Program To Print Following Pattern

```
*             *
* *           * *
* * *       * * *
* * * *     * * * *
* * * * *   * * * * *
* * * * *   * * * * *
* * * *     * * * *
* * *       * * *
* *           * *
*             *
```

2) Program To Print Following Pattern

```
5432*
543*1
54*21
5*321
*4321
```

3) Program To Print Following Pattern

```
*000*000*
0*00*00*0
00*0*0*00
000***000
```

4) Program To Print Following Pattern

```
1
2 4
3 6 9
4 8 12 16
5 10 15 20 25
6 12 18 24 30 36
7 14 21 28 35 42 49
8 16 24 32 40 48 56 64
9 18 27 36 45 54 63 72 81
10 20 30 40 50 60 70 80 90 100
```

5) Program To Print Following Pattern

```
1     1
 2   2
  3 3
   4
  3 3
 2   2
1     1
```

6) Program To Print Following Pattern

```
1   2   3   4   5
16              6
15              7
14              8
13 12 11 10 9
```

7) Program To Print Following Pattern

```
12344321
123**321
12****21
1******1
```

8) Program To Print Following Pattern

```
1   2   3   4   5   6   7   8   9   10
36  37  38  39  40  41  42  43  44  11
35  64  65  66  67  68  69  70  45  12
34  63  84  85  86  87  88  71  46  13
33  62  83  96  97  98  89  72  47  14
32  61  82  95  100 99  90  73  48  15
31  60  81  94  93  92  91  74  49  16
30  59  80  79  78  77  76  75  50  17
29  58  57  56  55  54  53  52  51  18
28  27  26  25  24  23  22  21  20  19
```

9) Program To Print Following Pattern

```
0
909
89098
7890987
678909876
56789098765
4567890987654
345678909876543
```

23456789098765432
1234567890987654321

10) Program to Print Following Pattern

```
*
* *
* * *
* * * *
* * * * *
* * * * * *
```

11) Program to Print Following Pattern

```
A
B                                                                   B
C                                   C                               C
D                       D                       D                   D
E E E E E
```

12) Program to Print Following Pattern

```
1 1 1 1 1 1 1 1 1 1
1                   1
1                   1
1                   1
1                   1
1                   1
1                   1
1                   1
1                   1
1 1 1 1 1 1 1 1 1 1
```

13) Program to Print Following Pattern

```
1               2               3               4               5
1                       2               3                       4
1                               2                               3
1                                                               2
1
```

14) Program to Print Following Pattern

```
*********
 *******
  *****
   ***
    *
```

15) Program to Print Following Pattern

```
            1
          1   1
        1   2   1
      1   3   3   1
    1  4    6   4   1
  1  5   10   10  5   1
```

16) Program to Print Following Pattern

```
* * * * *
 * * * *
  * * *
   * *
    *
```

17) Program to Print Following Pattern

```
1
2                                                   3
4                           5                       6
7 8 9 10
```

18) Program to Print Following Pattern

```
1
1                                                   2
1                           2                       3
1                   2               3               4
1 2 3 4 5
```

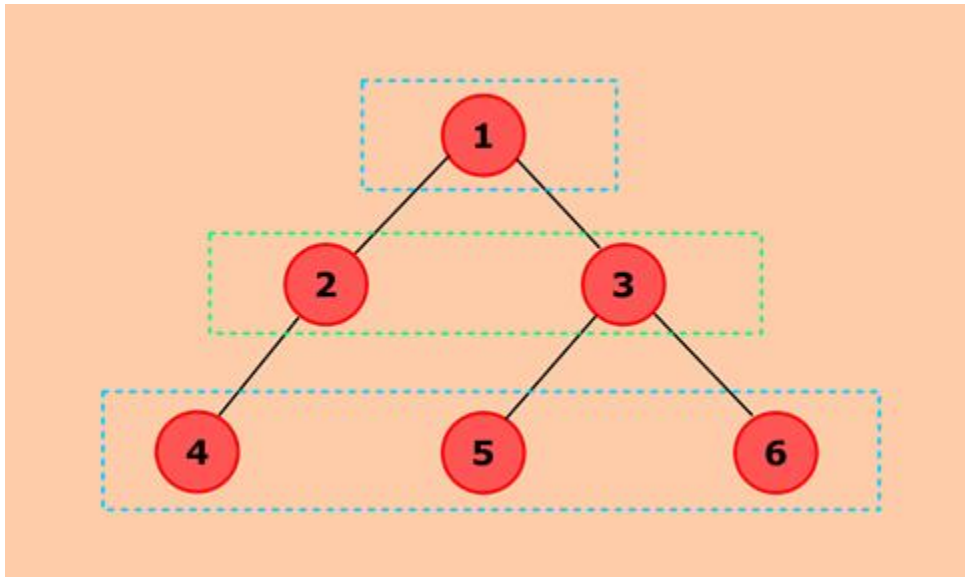
19) Program to Print Following Pattern

```
          *
        * * *
      * * * * *
    * * * * * * *
  * * * * * * * * *
```

Tree Programs

1) Program to Calculate the Difference Between the Sum of the Odd Level and Even Level Nodes of a Binary Tree

Input:

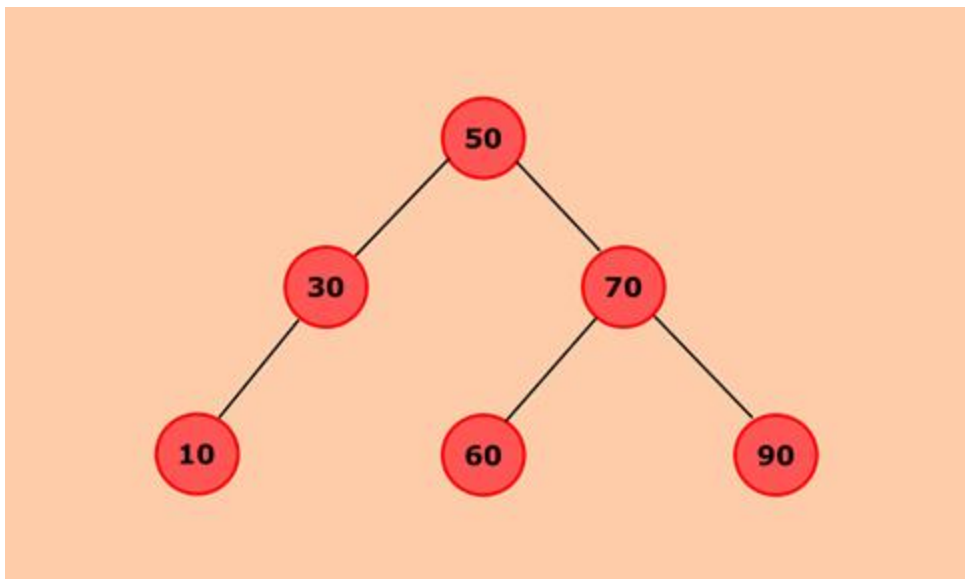


Output:

```
OddLevelSum = 1 + 4 + 5 + 6 = 16
EvenLevelSum = 2 + 3 = 5
Difference = |16 - 5| = 11
```

2) Program to Construct a Binary Search Tree and Perform Deletion and Inorder Traversal

Input:

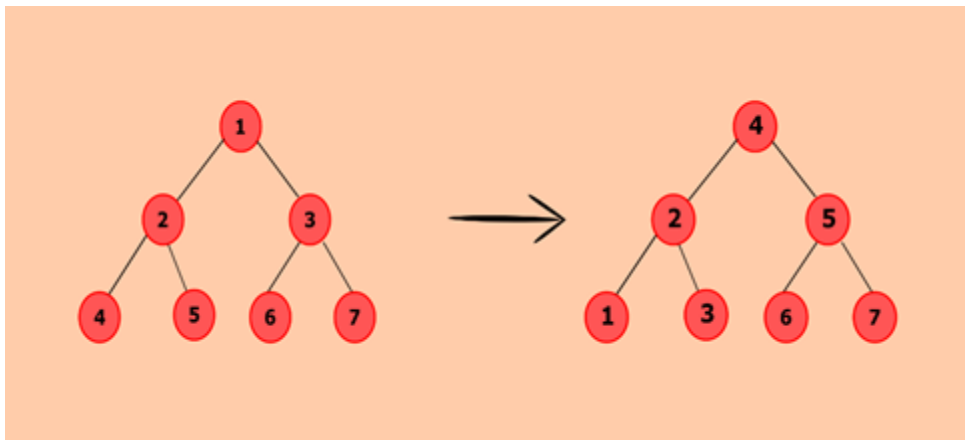


Output:

```
Binary search tree after insertion: 10 30 50 60 70 90
Binary search tree after deleting node 90: 10 30 50 60 70
Binary search tree after deleting node 30: 10 50 60 70
Binary search tree after deleting node 50: 10 60 70
```

3) Program to Convert Binary Tree to Binary Search Tree

Input:

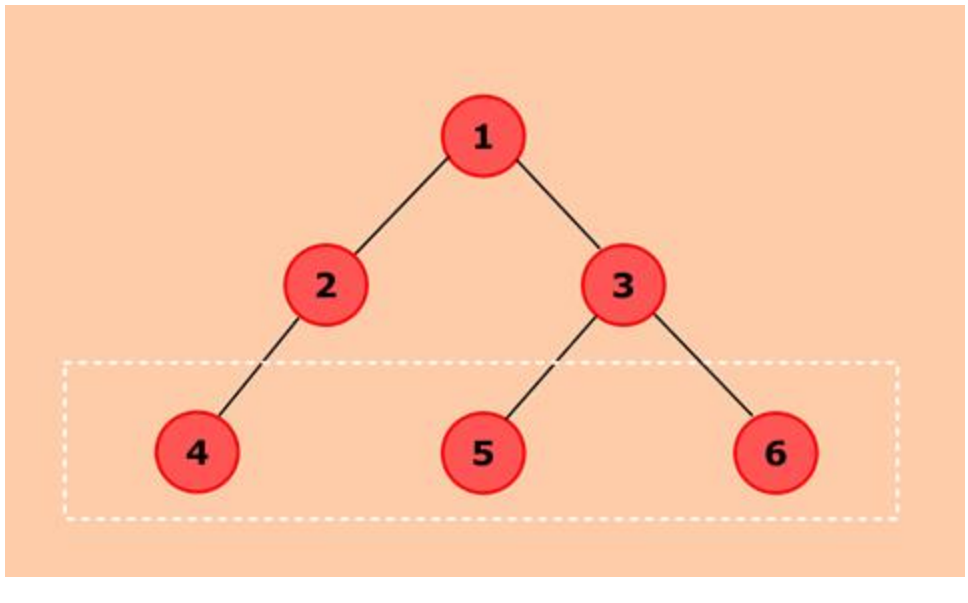


Output:

```
Inorder representation of binary tree: 4 2 5 1 6 3 7
Inorder representation of resulting binary search tree: 1 2 3 4 5 6 7
```

4) Program to Determine Whether all Leaves are at Same Level

Input:

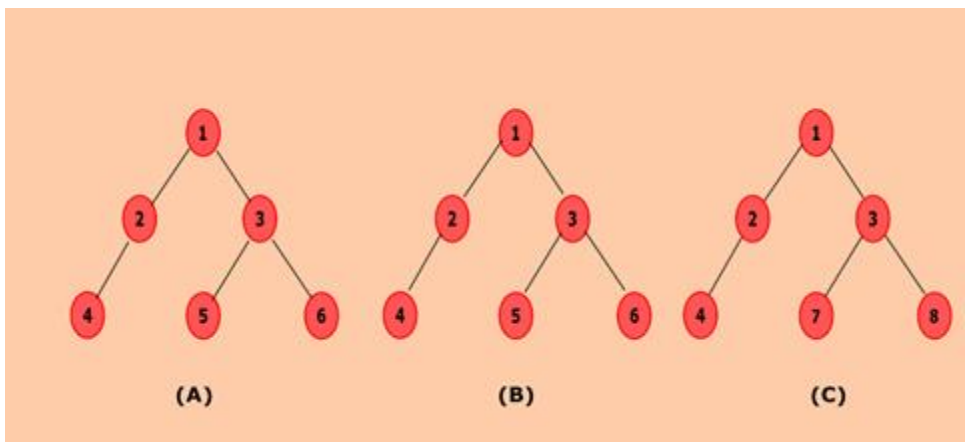


Output:

```
All leaves are at same level
```

5) Program to Determine Whether two Trees are Identical

Input:

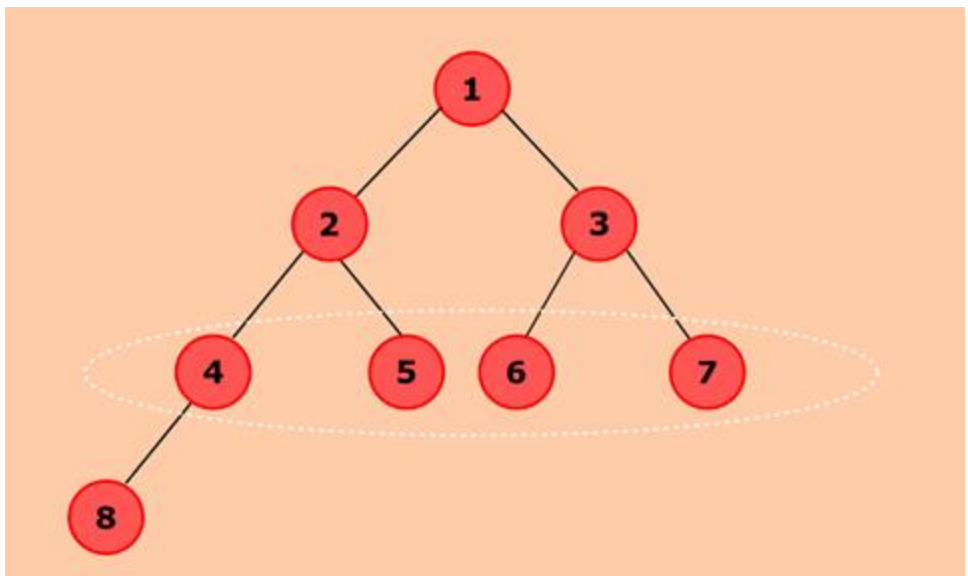


Output:

```
Both the binary trees are identical
```

6) Program to Find Maximum Width of a Binary Tree

Input:

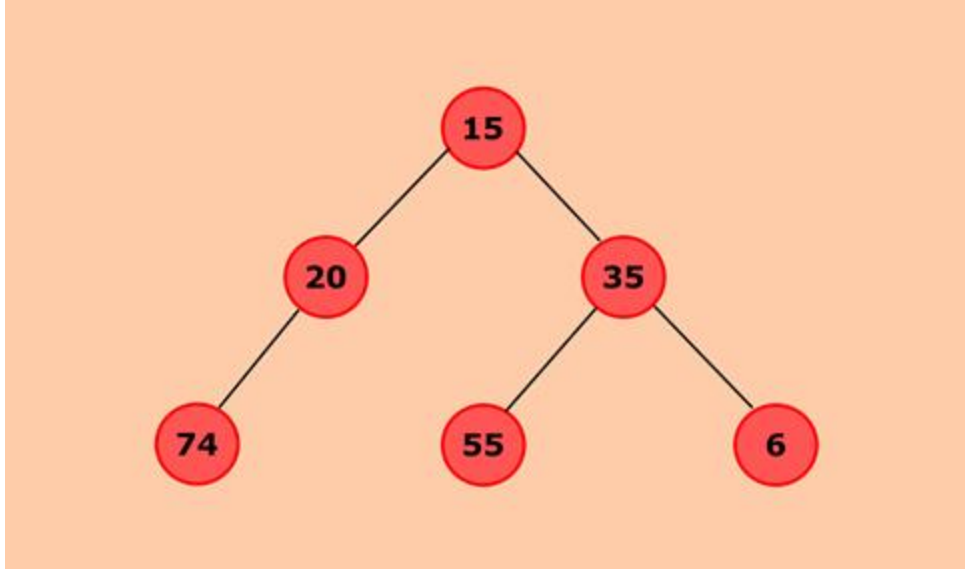


Output:

```
Maximum width of the binary tree: 4
```

7) Program to Find the Largest Element in a Binary Tree

Input:

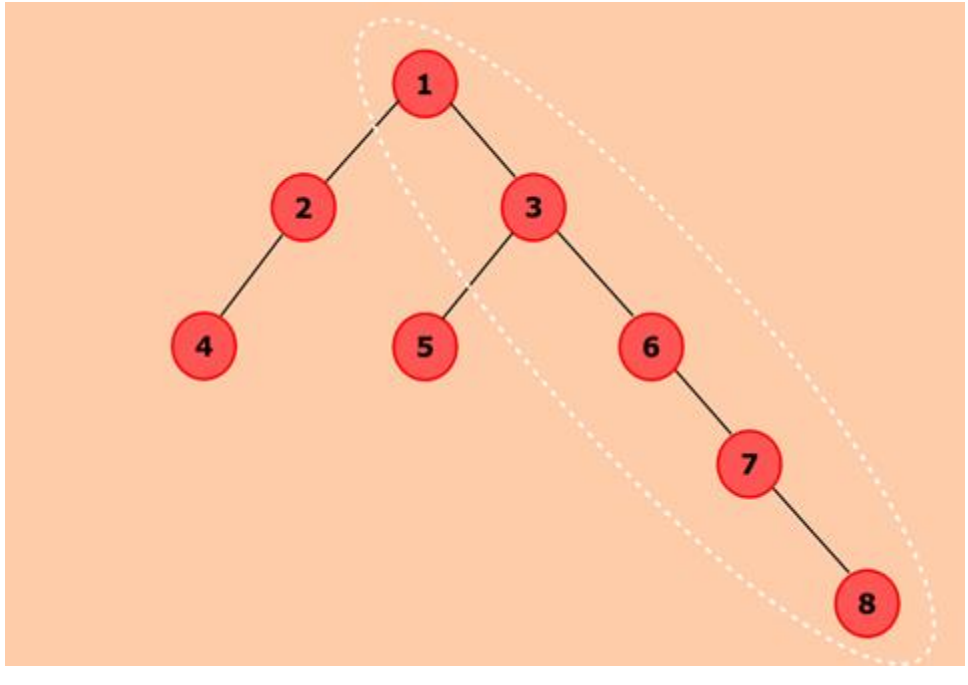


Output:

```
Largest element in the binary tree: 74
```

8) Program to Find the Maximum Depth or Height of a Tree

Input:

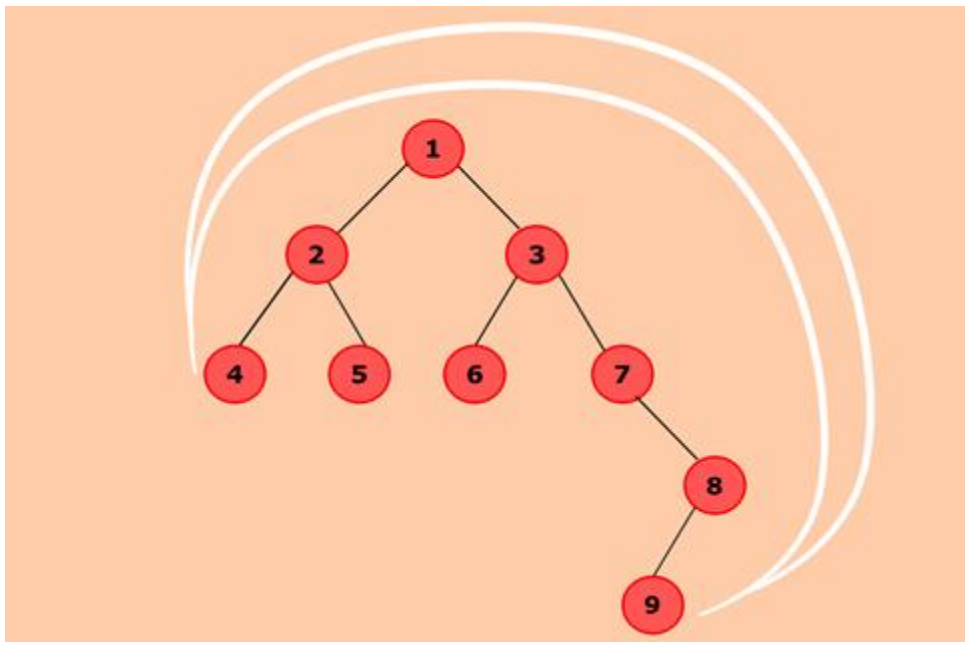


Output:

```
Maximum height of given binary tree: 5
```

9) Program to Find the Nodes Which are at the Maximum Distance in a Binary Tree

Input:

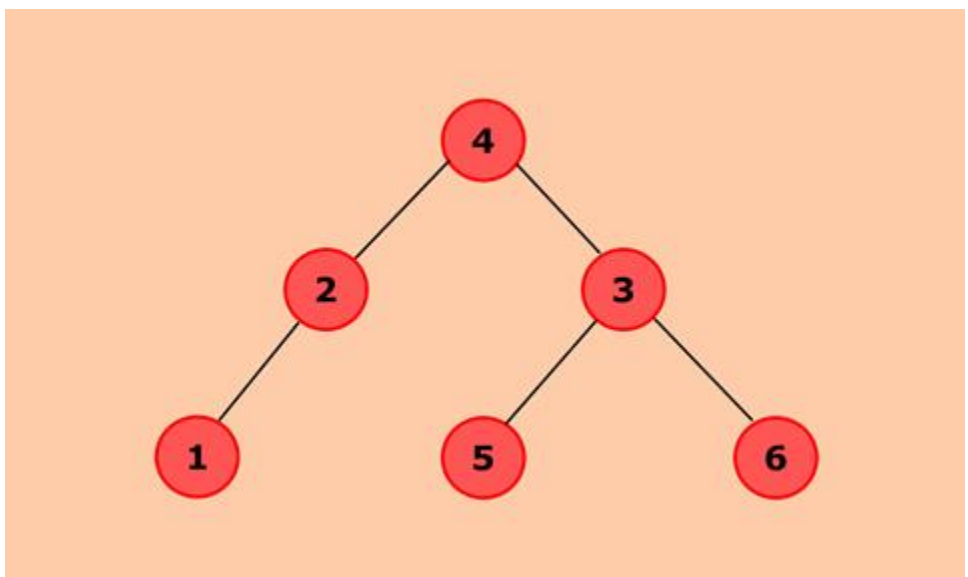


Output:

```
Nodes which are at maximum distance:
( 4,9 )
( 5,9 )
```

10) Program to Find the Smallest Element in a Binary Tree

Input:

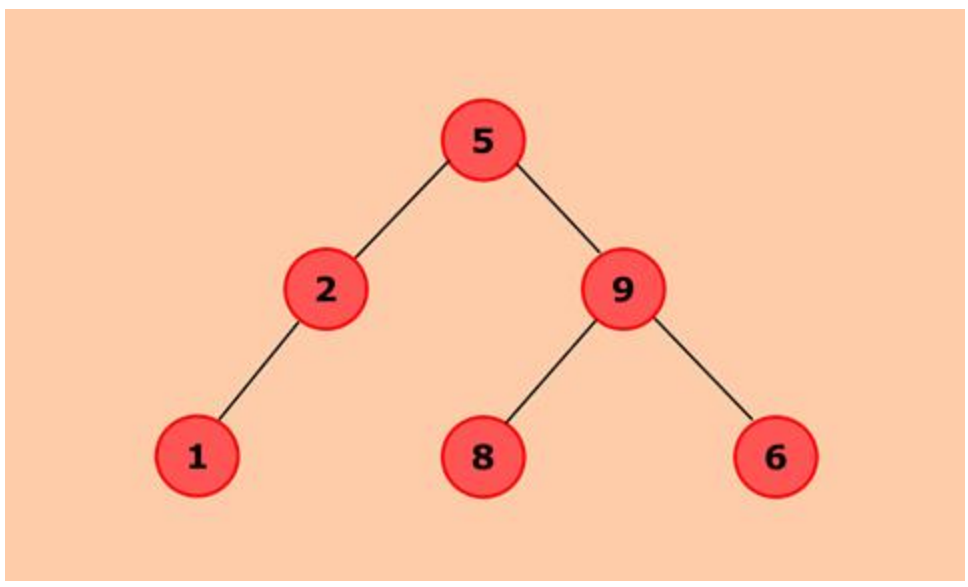


Output:

```
Smallest element in the binary tree: 1
```

11) Program to Find the Sum of all the Nodes of a Binary Tree

Input:

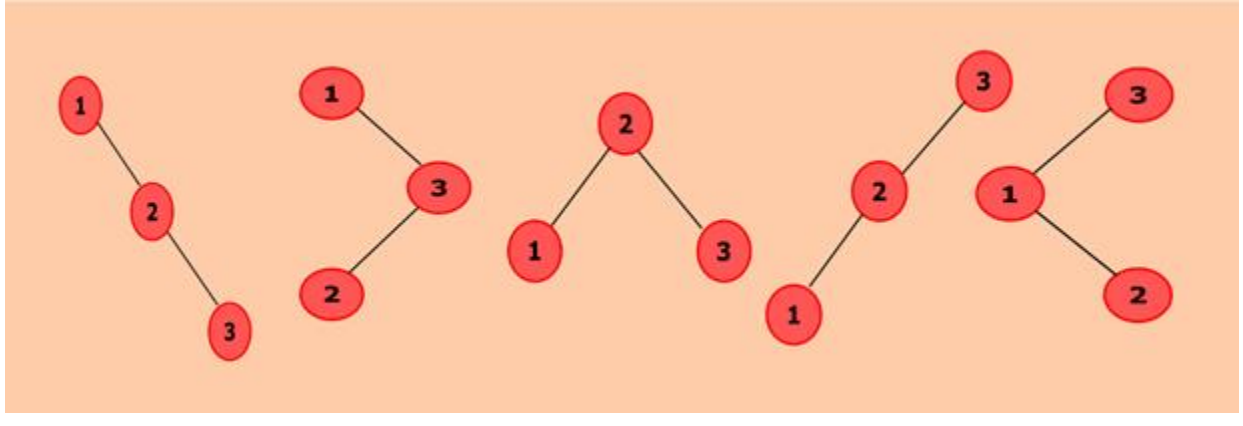


Output:

```
Sum of all nodes of binary tree: 31
```

12) Program to Find the Total Number of Possible Binary Search Trees with N Keys

Input:

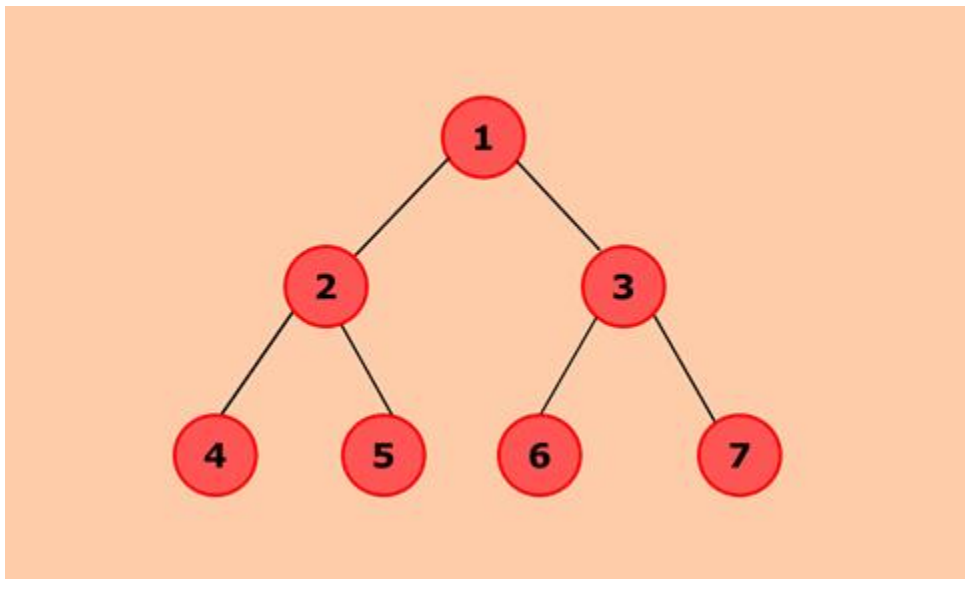


Output:

```
Total number of possible Binary Search Trees with given key: 42
```

13) Program to Implement Binary Tree using the Linked List

Input:



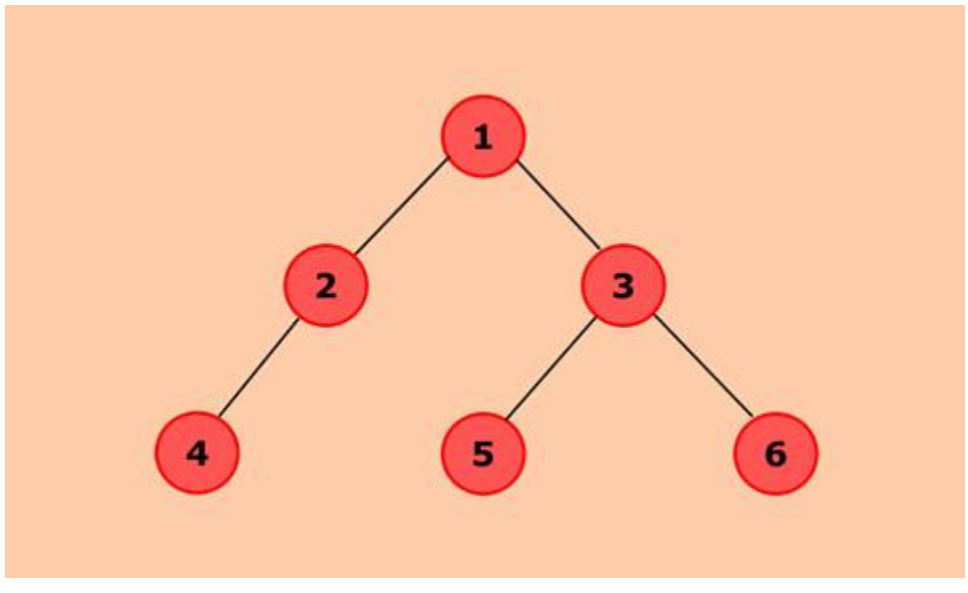
Output:

```
Binary tree after insertion: 1
Binary tree after insertion: 2 1 3
Binary tree after insertion: 4 2 5 1 3
Binary tree after insertion: 4 2 5 1 6 3 7
```

14) Program to Search a Node in a Binary Tree

Input:

Search for node 5 in the binary tree



Output:

```
Element is present in the binary tree
```

Singly Linked List Programs

1) Singly Linked List Examples

Input:

1. Head Node = 100
2. Second Node = 200
3. Third Node = 300

Output:

```
100
200
300
```

2) Program to create and display a singly linked list

Input:

1. #Add nodes to the list
2. sList.addNode(1);
3. sList.addNode(2);
4. sList.addNode(3);
5. sList.addNode(4);

Output:

```
Nodes of singly linked list: 1 2 3 4
```

3) Program to create a singly linked list of n nodes and count the number of nodes

Input:

1. #Add nodes to the list
2. sList.addNode(1);
3. sList.addNode(2);
4. sList.addNode(3);
5. sList.addNode(4);

Output:

```
Nodes of singly linked list: 1 2 3 4
Count of nodes present in the list: 4
```

4) Program to create a singly linked list of n nodes and display it in reverse order

Input:

1. #Add nodes to the list
2. sList.addNode(1);
3. sList.addNode(2);
4. sList.addNode(3);
5. sList.addNode(4);

Output:

```
Original List: 1 2 3 4
Reversed List: 4 3 2 1
```

5) Program to delete a new node from the beginning of the singly linked list

Input:

1. #Add nodes to the list
2. sList.addNode(1);
3. sList.addNode(2);
4. sList.addNode(3);
5. sList.addNode(4);

Output:

```
Original List: 1 2 3 4
Updated List: 2 3 4
Updated List: 3 4
Updated List: 4
Updated List: List is empty
```

6) Program to delete a new node from the middle of the singly linked list

Input:

1. #Add nodes to the list
2. sList.addNode(1);
3. sList.addNode(2);
4. sList.addNode(3);
5. sList.addNode(4);

Output:

```
Original List: 1 2 3 4
Updated List: 1 3 4
Updated List: 1 4
Updated List: 4
Updated List: List is empty
```

7) Program to delete a node from the end of the singly linked list

Input:

1. #Add nodes to the list
2. sList.addNode(1);
3. sList.addNode(2);
4. sList.addNode(3);
5. sList.addNode(4);

Output:

```
Original List: 1 2 3 4
Updated List: 1 2 3
Updated List: 1 2
Updated List: 1
Updated List: List is empty
```

8) Program to determine whether a singly linked list is the palindrome

Input:

1. #Add nodes to the list
2. sList.addNode(1);
3. sList.addNode(2);
4. sList.addNode(3);
5. sList.addNode(2);
6. sList.addNode(1);

Output:

```
Nodes of the singly linked list: 1 2 3 2 1
Given singly linked list is a palindrome
```

9) Program to find the maximum and minimum value node from a singly linked list

Input:

1. #Add nodes to the list
2. sList.addNode(5);
3. sList.addNode(8);
4. sList.addNode(1);
5. sList.addNode(6);

Output:

```
Minimum value node in the list: 1
Maximum value node in the list: 8
```

10) Program to insert a new node at the middle of the singly linked list

Input:

```
#Adds data to the list
sList.addNode(1);
sList.addNode(2);
#Inserting node '3' in the middle
sList.addInMid(3);
#Inserting node '4' in the middle
sList.addInMid(4);
```

Output:

```
Original list: 1 2
Updated List: 1 3 2
Updated List: 1 3 4 2
```

11) Program to insert a new node at the beginning of the singly linked list

Input:

```
#Adding 1 to the list
sList.addAtStart(1);
#Adding 2 to the list
sList.addAtStart(2);
#Adding 3 to the list
sList.addAtStart(3);
#Adding 4 to the list
sList.addAtStart(4);
```

Output:

```
Adding nodes to the start of the list: 1
Adding nodes to the start of the list: 2 1
Adding nodes to the start of the list: 3 2 1
Adding nodes to the start of the list: 4 3 2 1
```

12) Program to insert a new node at the end of the singly linked list

Input:

```
#Adding 1 to the list
sList.addAtEnd(1);
#Adding 2 to the list
sList.addAtEnd(2);
#Adding 3 to the list
sList.addAtEnd(3);
#Adding 4 to the list
sList.addAtEnd(4);
```

Output:

```
Adding nodes to the end of the list: 1
Adding nodes to the end of the list: 1 2
Adding nodes to the end of the list: 1 2 3
Adding nodes to the end of the list: 1 2 3 4
```

13) Program to remove duplicate elements from a singly linked list

Input:

```
#Adds data to the list
sList.addNode(1);
sList.addNode(2);
sList.addNode(3);
sList.addNode(2);
sList.addNode(2);
sList.addNode(4);
sList.addNode(1);
```

Output:

```
Originals list: 1 2 3 2 2 4 1
List after removing duplicates: 1 2 3 4
```

14) Program to search an element in a singly linked list

Input:

```
#Add nodes to the list
sList.addNode(1);
sList.addNode(2);
sList.addNode(3);
sList.addNode(4);
#Search for node 2 in the list
sList.searchNode(2);
#Search for the nodein the list
sList.searchNode(7);
```

Output:

```
Element is present in the list at the position : 2
Element is not present in the list
```

15) Program to sort the elements of the singly linked list

Input:

```
#Adds data to the list
sList.addNode(9);
sList.addNode(7);
sList.addNode(2);
sList.addNode(5);
sList.addNode(4);
```

Output:

```
Original list: 9 7 2 5 4
Sorted list: 2 4 5 7 9
```

16) Program to swap nodes in a singly linked list without swapping data

Input:

```
1. #Add nodes to the list
2. sList.addNode(1);
3. sList.addNode(2);
4. sList.addNode(3);
5. sList.addNode(4);
6. sList.addNode(5);
7. #Swaps the node 2 with node 5
8. sList.swap(2,5);
```

Output:

```
Original list: 1 2 3 4 5
List after swapping nodes: 1 5 3 4 2
```

17) Program to swap the last element of the singly linked list from the first one

Input:

```
1. #Add nodes to the list
2. sList.addNode(1);
3. sList.addNode(2);
4. sList.addNode(3);
5. sList.addNode(4);
```

Output:

```
Originals list: 1 2 3 4
List after swapping the first node with last: 4 2 3 1
```

Circular Linked List Programs

1) Program to Create a Circular Linked List of N Nodes and Count the Number of Nodes

Input:

```
1. #Adds data to the list
2. cl.add(1);
3. cl.add(2);
4. cl.add(4);
5. cl.add(1);
6. cl.add(2);
```

7. cl.add(3);

Output:

```
Count of nodes present in circular linked list: 6
```

2) Program to Create a Circular Linked List of N Nodes and Display it in Reverse Order

Input:

1. #Adds data to the list
2. cl.add(1);
3. cl.add(2);
4. cl.add(3);
5. cl.add(4);
6. cl.add(5);
7. cl.add(6);

Output:

```
Original List: 1 2 3 4 5 6
Reversed List: 6 5 4 3 2 1
```

3) Program to Create and Display a Circular Linked List

Input:

1. #Adds data to the list
2. cl.add(1);
3. cl.add(2);
4. cl.add(3);
5. cl.add(4);

Output:

```
Nodes of the circular linked list: 1 2 3 4
```

4) Program to Delete a New Node From the Beginning of the Circular Linked List

Input:

1. #Adds data to the list
2. cl.add(1);
3. cl.add(2);
4. cl.add(3);
5. cl.add(4);

Output:

```
Original List:1 2 3 4
Updated List:2 3 4
```

```
Updated List:3 4
Updated List:4
Updated List: List is empty
```

5) Program to Delete a New Node From the End of the Circular Linked List

Input:

1. #Adds data to the list
2. cl.add(1);
3. cl.add(2);
4. cl.add(3);
5. cl.add(4);

Output:

```
Original List:1 2 3 4
Updated List:1 2 3
Updated List:1 2
Updated List:1
Updated List: List is empty
```

6) Program to Delete a New Node From the Middle of the Circular Linked List

Input:

1. #Adds data to the list
2. cl.add(1);
3. cl.add(2);
4. cl.add(3);
5. cl.add(4);

Output:

```
Original List:1 2 3 4
Updated List:1 3 4
Updated List:1 4
Updated List:4
Updated List: List is empty
```

7) Program to Find the Maximum and Minimum Value Node From a Circular Linked List

Input:

1. #Adds data to the list
2. cl.add(5);
3. cl.add(20);
4. cl.add(10);
5. cl.add(1);

Output:

```
Minimum value node in the list: 1
```

```
Maximum value node in the list: 20
```

8) Program to Insert a New Node at the Beginning of the Circular Linked List

Input:

```
#Adding 1 to the list
cl.addAtStart(1);
#Adding 2 to the list
cl.addAtStart(2);
#Adding 3 to the list
cl.addAtStart(3);
#Adding 4 to the list
cl.addAtStart(4);
```

Output:

```
Adding nodes to the start of the list: 1
Adding nodes to the start of the list: 2 1
Adding nodes to the start of the list: 3 2 1
Adding nodes to the start of the list: 4 3 2 1
```

9) Program to Insert a New Node at the End of the Circular Linked List

Input:

```
#Adding 1 to the list
cl.addAtEnd(1);
#Adding 2 to the list
cl.addAtEnd(2);
#Adding 3 to the list
cl.addAtEnd(3);
#Adding 4 to the list
cl.addAtEnd(4);
```

Output:

```
Adding nodes to the end of the list: 1
Adding nodes to the end of the list: 1 2
Adding nodes to the end of the list: 1 2 3
Adding nodes to the end of the list: 1 2 3 4
```

10) Program to Insert a New Node at the Middle of the Circular Linked List

Input:

```
#Adds data to the list
cl.add(1);
cl.add(2);
cl.add(3);
```

5. cl.add(4);
6. #Inserting node '5' in the middle
7. cl.addInMid(5);
8. #Inserting node '6' in the middle
9. cl.addInMid(6);

Output:

```
Original list: 1 2 3 4
Updated List: 1 2 5 3 4
Updated List: 1 2 5 6 3 4
```

11) Program to Remove Duplicate Elements From a Circular Linked List

Input:

1. #Adds data to the list
2. cl.add(1);
3. cl.add(2);
4. cl.add(3);
5. cl.add(2);
6. cl.add(2);
7. cl.add(4);

Output:

```
Originals list:1 2 3 2 2 4
List after removing duplicates:1 2 3 4
```

12) Program to Search an Element in a Circular Linked List

Input:

1. #Adds data to the list
2. cl.add(1);
3. cl.add(2);
4. cl.add(3);
5. cl.add(4);
6. #Search for node 2 in the list
7. cl.search(2);
8. #Search for node in the list
9. cl.search(7);

Output:

```
Element is present in the list at the position : 2
Element is not present in the list
```

13) Program to Sort the Elements of the Circular Linked List

Input:

```
#Adds data to the list
cl.add(70);
cl.add(90);
cl.add(20);
cl.add(100);
cl.add(50);
```

Output:

```
Original list:70 90 20 100 50
Sorted list:20 50 70 90 100
```

Doubly Linked List Programs

1) Program to Convert a Given Binary Tree to Doubly Linked List

Input:

```
#Add nodes to the binary tree
bt.root = Node(1);
bt.root.left = Node(2);
bt.root.right = Node(3);
bt.root.left.left = Node(4);
bt.root.left.right = Node(5);
bt.root.right.left = Node(6);
bt.root.right.right = Node(7);
```

Output:

```
Nodes of generated doubly linked list: 4 2 5 1 6 3 7
```

2) Program to Create a Doubly Linked List From a Ternary Tree

Input:

```
#Add nodes to the ternary tree
tree.root = Node(5);
tree.root.left = Node(10);
tree.root.middle = Node(12);
tree.root.right = Node(15);
tree.root.left.left = Node(20);
tree.root.left.middle = Node(40);
tree.root.left.right = Node(50);
tree.root.middle.left = Node(24);
tree.root.middle.middle = Node(36);
tree.root.middle.right = Node(48);
tree.root.right.left = Node(30);
tree.root.right.middle = Node(45);
tree.root.right.right = Node(60);
```

Output:

```
Nodes of the generated doubly linked list: 5 10 20 40 50 12 24 36 48 15 30 45 60
```

3) Program to Create a Doubly Linked List of N Nodes and Count the Number of Nodes

Input:

```
#Add nodes to the list
dList.addNode(1);
dList.addNode(2);
dList.addNode(3);
dList.addNode(4);
dList.addNode(5);
```

Output:

```
Nodes of doubly linked list: 1 2 3 4 5
Count of nodes present in the list: 5
```

4) Program to Create a Doubly Linked List of N Nodes and Display it in Reverse Order

Input:

```
#Add nodes to the list
dList.addNode(1);
dList.addNode(2);
dList.addNode(3);
dList.addNode(4);
dList.addNode(5);
```

Output:

```
Original List: 1 2 3 4 5
Reversed List: 5 4 3 2 1
```

5) Program to Create and Display a Doubly Linked List

Input:

```
#Add nodes to the list
dList.addNode(1);
dList.addNode(2);
dList.addNode(3);
dList.addNode(4);
dList.addNode(5);
```

Output:

```
Nodes of doubly linked list: 1 2 3 4 5
```

6) Program to Delete a New Node From the Beginning of the Doubly Linked List

Input:

```
1. #Add nodes to the list
2. dList.addNode(1);
3. dList.addNode(2);
4. dList.addNode(3);
5. dList.addNode(4);
6. dList.addNode(5);
```

Output:

```
Original List: 1 2 3 4 5
Updated List: 2 3 4 5
Updated List: 3 4 5
Updated List: 4 5
Updated List: 5
Updated List: List is empty
```

7) Program to Delete a New Node From the End of the Doubly Linked List

Input:

```
1. #Add nodes to the list
2. dList.addNode(1);
3. dList.addNode(2);
4. dList.addNode(3);
5. dList.addNode(4);
6. dList.addNode(5);
```

Output:

```
Original List: 1 2 3 4 5
Updated List: 1 2 3 4
Updated List: 1 2 3
Updated List: 1 2
Updated List: 1
Updated List: List is empty
```

8) Program to Delete a New Node From the Middle of the Doubly Linked List

Input:

```
1. #Add nodes to the list
2. dList.addNode(1);
3. dList.addNode(2);
4. dList.addNode(3);
5. dList.addNode(4);
6. dList.addNode(5);
```

Output:

```
Original List: 1 2 3 4 5
Updated List: 1 2 4 5
Updated List: 1 4 5
Updated List: 1 5
Updated List: 5
Updated List: List is empty
```

9) Program to Find the Maximum and Minimum Value Node From a Doubly Linked List

Input:

1. #Add nodes to the list
2. dList.addNode(5);
3. dList.addNode(7);
4. dList.addNode(9);
5. dList.addNode(1);
6. dList.addNode(2);

Output:

```
Minimum value node in the list: 1
Maximum value node in the list: 9
```

10) Program to Insert a New Node at the Beginning of the Doubly Linked List

Input:

1. #Adding 1 to the list
2. dList.addAtStart(1);
3. #Adding 2 to the list
4. dList.addAtStart(2);
5. #Adding 3 to the list
6. dList.addAtStart(3);
7. #Adding 4 to the list
8. dList.addAtStart(4);
9. #Adding 5 to the list
10. dList.addAtStart(5);

Output:

```
Adding a node to the start of the list: 1
Adding a node to the start of the list: 2 1
Adding a node to the start of the list: 3 2 1
Adding a node to the start of the list: 4 3 2 1
Adding a node to the start of the list: 5 4 3 2 1
```

11) Program to Insert a New Node at the End of Doubly Linked List

Input:

1. #Adding 1 to the list
2. dList.addAtEnd(1);

```
3. #Adding 2 to the list
4. dList.addAtEnd(2);
5. #Adding 3 to the list
6. dList.addAtEnd(3);
7. #Adding 4 to the list
8. dList.addAtEnd(4);
9. #Adding 5 to the list
10. dList.addAtEnd(5);
```

Output:

```
Adding a node to the end of the list: 1
Adding a node to the end of the list: 1 2
Adding a node to the end of the list: 1 2 3
Adding a node to the end of the list: 1 2 3 4
Adding a node to the end of the list: 1 2 3 4 5
```

12) Program to Insert a New Node at the Middle of Doubly Linked List

Input:

```
1. #Add nodes to the list
2. dList.addNode(1);
3. dList.addNode(2);
4. #Adding node '3' in the middle
5. dList.addInMid(3);
6. #Adding node '4' in the middle
7. dList.addInMid(4);
8. #Adding node '5' in the middle
9. dList.addInMid(5);
```

Output:

```
Original list: 1 2
Updated List: 1 3 2
Updated List: 1 3 4 2
Updated List: 1 3 5 4 2
```

13) Program to Remove Duplicate Elements From a Doubly Linked List

Input:

```
1. #Add nodes to the list
2. dList.addNode(1);
3. dList.addNode(2);
4. dList.addNode(3);
5. dList.addNode(2);
6. dList.addNode(2);
7. dList.addNode(4);
8. dList.addNode(5);
9. dList.addNode(3);
```

Output:

```
Originals list: 1 2 3 2 2 4 5 3
List after removing duplicates: 1 2 3 4 5
```

14) Program to Rotate Doubly Linked List by N Nodes

Input:

```
#Add nodes to the list
dList.addNode(1);
dList.addNode(2);
dList.addNode(3);
dList.addNode(4);
dList.addNode(5);
```

Output:

```
Original List: 1 2 3 4 5
Updated List: 4 5 1 2 3
```

15) Program to Search an Element in a Doubly Linked List

Input:

```
#Add nodes to the list
dList.addNode(1);
dList.addNode(5);
dList.addNode(4);
dList.addNode(2);
dList.addNode(3);
```

Output:

```
Node is present in the list at the position : 3
Node is not present in the list
```

16) Program to Sort the Elements of the Doubly Linked List

Input:

```
#Add nodes to the list
dList.addNode(7);
dList.addNode(1);
dList.addNode(4);
dList.addNode(5);
dList.addNode(2);
```

Output:

```
Original list: 7 1 4 5 2
Sorted list: 1 2 4 5 7
```

Miscellaneous

Program to Convert cm to Feet and Inches

Program to Convert Feet to cm

Program to Convert Feet to Inches

Program to Convert Inches to cm

Program to Convert Inches to Feet

Program to Convert Kilometer to cm

Pyramid Programs in VB

Program to calculate Volume of Cone

Write a Program to calculate the Perimeter of Hexagon

Write a Program to calculate the Area of Rhombus

Write a Program to calculate the surface Area of Cone

Write a Program to calculate the Surface Area of Hemisphere

Write a Program to find the Perimeter of Ellipse

Write a program to calculate the Altitude of Isosceles Triangle

Write a Program to calculate the Area of Isosceles Triangle

Write a program to find the Volume of Tetrahedron

Write a program to find the Area of an Icosahedron

Write a program to find the Volume of Octahedron

Write a program to find the Area of Tetrahedron

Write a program to find the Surface Area of the Pentagonal Prism

Write a program to find the Area of the Rectangular Prism

Write a program to find the Surface Area of a Triangular Prism

Write a Program to find the Area of Hexagonal Prism

Write a Program to find the Volume of a Hexagonal Prism

Write a program to find the Volume of the Rectangular Prism

Program to Convert centimeter to millimeter

Program to Convert centimeter to meter

Program to Convert Feet to millimeter

Program to Convert Meter to Centimeter

Program to Convert mm to cm

Program to Convert millimeter to Feet

Program to Convert Feet to m

Program to Convert Inches to meter

Program to Convert Inches to mm

Program to Convert m to Feet and Inches

Write a program to calculate the Perimeter of a Rhombus

Write a program to find the quotient and remainder

Program to convert Kilobytes to bytes and bits

Program to find the area and perimeter of the semicircle

Program to find the area and perimeter of trapezium

Program to find the type of triangle from the given coordinates

Program to convert hours into minutes and seconds

Program to convert temperature degree from Celsius to Kelvin

Write a program to find the sum of even numbers