Formation of n-sided polygon with Positive integers using Programming Language

Koppuravuri Sai Charan[1], S.N.R.G. Bharat Iragavarapu[2]
[1]Student, Bachelor of Technology, Department of Electronics and Communication Engineering
[2]Assistant Professor, Department of Mathematics,
[1,2] GVP College of Engineering (Autonomous), Visakhapatnam, AP, India

***Abstract* -**In this paper, using computer programming language C++, we determine the number of n-sided polygons, by breaking a stick into (n-1) parts and by using a stick of given length say p units, p being a positive integer.

Key Words: Polygon, Triangle inequality, Polygon, inequality condition, Programming language C++

I. INTRODUCTION

In [1, 2, 3, 4, 5, 6, 7] the authors developed to form a triangle to 4, 5, 6, 7, 8, 9, 10 sided polygons through breaking a stick using programming language. But, for a polygon with sides more than10 this process is difficult by using programming language C.

In this paper using programming language C++ we form all possible n-sided polygons with positive integers through breaking a stickinto (n-1) parts. This n-sided polygon satisfies the condition sum of the (n-1) side lengths is greater than the largest side length n.

II. MAIN RESULT

2.1Algorithm

STEP 1: START

STEP 2: Enter the perimeter and Number of sides of polygon

STEP 3: Read the values as perimeter and sidesCount, both of type int

STEP 4: If the perimeter is less than sides count or sidesCount is less than 3 then go to step 24

STEP 5: Place the value perimeter / sidesCount in the entire array arr[] of size of sidesCount

STEP 6: Increment each of the numbers in array by 1 for perimeter % sidesCount times

STEP 7: Calculate the max value as (perimeter / 2) – 1 and min value is 1 both of type int

STEP 8: Initialize the currentIndex value as 0 initially of type int

STEP 9: Create a patternTree to store combinations or arr[] values

STEP 10: If the combination already exists in the tree then go to step 13

STEP 11: Add the combination to the tree by using traversal between nodes

STEP 12: print the combination or arr[] values

STEP 13: If arr[currentIndex] is 1 and currentIndex is less than sidesCount-1 then currentIndex is incremented by 1

STEP 14: arr[currentIndex] is decremented by 1

STEP 15: Declare a variable i and initialize it with currentIndex + 1

STEP 16: Increment i by 1

STEP 17: If i is not less than sidesCount then go to step 23

STEP 18: If i is not equal to sidesCount – 1 and arr[i] is not equal to arr[i + 1] or if i is equal to sidesCount – 1 then continue else go to step 17

STEP 19: Increment arr[i] by 1

STEP 20: If arr[i] is greater than max then go to step 22

STEP 21: Go to step 10

STEP 22: Decrement arr[i] by 1 and go to step 17

STEP 23: Increment arr[currentIndex] by 1

STEP 24: STOP

2.2Result Analysis

Step 1: Enter the stick length and number of sides of your polygon.

Step 2: The code displays

i) The Combinations of all possible side lengths of polygon with given stick length (perimeter) and

ii) Gives the Total number of such combinations

The above procedure can be explained below:

For example,

Consider the stick length (perimeter) as 20 and we want to form a 10 sided polygon.

Then by using the programming language C++ we display all the possible combinations and also we find the total number of such combinations. Here every combination satisfies the condition sum of the (n-1) side lengths is greater than the largest side length n. The code gives the results in an efficient time for wide range of sides even greater then 50.

The process of finding such combinations manually is very difficult that’s the reason we introduced the programming language.

Moreover we optimised the as efficient as possible to display the combinations in a very less time by using tree concept and by the elimination the loops as far as possible (as loops consume time). This can be understood from the above algorithm.

All the possible combinations of side lengths of a 10 sided polygon using the stick length 20 are shown in the Output screens as shown in Outputs section.

2.3 Outputs

```
Enter the perimeter and number of sides of polygon
20 10


2   2   2   2   2   2   2   2   2   2

1   2   2   2   2   2   2   2   2   3

1   1   2   2   2   2   2   2   3   3

1   1   1   2   2   2   2   3   3   3

1   1   1   1   2   2   3   3   3   3

1   1   1   1   1   3   3   3   3   3

1   1   1   1   1   2   3   3   3   4

1   1   1   1   1   1   3   3   4   4

1   1   1   1   1   1   2   4   4   4

1   1   1   1   1   1   1   4   4   5

1   1   1   1   1   1   1   2   5   6

1   1   1   1   1   1   1   1   6   6

1   1   1   1   1   1   1   1   5   7

1   1   1   1   1   1   1   1   4   8

1   1   1   1   1   1   1   1   3   9

1   1   1   1   1   1   1   3   4   6

1   1   1   1   1   1   1   2   4   7

1   1   1   1   1   1   2   3   4   5

1   1   1   1   1   1   3   3   3   5

1   1   1   1   1   1   2   3   3   6
```

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

Total Combinations: 40

```
Enter the perimeter and number of sides of polygon
23 9

     2     2     2     2     3     3     3     3     3

     1     2     2     3     3     3     3     3     3

     1     1     3     3     3     3     3     3     3

     1     1     2     3     3     3     3     3     4

     1     1     1     3     3     3     3     4     4

     1     1     1     2     3     3     4     4     4

     1     1     1     1     3     4     4     4     4

     1     1     1     1     2     4     4     4     5

     1     1     1     1     1     4     4     5     5

     1     1     1     1     1     3     5     5     5

     1     1     1     1     1     2     5     5     6
```

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

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

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

Total Combinations: 55

enter perimeter and number of sides of polygon
25 13

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

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

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

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

1	1	1	1	1	1	1	1	1	2	2	5	7
1	1	1	1	1	1	1	2	2	2	2	4	6
1	1	1	1	1	1	1	1	2	2	2	4	7
1	1	1	1	1	1	1	1	1	2	2	4	8
1	1	1	1	1	2	2	2	2	2	2	3	5
1	1	1	1	1	1	2	2	2	2	2	3	6
1	1	1	1	1	1	1	2	2	2	2	3	7
1	1	1	1	1	1	1	1	2	2	2	3	8
1	1	1	1	1	1	1	1	1	2	2	3	9
1	1	1	2	2	2	2	2	2	2	2	2	4
1	1	1	1	2	2	2	2	2	2	2	2	5
1	1	1	1	1	2	2	2	2	2	2	2	6
1	1	1	1	1	1	2	2	2	2	2	2	7
1	1	1	1	1	1	1	2	2	2	2	2	8
1	1	1	1	1	1	1	1	2	2	2	2	9
1	1	1	1	1	1	1	1	1	2	2	2	10
1	1	1	1	1	1	1	1	1	1	2	2	11

Total Combinations: 75

REFERENCES

[1] S.N.R.G.Bharat Iragavarapu, M.Anuraag Chandra Breaking a Stick to form a triangle, Journal of Computational Mathematics and Ap-plied Mathematics, Volume 1, Issue 1, Mantech Publications, 2016.

[2] S.N.R.G.Bharat Iragavarapu, Chandolu Somarjun, Breaking a Stick to form a Pentagon with Positive Integers using Programming Language Python, International Research Journal of Engineering and Technology (IRJET), Volume 4, Issue 8, Pg 95-97, Aug- 2017.

[3] S.N.R.G.Bharat Iragavarapu, Breaking a Stick to form a Hexagon with Positive Integers using Programming Language Python, International Journal for Scientific Research and Development (IJSRD), Vol. 5, Issue 06, 2017, Pg 1005-1006.

[4] S.N.R.G.Bharat Iragavarapu, Breaking a Stick to form a Heptagon with Positive Integers Journal of Journal of information technology and sciences, MAT Journals, Page 1-5, 2017

[5] S.N.R.G.Bharat Iragavarapu, vasudeva Rao, Breaking a Stick to form a Octagon with Positive Integers, International Journal for Research in Applied Science & Engineering Technology (IJRASET**),** Volume 5 Issue VIII, ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor:6.887, Pg 2183-2187,August 2017.

[6] S.N.R.G.Bharat Iragavarapu, konathala Chetan, Breaking a Stick to form a Nanogon with Positive Integers using Programming Language MATLAB, International Research Journal of Engineering and Technology (IRJET), Volume 4, Issue 9, Pg 37-40, Sep- 2017.

[7] S.N.R.G.Bharat Iragavarapu, vasudeva Rao, Breaking a Stick to form a Decagon with Positive Integers using MATLAB International Journal for Research in Applied Science & Engineering Technology (IJRASET**),** Volume 6 Issue III, ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor:6.887, Pg 1952-1955, March 2018.