JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Breaking a Stick to form a Polygon with Positive Integers using Python

[1]S N R G BHARAT IRAGAVARAPU, [2]SATYA VAMSI KUMAR APPALA
[1]Assistant Professor, [2]Lecturer
[1] Department of Mathematics, [2] Computer science department
[1]Gayatri Vidya Parishad College of Engineering (Autonomous), Visakhapatnam, India
[2]B V Raju College, Vishnupur, Bhimavaram, India.

Abstract: In this paper, using a computer programming language, we determine the number of polygons (e.g. triangle/pentagon/decagon etc.) that can be formed by using a stick of given length say n units, n being a positive integer greater than 2.

Keywords **- Triangle Inequality, Polygon, Inequality Condition, Python**

I. INTRODUCTION

In [1, 2, 3, and so on] we formed a triangle, quadrilateral and pentagon, etc. through breaking a stick using programming language. In this paper, by using Python language we form all possible polygons by generalizing with positive integers through breaking stick, for any such n. For example, suppose we take a stick of length 15 units and cut this stick at 5 places to form 6 parts of the stick. Let a, b, c, d, e, f be the lengths of the six parts of the stick and assume that a, b, c, d, e, f are positive integers. Hence we have the basic relation $a + b + c+ d + e + f = n$. Here number n is given but a, b, c, d, e, f are variable numbers. For formation of a hexagon having side lengths a, b, c, d, e, f we need to see that the condition $a + b + c + d + e > f$ and f is the largest side length compare to others i.e. the sum of the remaining side lengths is greater than the largest side length. Here $(a, b, c, d, e, f) = (b, c, d, e, f, a) = (c, d, e, f, a, b) = (d, e, f, a, b, c) = (e, f, a, b, c, d) = (f, a, b, c, d, e)$.

This is very difficult if the numbers of our selection are considerably large. Now our aim is to form any polygon with Positive Integers using Python language

II. MAIN RESULT

2.1Algorithm

```
Step 1: start
Step 2: import combinations and chain modules from itertools package
Step 3: Read the length of the stick
Step 4: Read number of parts you want to breakdown stick
Step 5: Find possible permutations and combinations (where permutations and combinations == no of parts+1) which
        equals to stick length.
Step 6: consider only the combinations list after removing permutations from the list.
Step 7: Each side in the combinations list can form a valid polygon or not by checking some of the conditions like
          a. Sum of n-1 sides is greater than nth side
          b. nth side should be greater than all other sides
          return the sides which satisfy above conditions
Step 8: print number of polygons that can be formed and their possible sides.
Step 9: stop
```

2.2 Python program

```
from itertools import combinations, chain
length = int(input("Enter the stick length:"))
# length = int(length)
parts = int(input("Enter number of parts you want to breakdown stick:"))
# parts = int(parts)
def perm_comb(n):
    """
    :param n:
    :return: a list of permutations and combinations of values equal to n
    Generate the series of +ve integer lists which sum to a +ve integer, n.
    """
    from operator import sub
    b, mid, e = [0], list(range(1, n)), [n]
    splits = (d for i in range(n) for d in combinations(mid, i))
    return (list(map(sub, chain(s, e), chain(b, s))) for s in splits)
perm_comb_lst = []
for p in perm_comb(length):
    """
    possible permutations and combinations(where permutations and combinations == no of parts+1) which equals to stick length
    :return: permutations and combinations list
    """
    if len(p) == parts+1:
        perm_comb_lst.append(p)
def remove_perm(perm_comb_lst):
    """
    :return: combinations list
    """
    return([list(i) for i in {*[tuple(sorted(i)) for i in perm_comb_lst]}])
comb_lst = remove_perm(perm_comb_lst)
# print("combinations list having length equals to no of parts+1 is :\n", comb_lst)
def check_valid_polygon(sides):
    """
    each side in the comb_lst can form a valid polygon or not by checking some of the conditions like
        a. Sum of n-1 sides is greater than nth side
        b. nth side should be greater than all other sides
    :return: sides which satisfy above conditions
    """
    try:
        for side in sides:
            other_sides = (sum(sides) - side)
            if side > other_sides:
                return ''
            elif side == other_sides:
                return ''
        else:
```

```
        # print("sides in check polygon",sides)
        last_element = sides[len(sides)-1]
        last_element_index=len(sides)
        for i in sides[0:last_element_index-1]:
          if last_element > i:
            flag = True
            pass
          else:
            flag = False
            break
        if flag == True:
          return sides
        else:
          return ''
        return sides
    except Exception as err:
        raise err
sides = comb_lst
res_list = []
for i in sides:
    res_list.append(check_valid_polygon(i))
print("Response list is", res_list)
polygon_list=[]
for i in res_list:
    if i != '':
        polygon_list.append(i)
    else:
        continue;
print("No of polygons that can be formed are: " + str(len(polygon_list)) + " and their sides list are:\n", polygon_list)
```

2.2 Result analysis

We are required to display all the combinations that follow the polygon inequality. This can be achieved with help of the following steps.
Step 1: Write all permutations in form of triads for a given integer.
Step 2: Eliminate equivalent permutations so that only the combinations remain.
Step 3: Display only the combinations that satisfy the triangle inequality.
The above procedure can be explained below:

For example,

- Consider a stick length 12.
- Let the combinations are (1, 1, 1, 1, 3, 5), (1, 1, 1, 2, 2, 5), (1, 1, 1, 2, 3, 4), (1, 1, 2, 2, 2, 4), (1, 2, 2, 2, 2, 3).
- The total number of hexagons with stick length 12 are 5

  We can represent this result in outputs

2.3 Outputs

We will get different outputs for different stick lengths.

Output-1

```
Enter the stick length:12
Enter number of parts you want to breakdown stick: 5
No of polygons that can be formed are: 5 and their sides list are:
        [[1, 1, 1, 2, 3, 4],
        [1, 1, 2, 2, 2, 4],
        [1, 1, 1, 1, 3, 5],
        [1, 1, 1, 2, 2, 5],
        [1, 2, 2, 2, 2, 3]]
```

By using above combinations we can form a hexagon.

Output-2

```
Enter the stick length: 12
Enter number of parts you want to breakdown stick:4
No of polygons that can be formed are: 6 and their sides list are:
        [[1, 1, 1, 4, 5],
         [1, 1, 2, 3, 5],
         [1, 1, 3, 3, 4],
         [1, 2, 2, 2, 5],
         [1, 2, 2, 3, 4],
         [2, 2, 2, 2, 4]]
```

By using above combinations we can form a pentagon

Output-3

```
Enter the stick length:25
Enter number of parts you want to breakdown stick:2
No of polygons that can be formed are: 12 and their sides list are:
[[8, 8, 9],
 [6, 8, 11],
 [7, 7, 11],
 [6, 9, 10],
 [4, 9, 12],
 [6, 7, 12],
 [5, 9, 11],
 [5, 8, 12],
 [7, 8, 10],
 [3, 10, 12],
 [2, 11, 12],
 [4, 10, 11]]
```

By using above combinations we can form a triangle.

III. Conclusion

By using this program, we can easily find the number of polygons that can be formed through breaking a stick using python, it becomes novel and easy process.

REFERENCES

[1] S.N.R.G.Bharat Iragavarapu, M.Anuraag Chandra 2016. Breaking a Stick to form a triangle, Journal of Computational Mathematics and Ap-plied Mathematics, Mantech Publications 1,(1), 1-10.

[2] S.N.R.G.Bharat Iragavarapu, J. kushwanth, 2017. Formation of a Integer Quadrilateral through Breaking a Stick, International Journal of Innovative research and Advanced Studies, 4(3): 350-352.

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

[4] S.N.R.G.Bharat Iragavarapu, Konathala Chetan, 2017. Breaking a Stick to form a Nonagon with Positive Integers using Programming Language MATLAB, International Research Journal of Engineering and Technology (IRJET), 4(9): 37-40.