Generate Pythagorea

A Pythagorean triplet is a set of three positive integers a, b and c such that $a^2 + b^2 = c^2$. Given a limit, generate all Pythagorean Triples with values smaller than given limit.

```
Input : limit = 20
Output : 3 4 5
         8 6 10
         5 12 13
         15 8 17
         12 16 20
```

A **Simple Solution** is to generate these triplets smaller than given limit using three nested loop. For every triplet, check if Pythagorean condition is true, if true, then print the triplet. Time complexity of this solution is $O(limit^3)$ where 'limit' is given limit.

An **Efficient Solution** can print all triplets in O(k) time where k is number of triplets printed. The idea is to use square sum relation of Pythagorean triplet, i.e., addition of squares of a and b is equal to square of c, we can write these number in terms of m and n such that,

$a = m^2 - n^2$
$b = 2 * m * n$
$c = m^2 + n^2$

because,

$a^2 = m^4 + n^4 - 2 * m^2 * n^2$
$b^2 = 4 * m^2 * n^2$
$c^2 = m^4 + n^4 + 2* m^2 * n^2$

We can see that $a^2 + b^2 = c^2$, so instead of iterating for a, b and c we can iterate for m and n and can generate these triplets.

Below is the implementation of above idea :

C++

```
// C++ program to generate pythagorean
// triplets smaller than a given limit
#include <bits/stdc++.h>

// Function to generate pythagorean
// triplets smaller than limit
void pythagoreanTriplets(int limit)
{

    // triplet: a^2 + b^2 = c^2
    int a, b, c = 0;

    // loop from 2 to max_limitit
    int m = 2;

    // Limiting c would limit
    // all a, b and c
    while (c < limit) {

        // now loop on j from 1 to i-1
        for (int n = 1; n < m; ++n) {

            // Evaluate and print triplets using
            // the relation between a, b and c
            a = m * m - n * n;
            b = 2 * m * n;
            c = m * m + n * n;

            if (c > limit)
                break;

            printf("%d %d %d\n", a, b, c);
        }
        m++;
    }
}

// Driver Code
int main()
{
    int limit = 20;
    pythagoreanTriplets(limit);
    return 0;
}
```

Java

```java
// Java program to generate pythagorean
// triplets smaller than a given limit
import java.io.*;
import java.util.*;

class GFG {

    // Function to generate pythagorean
    // triplets smaller than limit
    static void pythagoreanTriplets(int limit)
    {

        // triplet: a^2 + b^2 = c^2
        int a, b, c = 0;

        // loop from 2 to max_limitit
        int m = 2;

        // Limiting c would limit
        // all a, b and c
        while (c < limit) {

            // now loop on j from 1 to i-1
            for (int n = 1; n < m; ++n) {
                // Evaluate and print
                // triplets using
                // the relation between
                // a, b and c
                a = m * m - n * n;
                b = 2 * m * n;
                c = m * m + n * n;

                if (c > limit)
                    break;

                System.out.println(a + " " + b + " " + c);
            }
            m++;
        }
    }

    // Driver Code
    public static void main(String args[])
    {
        int limit = 20;
        pythagoreanTriplets(limit);
    }
}

// This code is contributed by Manish.
```

Python3

```python
# Python3 program to generate pythagorean
# triplets smaller than a given limit

# Function to generate pythagorean
# triplets smaller than limit
def pythagoreanTriplets(limits) :
    c, m = 0, 2
```

```
    # Limiting c would limit
    # all a, b and c
    while c < limits :

        # Now loop on n from 1 to m-1
        for n in range(1, m) :
            a = m * m - n * n
            b = 2 * m * n
            c = m * m + n * n

            # if c is greater than
            # limit then break it
            if c > limits :
                break

            print(a, b, c)

        m = m + 1


# Driver Code
if __name__ == '__main__' :

    limit = 20
    pythagoreanTriplets(limit)


# This code is contributed by Shrikant13.
```

C#

```
// C# program to generate pythagorean
// triplets smaller than a given limit
using System;

class GFG {

    // Function to generate pythagorean
    // triplets smaller than limit
    static void pythagoreanTriplets(int limit)
    {

        // triplet: a^2 + b^2 = c^2
        int a, b, c = 0;

        // loop from 2 to max_limitit
        int m = 2;

        // Limiting c would limit
        // all a, b and c
        while (c < limit) {

            // now loop on j from 1 to i-1
            for (int n = 1; n < m; ++n)
            {

                // Evaluate and print
                // triplets using
                // the relation between
                // a, b and c
                a = m * m - n * n;
```

```
                b = 2 * m * n;
                c = m * m + n * n;

                if (c > limit)
                    break;

                Console.WriteLine(a + " "
                            + b + " " + c);
            }
            m++;
        }
    }

    // Driver Code
    public static void Main()
    {
        int limit = 20;
        pythagoreanTriplets(limit);
    }
}

// This code is contributed by anuj_67.
```

PHP

```
<?php
// PHP program to generate pythagorean
// triplets smaller than a given limit

// Function to generate pythagorean
// triplets smaller than limit
function pythagoreanTriplets($limit)
{

    // triplet: a^2 + b^2 = c^2
    $a;
    $b;
    $c=0;

    // loop from 2 to max_limitit
    $m = 2;

    // Limiting c would limit
    // all a, b and c
    while ($c < $limit)
    {

        // now loop on j from 1 to i-1
        for ($n = 1; $n < $m; ++$n)
        {

            // Evaluate and print
            // triplets using the
            // relation between a,
            // b and c
            $a = $m *$m - $n * $n;
            $b = 2 * $m * $n;
            $c = $m * $m + $n * $n;

            if ($c > $limit)
                break;
```

```
            echo $a, " ", $b, " ", $c, "\n";
        }
        $m++;
    }
}

    // Driver Code
    $limit = 20;
    pythagoreanTriplets($limit);

// This code is contributed by ajit.
?>
```

Output :

```
3 4 5
8 6 10
5 12 13
15 8 17
12 16 20
```

Time complexity of this approach is O(k) where k is number of triplets printed for a given limit (We iterate for m and n only and every iteration prints a triplet)

Note: The above method doesn't generate all triplets smaller than a given limit. For example "9 12 15" which is a valid triplet is not printed by above method. Thanks to Sid Agrawal for pointing this out.

References:
https://en.wikipedia.org/wiki/Formulas_for_generating_Pythagorean_triples

This article is contributed by **Utkarsh Trivedi**. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above

Recommended Posts:

Generate integer from 1 to 7 with equal probability

Generate all unique partitions of an integer

Generate all permutation of a set in Python

To Generate a One Time Password or Unique Identification URL

Generate all palindromic numbers less than n

Generate k digit numbers with digits in strictly increasing order

Generate all cyclic permutations of a number

Generate a pythagoras triplet from a single integer

Pythagorean Triplet with given sum

Generate a list of n consecutive composite numbers (An interesting method)

Algorithm to generate positive rational numbers

Count number of triplets with product equal to given number with duplicates allowed

Check if a number is a Pythagorean Prime or not

Count number of triplets (a, b, c) such that a^2 + b^2 = c^2 and 1 <= a <= b <= c <= n

Generate elements of the array following given conditions

Improved By : Manish_100, jit_t, vt_m, Shubham Mishra 22

Article Tags : Mathematical

Practice Tags : Mathematical

7

2

To-do Done

Based on **6** vote(s)

Feedback/ Suggest Improvement | Add Notes | Improve Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

A computer science portal for geeks

5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305
feedback@geeksforgeeks.org