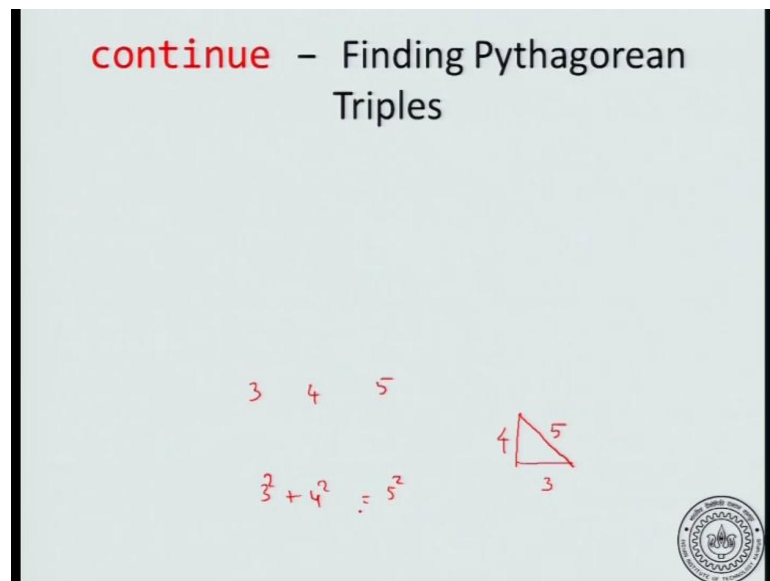


Introduction to Programming in C
Prof. Satyadev Nandakumar
Department of Computer Science and Engineering
Indian Institute of Technology, Kanpur

Lecture - 19

Let us do a sample program using continue statements, I will introduce the problem initially, the problem is that of finding Pythagorean triples.

(Refer Slide Time: 00:13)



By the way Pythagorean triples are numbers like are triplets of numbers like 3, 4 and 5. Because, you know that 3 square plus 4 square equal to 5 square. So, the Pythagorean triples because there can be a right triangle, where let say the basis 3, the altitudes is 4 and the hypotenuse is 5. So, 3, 4 and 5 could be the sides of right triangle, because they satisfy the Pythagorean identity.

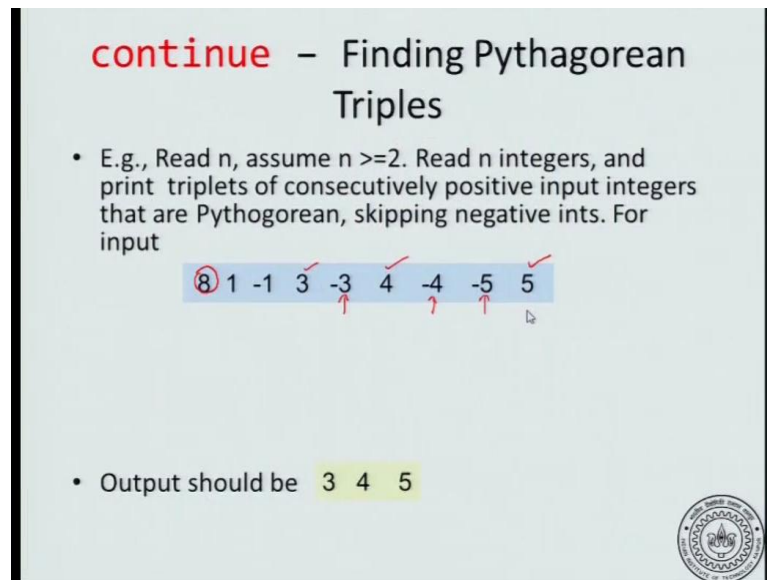
(Refer Slide Time: 00:53)

continue – Finding Pythagorean Triples

- E.g., Read n , assume $n \geq 2$. Read n integers, and print triplets of consecutively positive input integers that are Pythagorean, skipping negative ints. For input

8 1 -1 3 -3 4 -4 -5 5

Output should be 3 4 5



So, here is a problem we are given a stream of numbers and let us say there are n numbers. So, the initial number says how many other numbers there are. So, 8 says that there are 8 numbers to process, after you read n , n is greater than or equal to 2, you have to read n integers and then you have to identify Pythagorean triplets occurring consecutively. By consecutively we will say that consecutive positive integers. Because, in the middle there could be negative numbers so you have to just ignore them.

For example, you have that 3, 4 and 5 are consecutive, positive entries in this data. Because, minus 3, minus 4 and minus 5 are negative numbers. So, consecutive in this context need not mean that they occur together, it just means that, if we ignore the negative numbers and between then they are together. So, we have to identify all such Pythagorean triples. So, in this case the Pythagorean triple in the input sequence is 3, 4 and 5.

(Refer Slide Time: 02:24)

Code – Part I

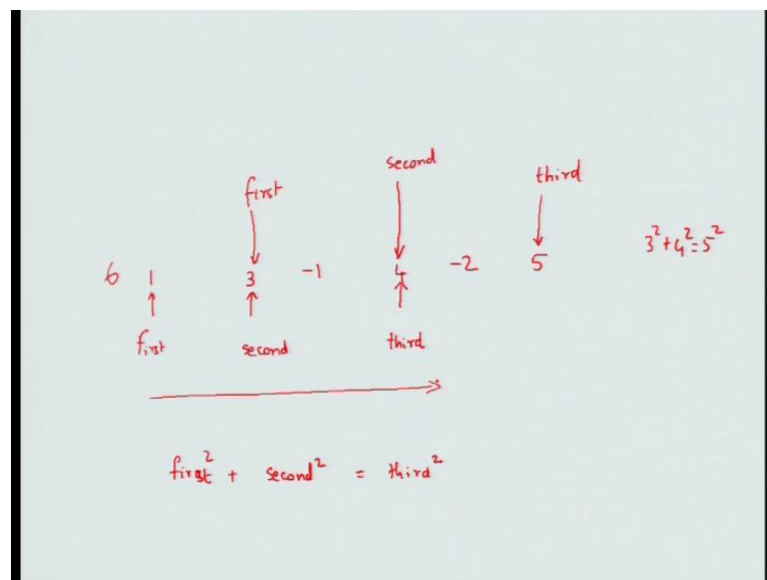
```
#include <stdio.h>
main()
{
    int curr, prev, pprev; /* current, prev, pprev positive nos. */
    int n;                /* number of integers */
    int i;                /* for loop counter */
    int count = 0;       /* no. of positive ints seen yet */
    scanf("%d", &n);

    /* the loop (continued on next slide) */

} /* End of main */
```

So, let us try to code it. I hope you see how it can be done? So, let us try to do it my hand.

(Refer Slide Time: 02:39)



So, let us say that I have and then some negative numbers in between and so on. So, some positive numbers, some negative numbers in between till I find the... So, I have let us if four, six numbers. So, the input is of the following for what I need to do is, at any point I may have to remember some triple. So for example, the first triple that I will find is the following. So, this is the first number, this is the second number and this is the

third number and what I have to do is to check whether, first square plus second square equal to third square. So, this is what I have to check?

Now, suppose that 1, 3 and 4 are not a Pythagorean triple, they are not. Because, 1 square plus 3 square is not 4 square. Then, what do you have to do? You have to advance all these first, second and third variables. So, let us try to advance the third variable, the next interesting number is 5, because that is the next positive number. So, the next iteration should check for the following, this should be the third number, 4 should be the second number and 3 should be the first number.

If you do that, then you know that 3 square plus 4 square equal to 5 square and you will identify a Pythagorean triple. So, what we do is, that we have to shift all these variables, first, second and third by one positive entry. So, this is what we have to do, we have to remember three numbers, the current number that we have seen, the previous positive number that we have seen and the previous to previous positive number that we have seen. So, this is one situation where you need to remember three variables.

And once you check whether the current triplets satisfy it, if you satisfy it fine. If you do not satisfy it, you have to advance the variables by one. So, first will take over second, second will take over third and third will go to the next positive number. So, this is the method of programming this, let us try to code this self ((Refer Time: 05:30)). So, we will write the code as follows, we need three variables, the current number, the previous number and the previous to previous number.

Currently will leave all of them undefined, n is the number of integers to read, i is we will eventually try to do. So, for the for loop we need a counter. So, I will basically count from 1 to n to ensure that n numbers have been read. I will also have an extra variable call count, i is suppose to count the numbers seen so far and count will count the positive numbers seen, so far. So, I need two half them in this code I mean in that. Now, after you do that you scan the n, which tells you how many numbers are there in the input? Now, a for loop has to go here which will do most of the work in the code. So, let us see what that loop looks like?

(Refer Slide Time: 06:48)

The Loop

```
for (i=0; i < n ; i = i+1) {
    scanf("%d", &curr);
    if (curr <= 0) { continue; } /* skip non-positive nos. */
    if (count == 0) {
        pprev = curr; count = 1;
    } else {
        if (count == 1) {
            prev = curr; count = 2;
        } else { /* count is 2 and will remain 2 */
            if (pprev*pprev + prev*prev == curr*curr){
                /* Pythagorean triple found */
                printf( "%d %d %d\n", pprev, prev, curr);
                pprev = prev;
                prev = curr;
            }
        }
    }
}
```

$pprev^2 + prev^2 = curr^2$

So, recall what we did by hand, you will look at the current number which is the next number to read, if the next number is 0 or less than 0 you say continue. So, this is the application of the continue statement here. So, if says if the current number is not positive, you just go onto the next iteration of the loop. Now, here is some logic which is not easy to read, but we can motivate it the following, if the current number that I have seen is the first positive number. When obviously, then this was the first number that I have read.

Therefore, there was no previous number and there was no previous to previous number. So, I will because this the first positive number that I am reading, then I will just set there the previous to previous number is the current number, also I have seen one positive number. So, I will say increment count, count equal to 1. So, if the current number that I have seen is positive and it is not the first positive number.

That means, if count equal to 1 I already seen one positive number, then what to you do is, you know that there is a previous to previous number, you set the previous number to the current number and you continue the loop setting that count equal to 2, which says that I have seen two positive numbers. So, I have a previous to previous number and I have a previous number, now I will read the next number. This is because in order to identify a triple, you need at least three numbers.

So, previous to previous and previous should already been to some positive values in the input, this is why we initially said that, we need at least two inputs. So, we will go back to the loop if count equal to 1, otherwise let us say that count is at least 2, so it is 2 or more. So, in this case we will just say that as for as count is consent I do not need to keep track of how many positive numbers are needed? It was used only to see that I have at least two positive numbers to begin with. So, that I can add the next number as the possible third number in the triple.

So, I will not update count from now one, you can also do that, but count after words serves no purpose. So, I will say that count is 2 and I will just adopt the convention that it will remain to. So, I will seen at least two positive numbers, now I have also a third number in the current. So, you have previous to previous, you have previous and you have current. So, these are the three numbers that you have.

So, what you have to check is, whether previous to previous square plus previous square equal to currents square. So, that is what we will check, we will check whether previous to previous square plus previous square is equal to current square, if that is true then you have found the Pythagorean triple. So, you will just say that I will printf that I have found the Pythagorean triple, which is found by previous to previous, previous and current.

Now, what I will do if the Pythagorean triple is found is that I will advance previous to previous by one. So, previous to previous will become previous, previous will become current. So, recall that figure that I first true and then we will go back to the loop. So, this is the code for kind identifying the Pythagorean triples and the encodes exactly the logic that we did by hand.