I sent this as a C++ file that is much easier to read. You can open it in Visual Studio if you use windows.
C++ Program for calculating sums:

```
#include <cstdlib>      // → These are pre-compiler directives. The tell the computer
#include <iostream>       // what libraries to include for C++.
using namespace std;
int main(int argc, char** argv) {         // → This is the function() main. It will run the code.
  int number, total = 0;          // → We are declaring two integer variables: number & total
                                    //    We are also initializing total with the value of 0
  cout << "This program will calculate a sum of numbers between 1 and any positive number
you enter \n";  //→cout statements print this line to the screen
  cout << "Please enter a positive integer: ";   //       → It will print everything in “quotes”
  cin >> number;              //  → cin (pronounced c-in) statements store the input from the
                                    user into the variable, in this case, number.
  if (number > 0)                 //→ if statement, used for data validation in this case.
    for (int count = 1; count <= number; count++){ //→ for loop, explained below
       total += count;}   //→ total is the accumulator.
                                    //This line means total = total + count. We establish the
                                    //variable count in the for loop.
     cout << "The sum of numbers between 1 and " << number << " equals " << total;
  return 0;}      // → THis line prints to the screen.
```

for loop -- good for a known number of iterations - a count controlled loop

must have three things:
1. initialize a counter
2. compare the counter to the max value
3. update the counter after each iteration.

```
for (initialization; test; update) {
        statement;
        statement;
        statement;
```

In our for loop above, we have:
for (int count = 1; count <= number; count++)
If we break this down:
int count = 1 → We are declaring and initializing the variable count to equal 1. We do this so we have a counter throughout loop.
count <= number → as long as the count variable is less than or equal to the user input number. This is the test section of our for loop.
count++ → This is the “update” section. This is where we will increment the count variable once per loop iteration, (each loop cycle).
Pseudo code for the above code.

```
start
  Declarations
        number = 0
        total = 0
        count = 1
  output “Enter a number to calculate the sum from one through the number you enter.”
```

```
  input number
  for (count, count <= number, count ++)
        total = total + count
  end for
  output “The total of the numbers 1 through “ number  “is” total
end
```

You also have while loops. These are great for running a program continuously until a sentinel value is input, or for validating data. You could also use a while loop for the above problem, although, I don’t think it is quite as easy, we’ll write the pseudocode for it:

First, here is the basic definition and format for a while loop (This specifically is for C++)
While Loops - repeat one or more statements as long as a boolean expression is true

```
while (condition) {
        statement;
        statement;
}
```

Now for the pseudocode!

```
start
  Declarations
        number = 0
        total = 0
        count = 1
  output “Enter a number to calculate the sum from one through the number you enter.”
  input number
  while ( count <= number){
        total = total + count
        count ++
  }endwhile
  output “The total of the numbers 1 through “ number  “is” total
end
```

We are accomplishing the same thing as the for loop, we just need an extra line of code to add the count incrementer. If you have trouble, it helps to hand execute the loop. Write what the value that each variable holds as each loop iterates, or cycles through.

It should look something like this:
Lets pretend we enter the number 5 for the number variable. (We know the output should be 15 in this case)
First loop:

```
 while ( count <= number)        (1 <= 5 )   count is 1, number is 5
        total = total + count    ( total = 0 + 1)  total is 1 after the addition, count is 1
        count ++                 count is now 2
```

Second Loop:

```
  while ( count <= number)       ( 2 <= 5 )  count is 2, number is still 5
        total = total + count    ( total = 1 + 2 )  total now equals 3
        count ++                 count is now 3
```

Third loop:

```
  while ( count <= number)       ( 3 <= 5)  count is 3, number will always be 5
```

```
        total = total + count       (total = 3 + 3)  total now equals 6
        count ++                    count is now 4
```

Fourth Loops:

```
  while ( count <= number)          ( 4 <= 5 ) count is 4, number is 5
        total = total + count       (total = 6 + 4) total is now 10
        count ++                    count is now 5
```

Fifth Loop: (Last one!)

```
  while ( count <= number)          ( 5 <= 5 ) count is equal to number, so we iterate
        total = total + count       (total = 10 + 5 ) total now equals 15
        count ++                    count now equals 6
```

The computer will test the loop one last time
It just won't execute

```
  while ( count <= number)          ( 6 <= 5 ) This isn't true, so we stop...
        total = total + count
        count ++
```

We now have that the total of the numbers from 1 through 5 is 15, which is correct
You can do this same thing for the for loop above if it will help to make sense.
Hope this helps!