Iteration in C

It is often necessary to execute a sequence of statements repeatedly to compute certain value. Every imperative programming language[a] provide different constructs (statements) to perform this iterative computation.

[a] A language that uses commands to change the content of a location.

Example III

Write a program to compute the following sum:

$$S_n = 1 + 2 + 3 + \cdots + n,$$

where n is an input.

The best way to do it is to use the closed form or the formula

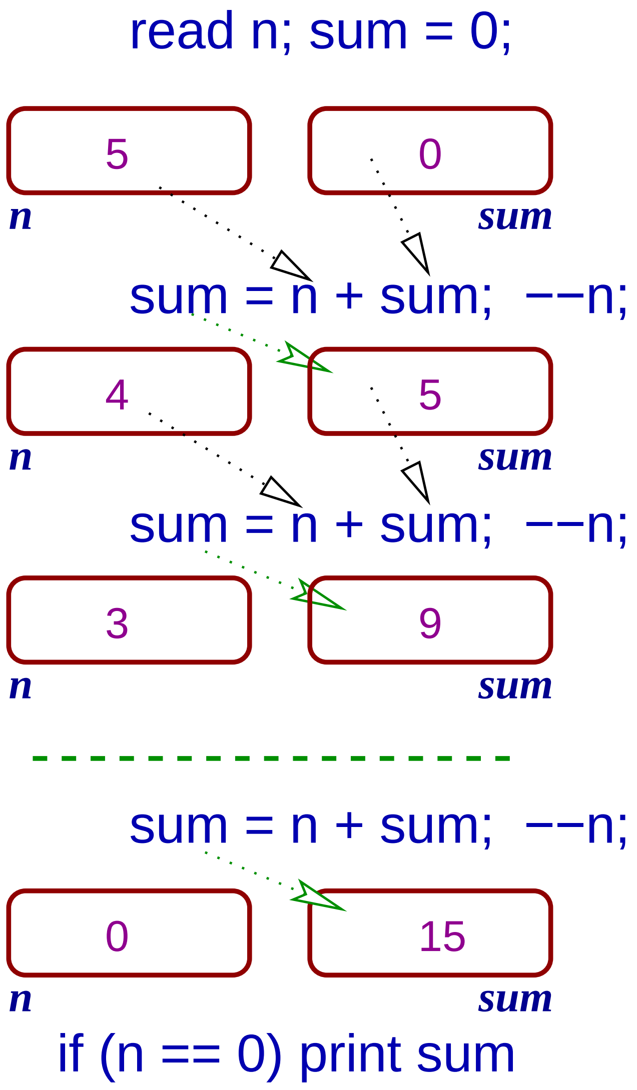
$$S_n = \frac{n(n+1)}{2}.$$

```
#include <stdio.h>
int main() // temp26a.c
{
    int n;
    printf("Enter a +ve integer: ");
    scanf("%d", &n);
    printf("1+ ...+%d = %d\n",
            n, n*(n+1)/2);
    return 0;
}
```

```
$ cc -Wall temp26a.c
$ ./a.out
Enter a +ve integer:  5
1+ ...+5 = 15
```

An Alternate Way

But there are alternate ways of doing this computation. This method is not good for this particular problem, but it can be generalized to use for similar problems where the closed form is difficult to obtain.



```
read n; sum = 0;
sum = n + sum;  −−n;
sum = n + sum;  −−n;
- - - - - - - - - - - - - - - - -
sum = n + sum;  −−n;
if (n == 0) print sum
```

```
#include <stdio.h>
int main() // temp26.c
{
    int n, sum = 0;
    printf("Enter a +ve integer: ");
    scanf("%d", &n);
    while(n > 0) {
        sum = n + sum ;
        --n ;
    }
    printf("sum: %d\n", sum);
    return 0;
}
```

```
$ cc -Wall temp26.c
$ ./a.out
Enter a +ve integer:  5
sum:  15
```

Note

The code

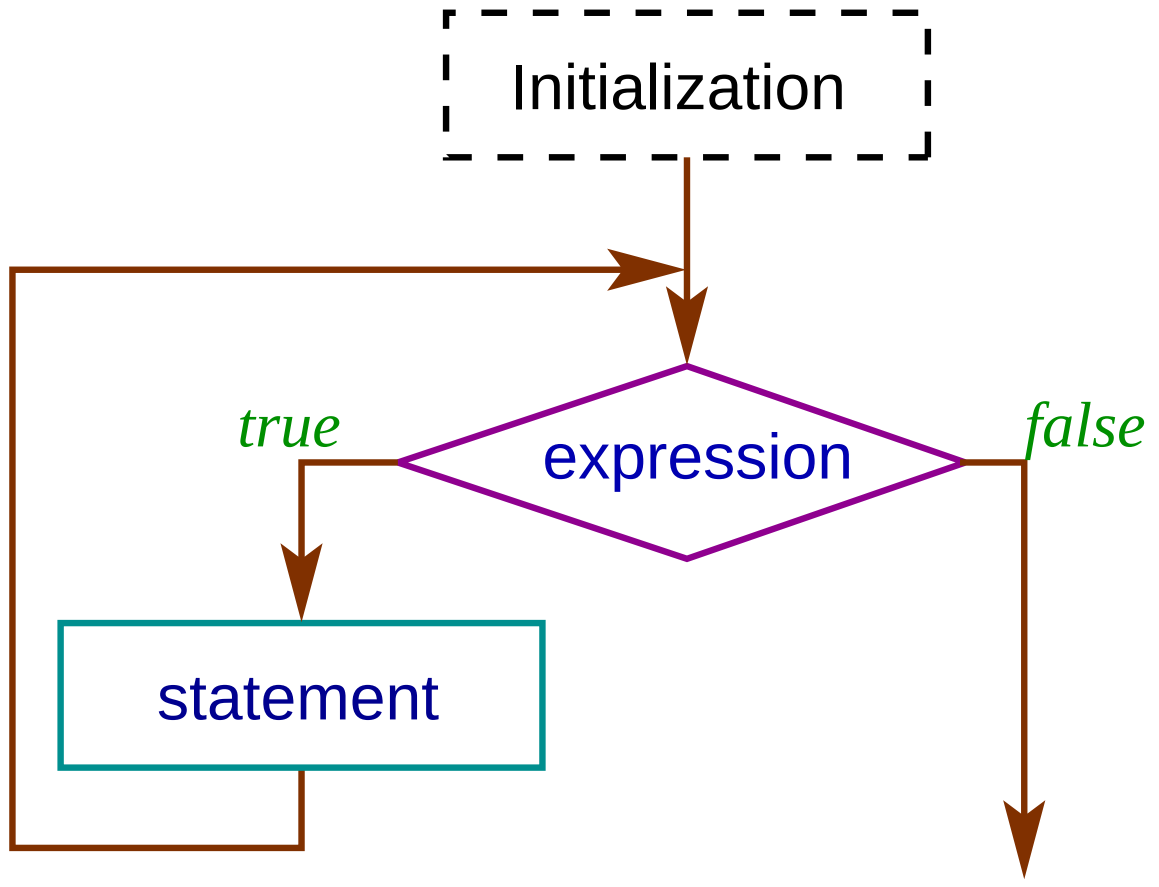
```
sum = n + sum ;

--n ;
```

is executed repeatedly with decremented values of `n` as long as `n` is not equal to zero.

`while` Statement

`while` statement in C is one of the constructs used for iterative computation. The structure or syntax of `while` is

while (expression) statement


Initialization
true
expression
false
statement

Note

- The `while`-loop will not be entered if the loop-control expression evaluates to false (zero) even before the first iteration.
- break statement can be used to come out of the `while` loop.

More Succinct Code

```
while(n>0) sum += n-- ;
```

Note

- Our `while` program destroys the input data. That can be avoided by introducing a third variable where the value of `n` can be copied.
- This program structure can be used to compute any sum of the following form where c is a known positive integer.

$$S_n = 1^c + 2^c + \cdots + n^c = \sum_{k=1}^{n} k^c.$$

```
#include <stdio.h>
int main() // temp27.c
{
    int n, sum = 0, m;
    printf("Enter a +ve integer: ");
    scanf("%d", &n);
    m = n ; // save the value
    while(n > 0) {
        sum = n*n*n*n + sum ;
        --n ;
    }
    printf("1^4 + ... + %d^4 = %d\n", m, sum);
    return 0;
}
```

Example IV

Write a program to compute the following sum:

$$S_n = 1^c + 2^c + 3^c + \cdots + n^c,$$

where n and c are input.

We use nested `while`-loops to solve the problem.

```
#include <stdio.h>
int main() // temp28.c
{
    int n, c, sum = 0, m;
    printf("Enter the number of terms: ");
    scanf("%d", &n);
    printf("Enter the power: ");
    scanf("%d", &c);
    m = n ; // save the value
    while(n > 0) {   // outer while
        int i=0, p=1; // local to the block
        while(i++ < c) p *= n; // inner while
        sum += p;
        --n ;
    }
    printf("1^%d + ... + %d^%d = %d\n",
           c, m, c, sum);
    return 0;
}
```

Note: n^c can be computed in a better using the representation of c.