CS110, Computer Programming Lab, Department of CSE, IIT Guwahati Spring' 18

Module 2, Stage 3: Looping Constructs

Reference class slides: 23, 24, 30, 31 Jan

Reference chapters from some text books:

Let us C, Yashvant Kanetkar (Chapter 3)

Programming in C, Stephen Kochan (Chapter 5)

Goals of this stage:

- Use of three looping constructs in C
    - while
    - do while
    - for

Pre-requisites:

- Flow charts
- Variable declaration
- Imperative statements
- Conditional constructs

We will NOT use arrays for this stage.

Problem Description:

We will see how looping constructs can be used to generate numbers in Fibonacci series.

https://en.wikipedia.org/wiki/Fibonacci_number

Aim: Generate kth number in the Fibonacci series.

F(1) = 0 F(2) = 1 F(k) = F(k-1) + F(k-2)

Algorithm:

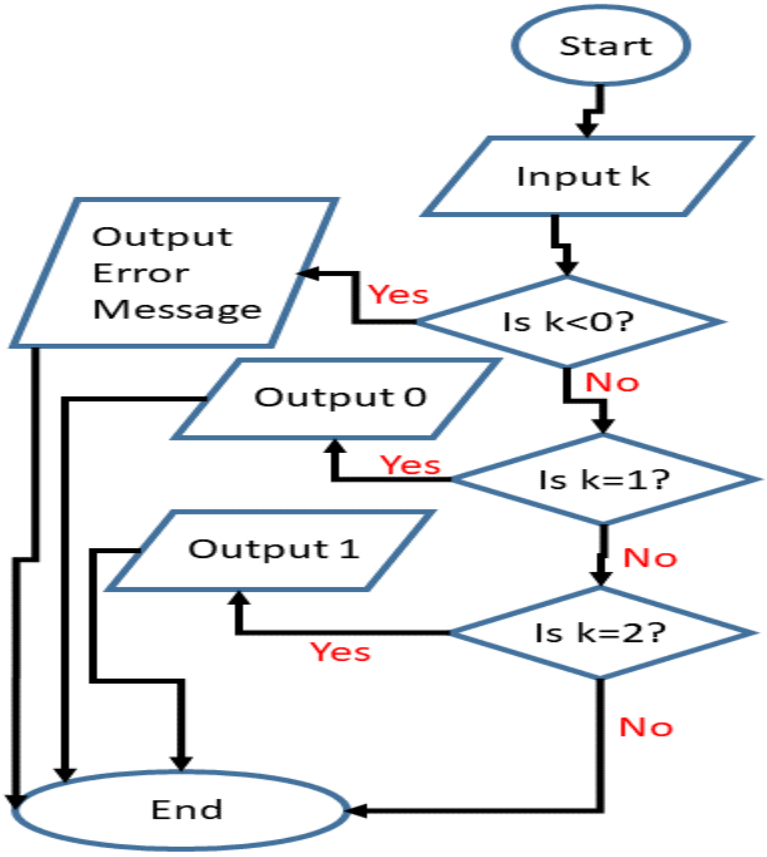
We will develop the algorithm and corresponding code in step-by-step manner

First, let us focus on simple cases to handle.

Value of k should be greater than zero.

Computing first two numbers of the series is trivial, as they are directly defined.

Here is the corresponding flow chart.



The corresponding C code:

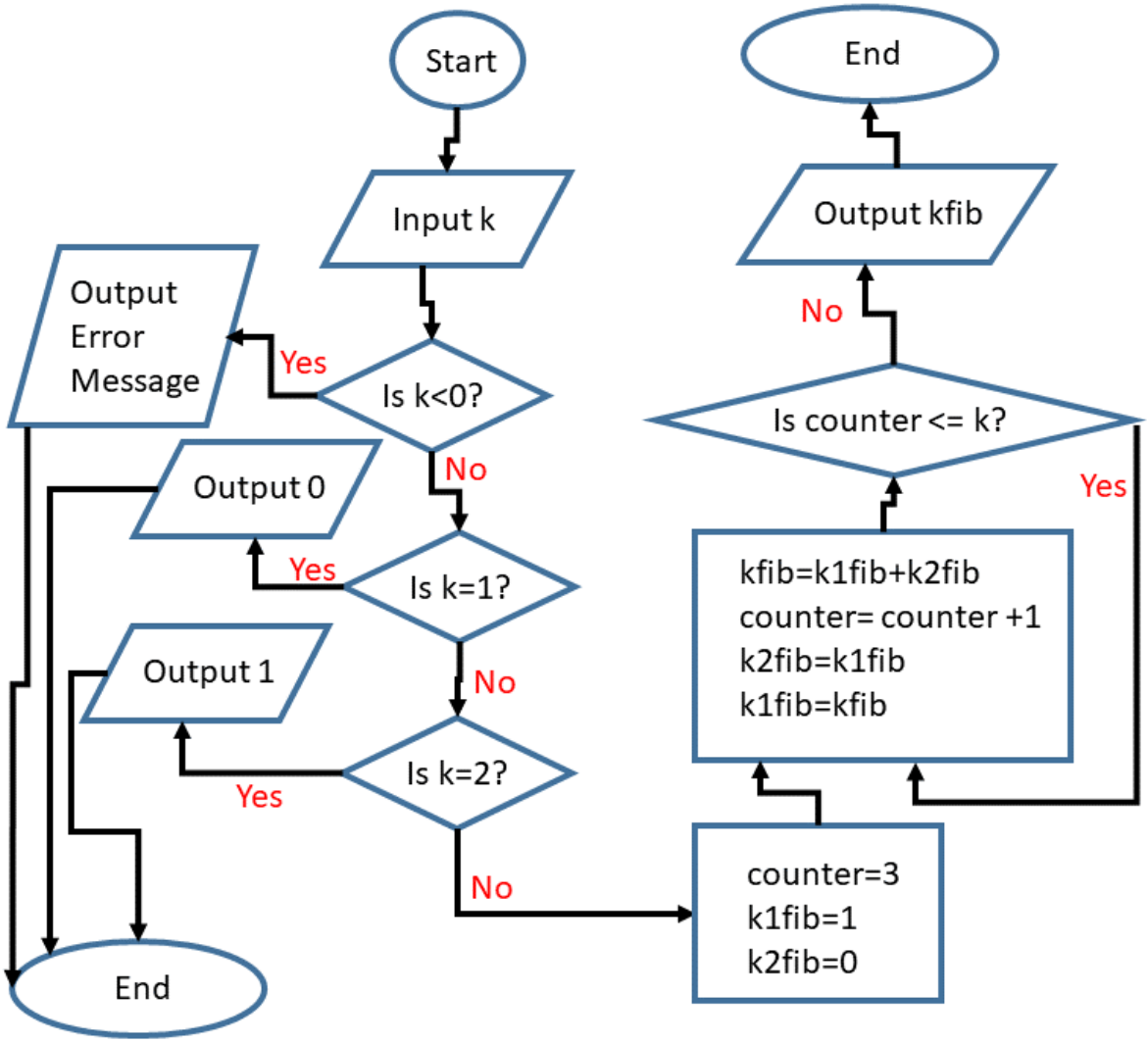
```c
#include < stdio.h > int main() {
    int k;
    printf("Enter value of k");
    scanf("%d", & k);
    if (k < 1) {
        printf("k should be greater than zero\n");
        return (0);
    }
    if (k == 1) {
        printf("F(1) is 0\n");
        return (0);
    }
    if (k == 2) {
        printf("F(2) is 1\n");
        return (0);
    }
    return (0);
}
```

Now we want to handle cases where value of k is greater than 2.

F(3) is computed using sum of F(2) and F(1) and in general F(k) is computed using sum of F(k-1) and F(k-2). Initially, we will compute F(counter) with value of counter as 3. Then we will check if value of counter is same as k. If the answer is yes, then we have computed the required value. Here is the required code using do-while loop.

```c
#include <stdio.h> int main() {
    int k;
    int counter, kfib, k1fib, k2fib;
    printf("Enter value of k");
    scanf("%d", & k);
    if (k < 1) {
        printf("k should be greater than zero\n");
        return (0);
    }
    if (k == 1) {
        printf("F(1) is 0\n");
        return (0);
    }
    if (k == 2) {
        printf("F(2) is 1\n");
        return (0);
    }
    counter = 3; //Indicats which number in the fibonacci series we are going to compute
    k1fib = 1; //F(counter-1)
    k2fib = 0; //F(counter-2)
    do {
        kfib = k1fib + k2fib; //Compute required fibonacci number
        counter++;
        k2fib = k1fib;
        k1fib = kfib; //Get ready to compute next fibonacci number
    } while (counter <= k);
    printf("F(%d) is %d\n", k, kfib);
    return (0);
}
```

And here is the corresponding flow chart.



Now, let us convert the do-while loop to while loop.

Remember, that do-while loop ensures at least one execution of the loop body. Whereas the while loop does not provide any such guarantee.

Here is the corresponding code, using the while loop.

```
#include < stdio.h > int main() {
    int k;
    int counter, kfib, k1fib, k2fib;
    printf("Enter value of k");
    scanf("%d", & k);
    if (k < 1) {
        printf("k should be greater than zero\n");
        return (0);
    }
    if (k == 1) {
        printf("F(1) is 0\n");
        return (0);
    }
    if (k == 2) {
        printf("F(2) is 1\n");
```

```
        return (0);
    }
    counter = 3; //Indicats which number in the fibonacci series we are going to compute
    k1fib = 1; //F(counter-1)
    k2fib = 0; //F(counter-2)
    while (counter <= k) {
        kfib = k1fib + k2fib; //Compute required fibonacci number
        counter++;
        k2fib = k1fib;
        k1fib = kfib; //Get ready to compute next fibonacci number
    }
    printf("F(%d) is %d\n", k, kfib);
    return (0);
}
```

We can convert the while loop to for loop by asking three questions

What are the initializations required

- Initialize variables counter, k1fib, k2fib

What is the looping condition

- Value of counter should be less than or equal to k

What are the book keeping operations performed in each iteration of the loop

- Increment value of counter

```
# include < stdio.h > int main() {
    int k;
    int counter, kfib, k1fib, k2fib;
    printf("Enter value of k");
    scanf("%d", & k);
    if (k < 1) {
        printf("k should be greater than zero\n");
        return (0);
    }
    if (k == 1) {
        printf("F(1) is 0\n");
        return (0);
    }
    if (k == 2) {
        printf("F(2) is 1\n");
        return (0);
    }
    counter = 3; //Indicats which number in the fibonacci series we are going to compute
    k1fib = 1; //F(counter-1)
    k2fib = 0; //F(counter-2)
    for (counter = 3, k1fib = 1, k2fib = 0; counter <= k; counter++) {
        kfib = k1fib + k2fib; //Compute required fibonacci number
        k2fib = k1fib;
        k1fib = kfib; //Get ready to compute next fibonacci number
    }
    printf("F(%d) is %d\n", k, kfib);
    return (0);
}
```

Your code for this stage will be evaluated on the following criteria

- Have you drawn the flowchart on paper for your code?
- Have you given meaningful names to variables?
- Have you added comments at appropriate places in your code?
- Have you indented your code properly?
- Have you prepared multiple test cases on paper to check correctness of your code?
- Is your code working correctly for all those test cases?