SUNG KYUN KWAN UNIVERSITY (SKKU)

Flow Charts

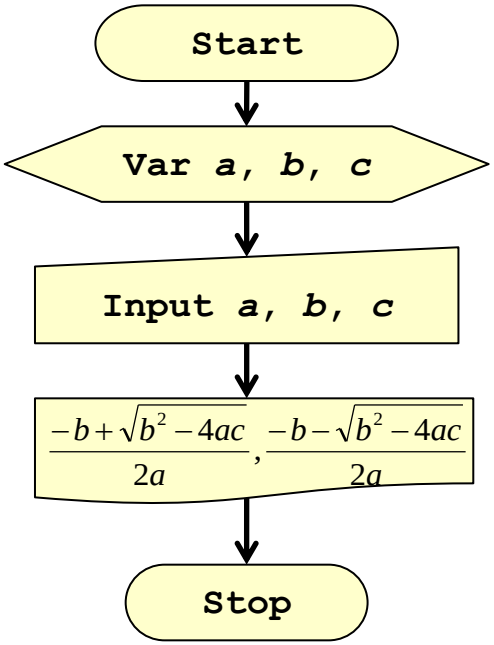
SK UNIVERSITY

Flow Chart

- A *flowchart* is a type of diagram that represents an *algorithm or process*, showing the steps as various symbols, and their order by connecting them with arrows.

- The diagrammatic representation shows a *solution* to a given *problem*.

A Quick Glance at Flow Chart



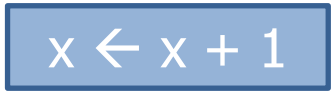
Why Flow Chart?

- Flow charts are useful in analyzing, designing, documenting or managing a process or program.
  - The process is more understandable for humans.
  - Potential errors/mistakes/exceptions are minimized.
  - You can easily organize large problems.

- Well-established flow charts can be easily translated to computer programs (here, to "C" language).

Flow Chart Components

- **Arrows:** 
  - Shows "flow of control"
- **Start and end symbols:** 

  - Represents the start and end of a process
- **Generic processing steps:** 

  - Represents generic computations (e.g., add 1 to x)
- **Prepare conditional (declare variables) :** 

  - Shows operations which have no effect other than preparing a value for a subsequent conditional or decision step
  - Declare variables for the flow chart

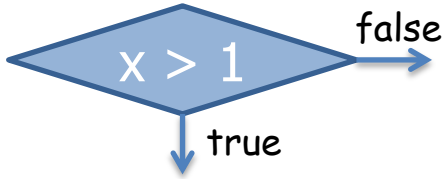
Flow Chart Components

- **Input/Output:** 

  - Data input/output given by a user

- **Document:**  or 

  - Report data so that humans can read them

- **Decision (or Conditional) :** 

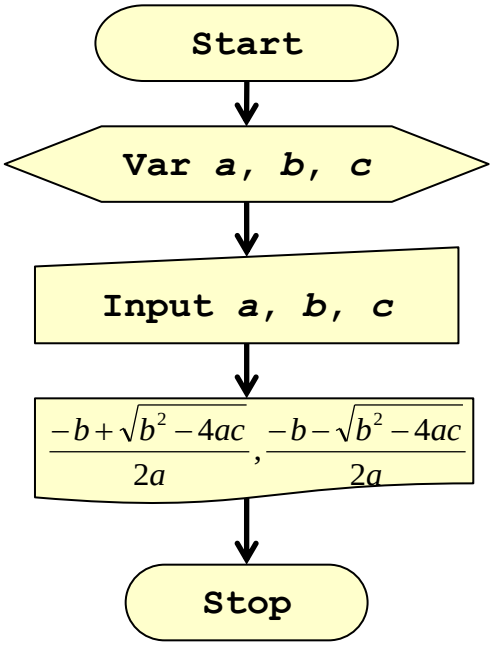
  - Make a decision on which branch the flow goes to.
  - Commonly asks yes/no or true/false question

How to Make Flow Chart

- Construct it from the primary basic elements
- Expand it step by step
- Test whether all the cases are handled
- Test again whether correct outputs are obtained.

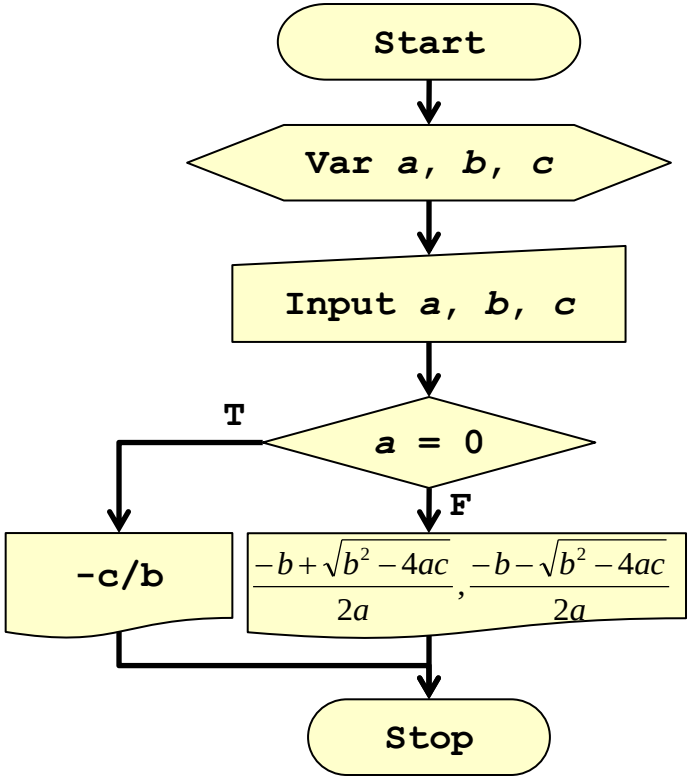
How to Make Flow Chart: Examples

- Given *a*, *b* and *c*, find the solutions of $ax^2+bx+c = 0$



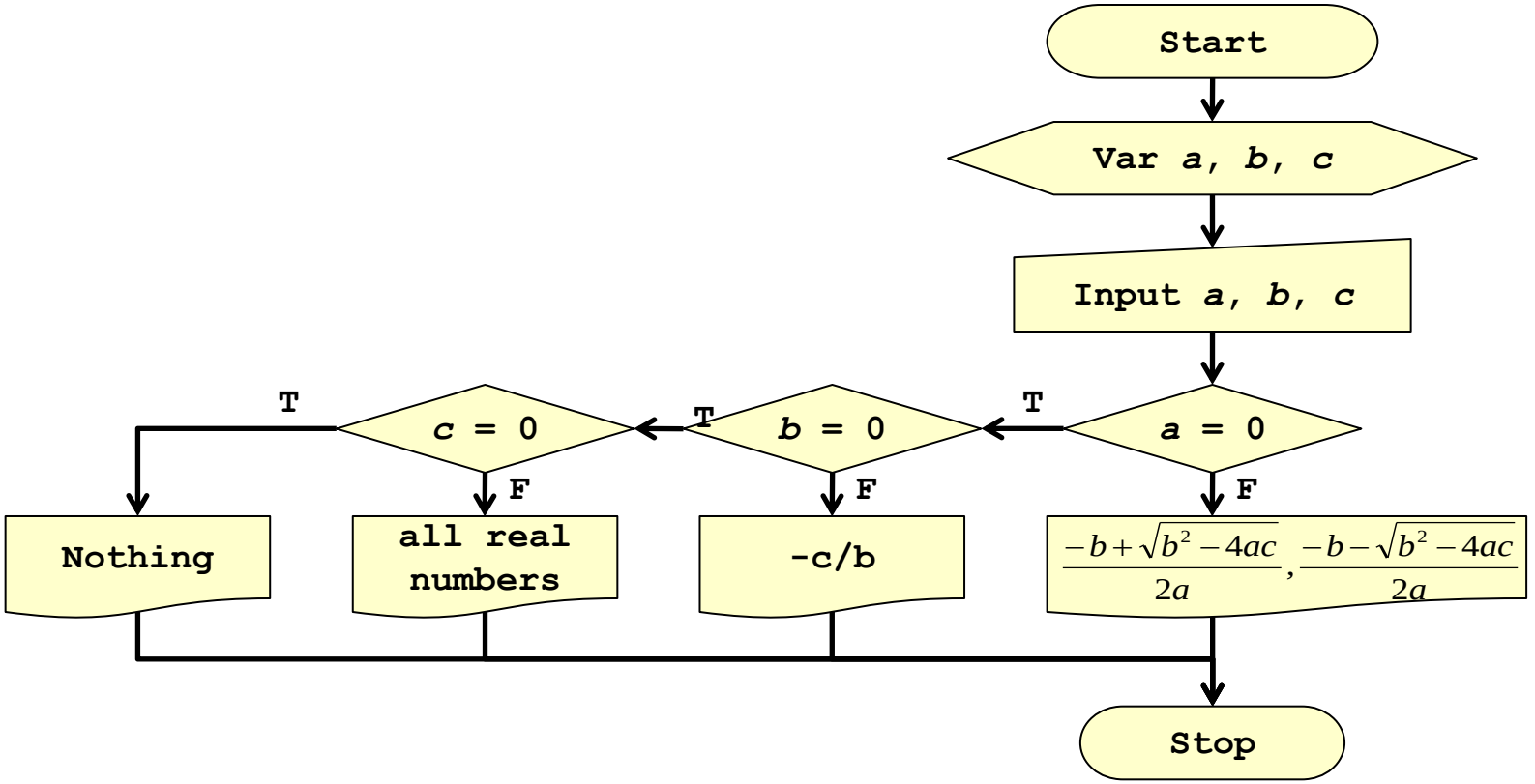
How to Make Flow Chart: Examples

- Given *a*, *b* and *c*, find the solutions of $ax^2+bx+c = 0$



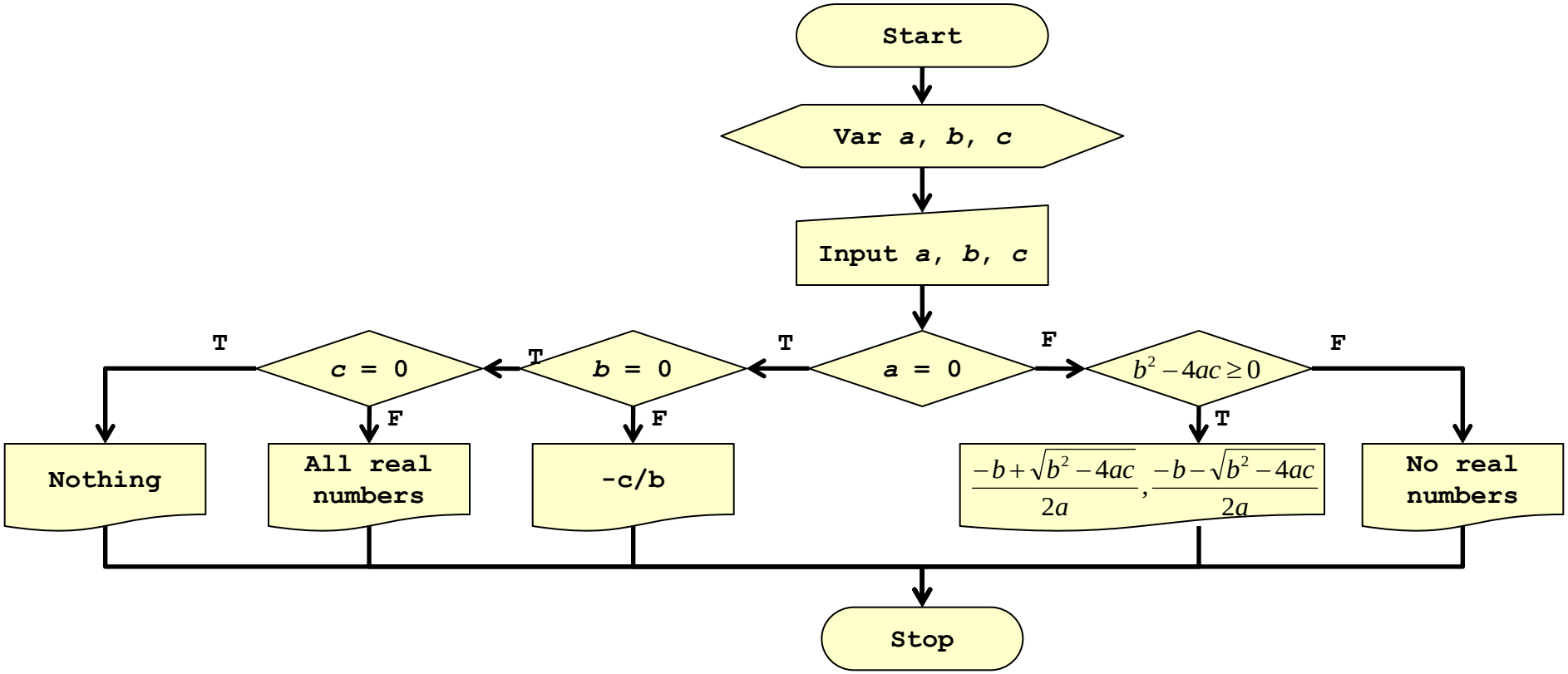
How to Make Flow Chart: Examples

- Given *a*, *b* and *c*, find the solutions of $ax^2+bx+c = 0$



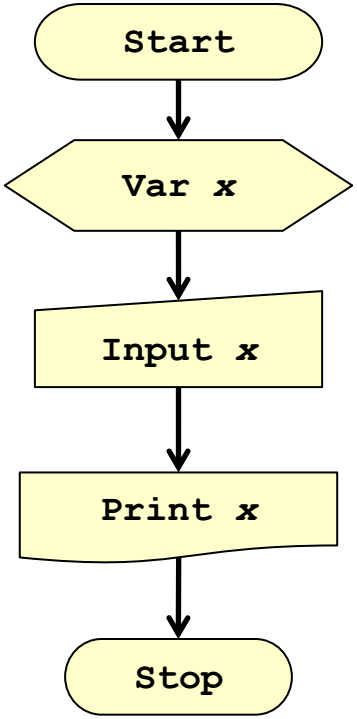
How to Make Flow Chart: Examples

- Given *a*, *b* and *c*, find the solutions of $ax^2+bx+c = 0$



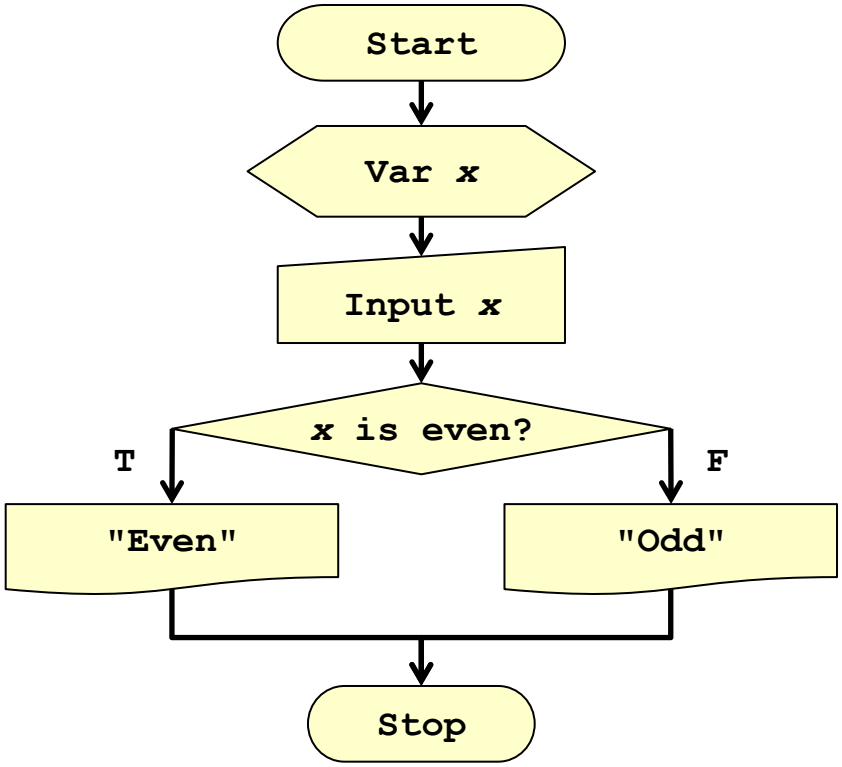
Example 1

- **Read a number and print it**



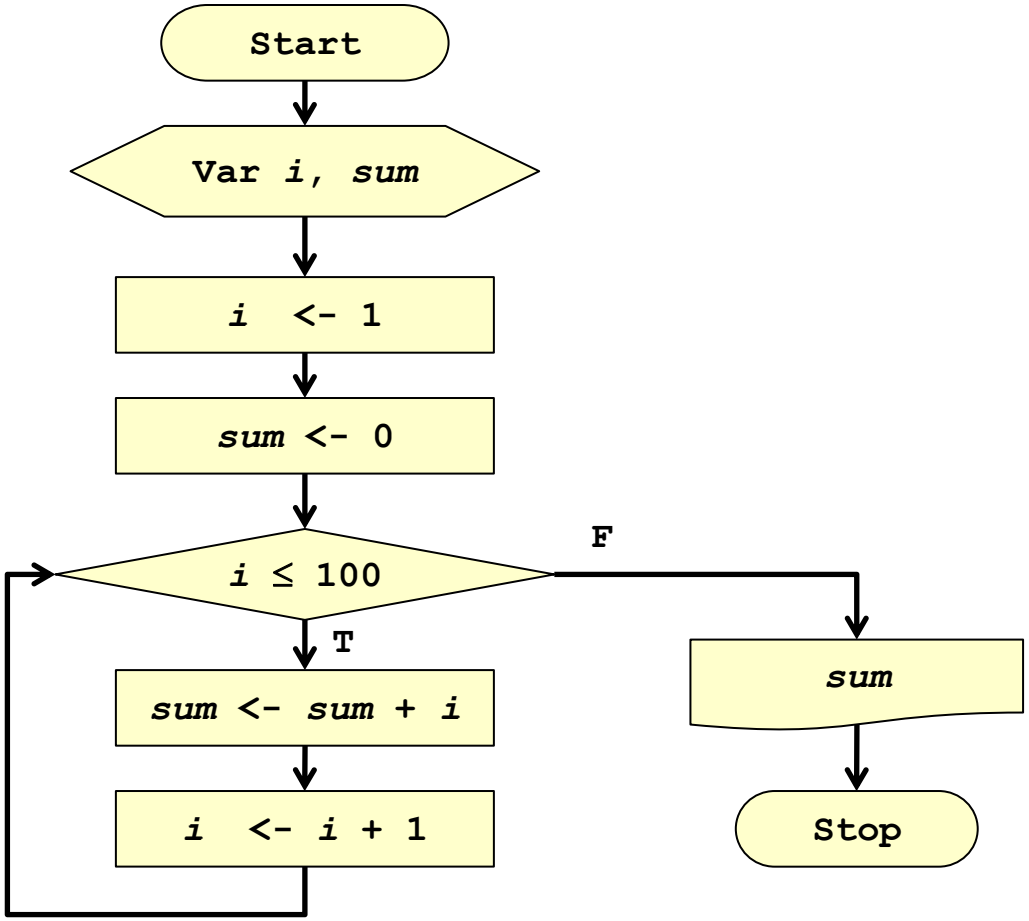
Example 2

- Read a number and print "Even" if it is even or "Odd" if it is odd



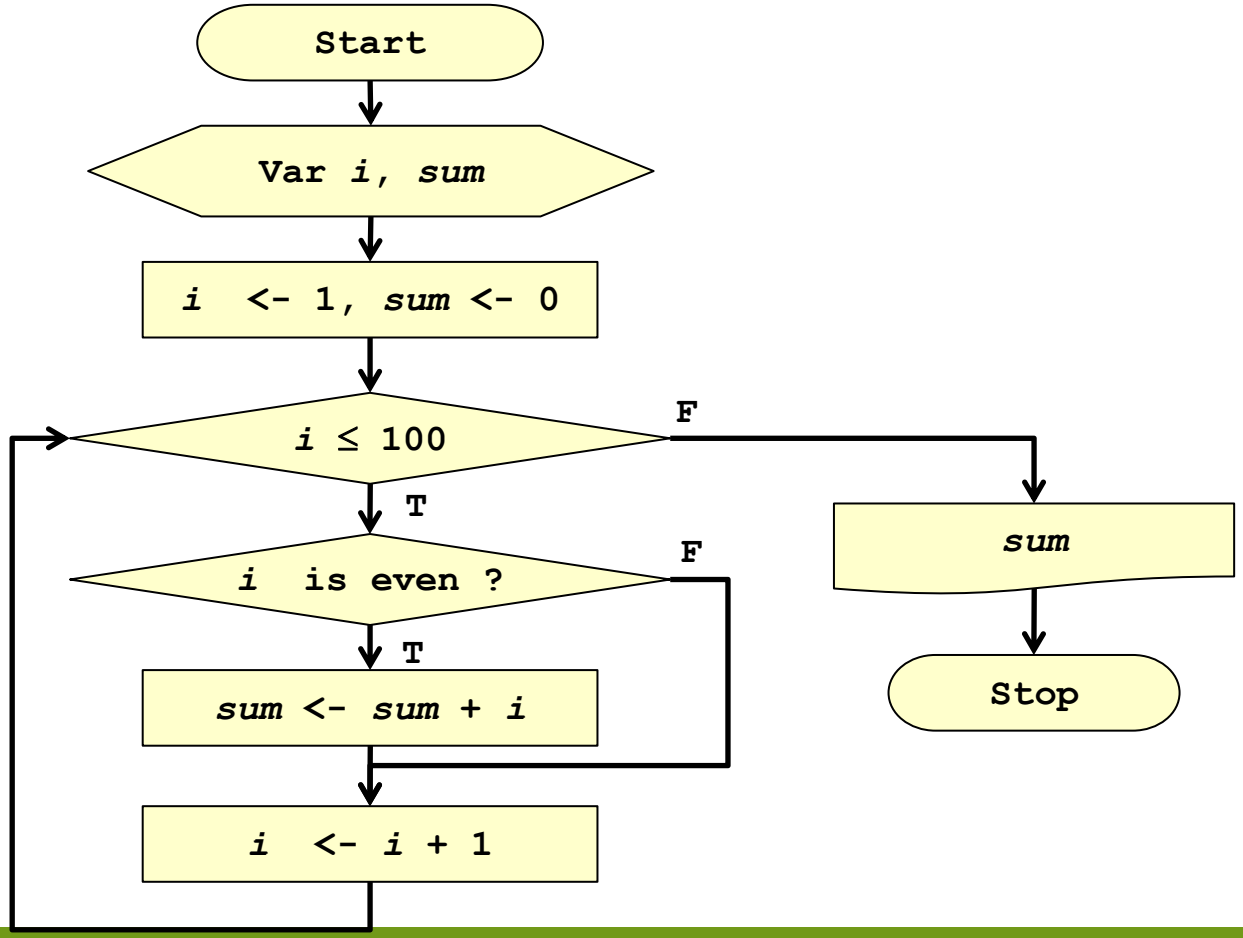
Example 3

- **Add all integers between 1 and 100**



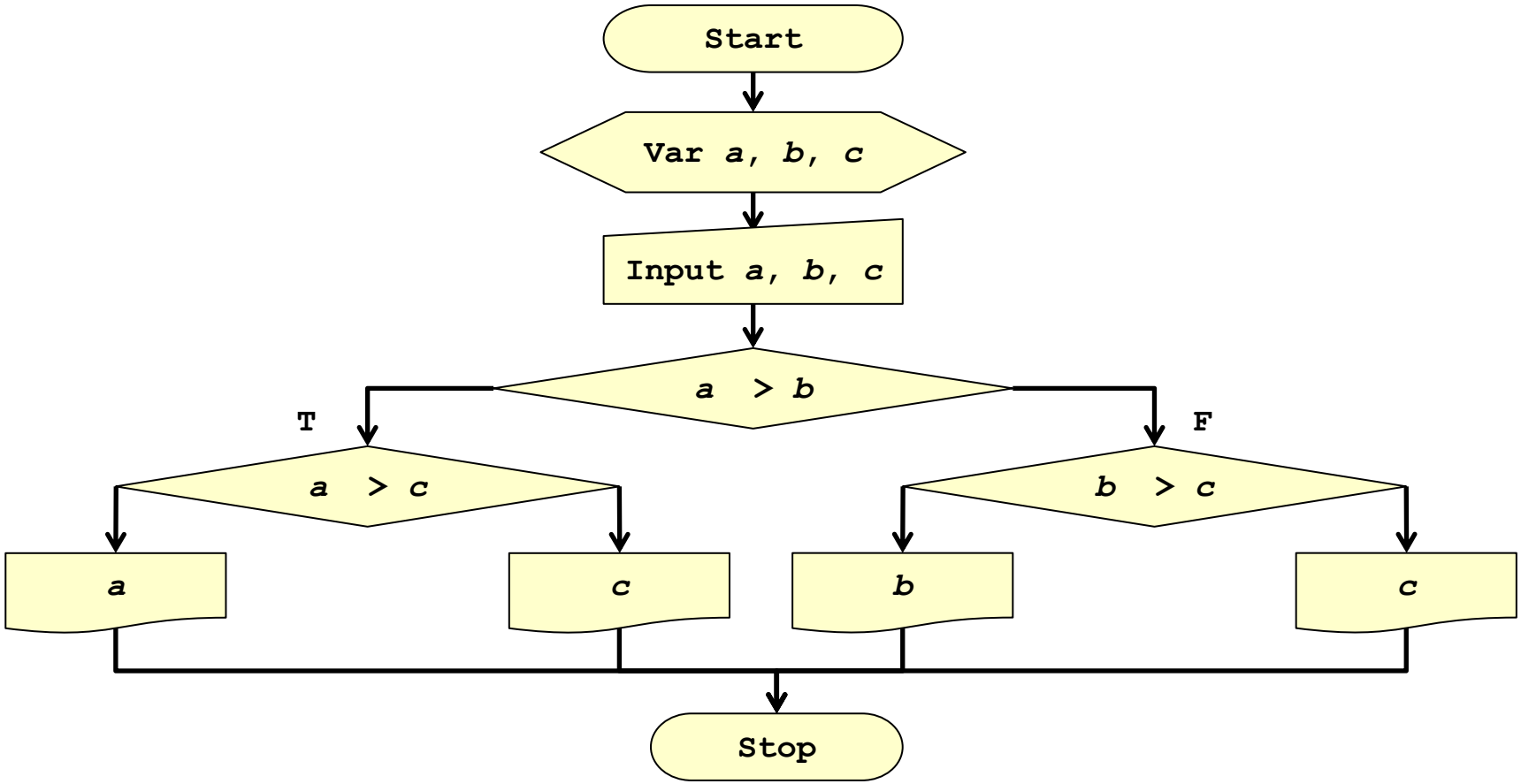
Example 4

- **Add all even integers between 1 and 100**



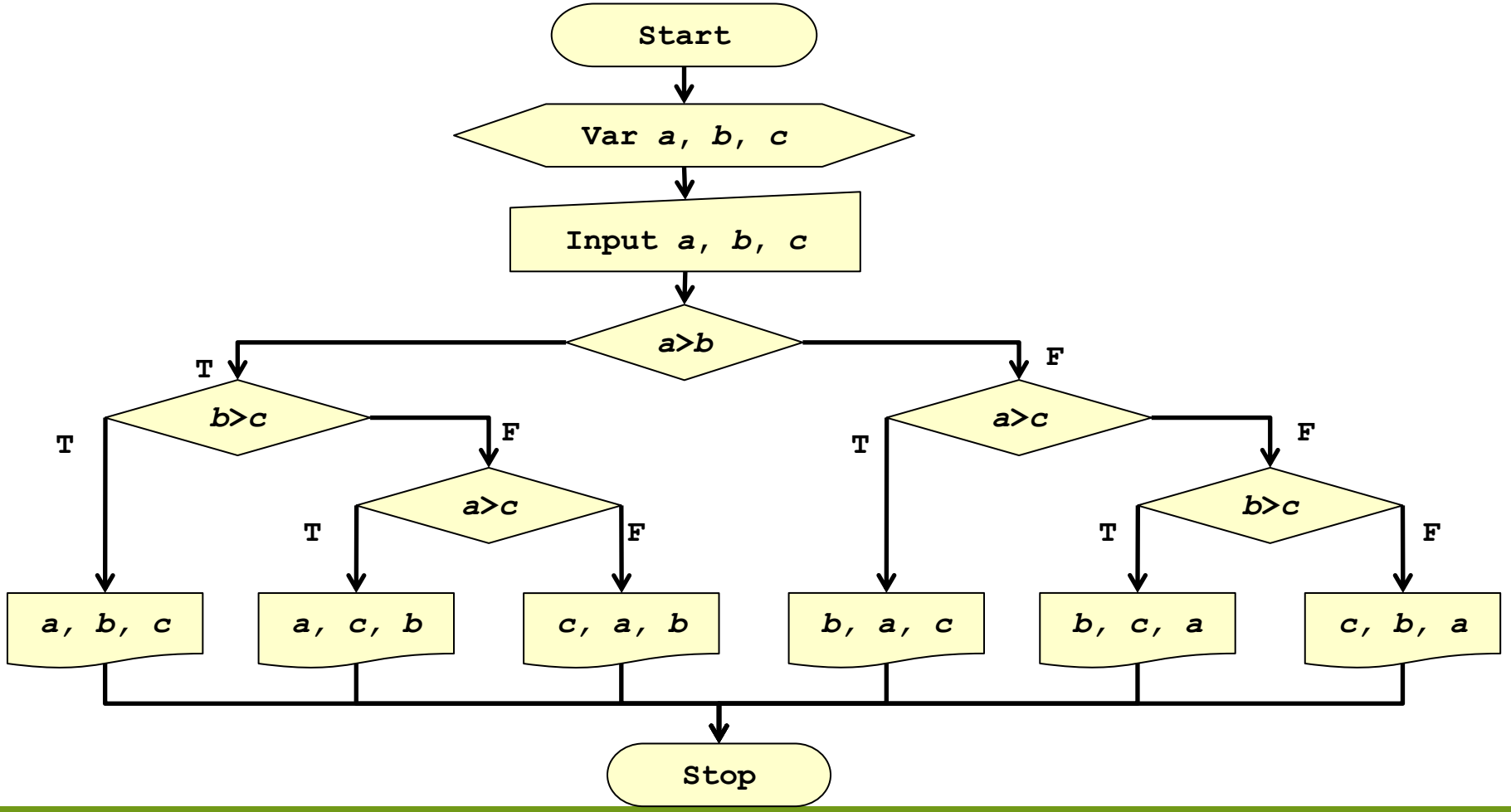
Example 5

- Find the maximum among 3 numbers



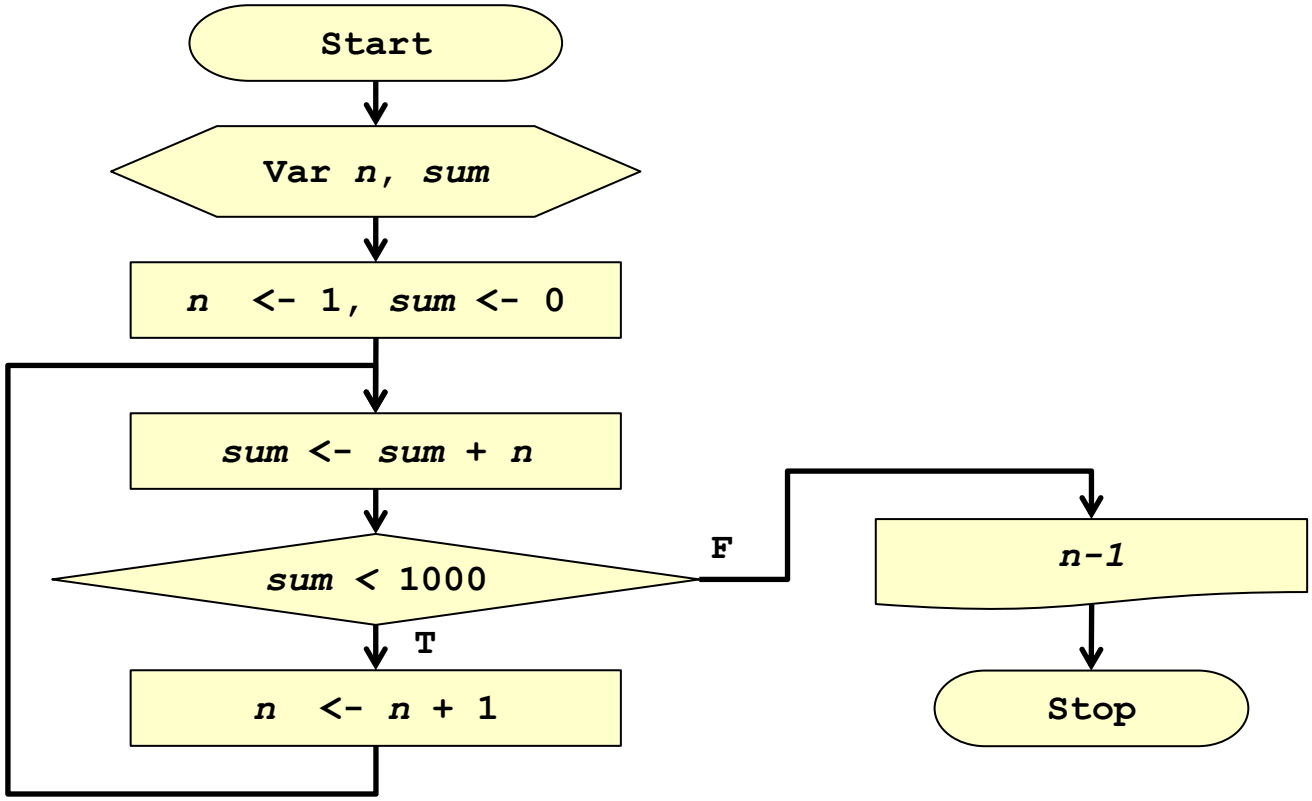
Example 6

- Sort 3 numbers



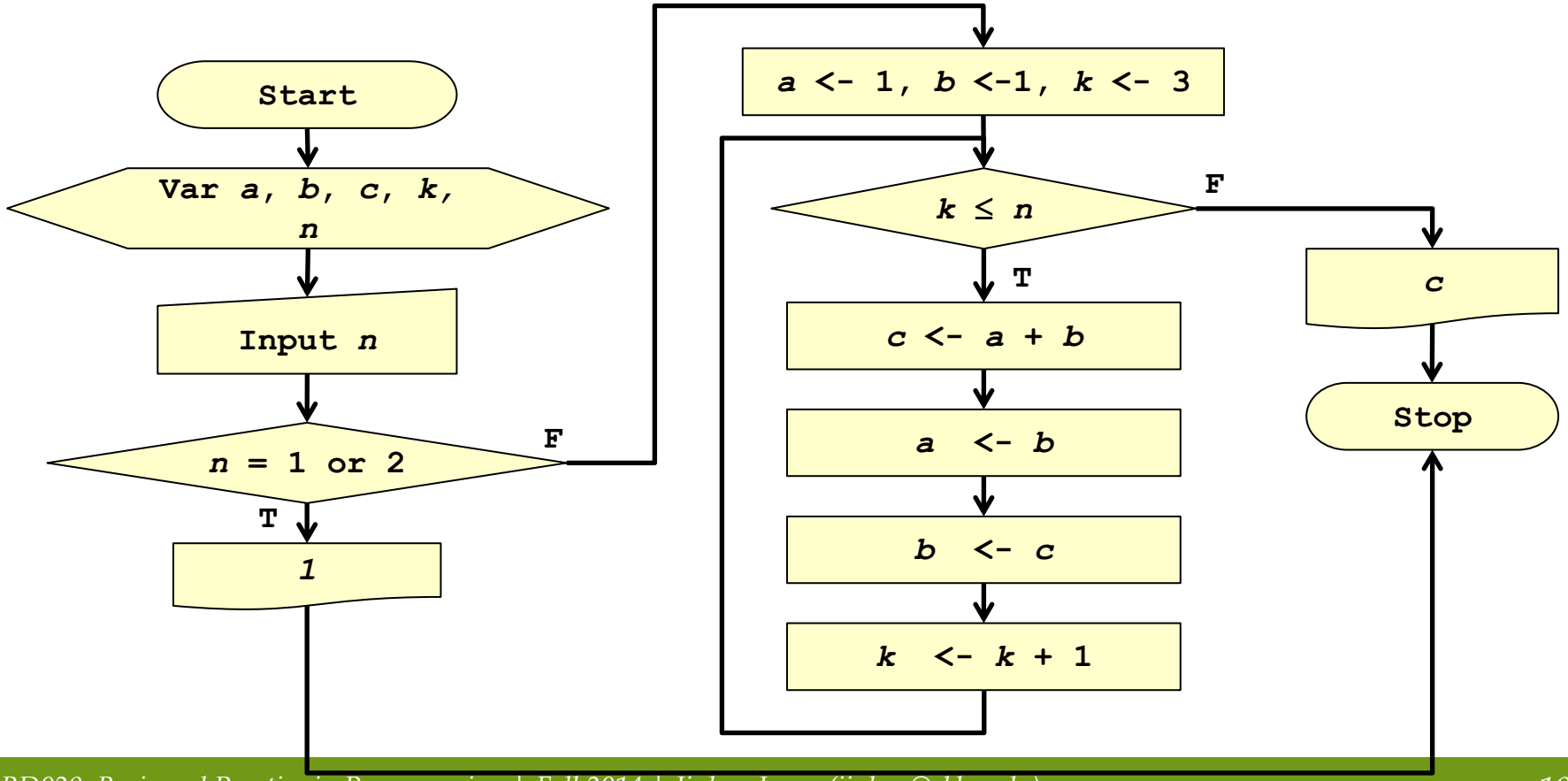
Example 7

- Find the largest *n* such that 1+2+…+*n*<1000



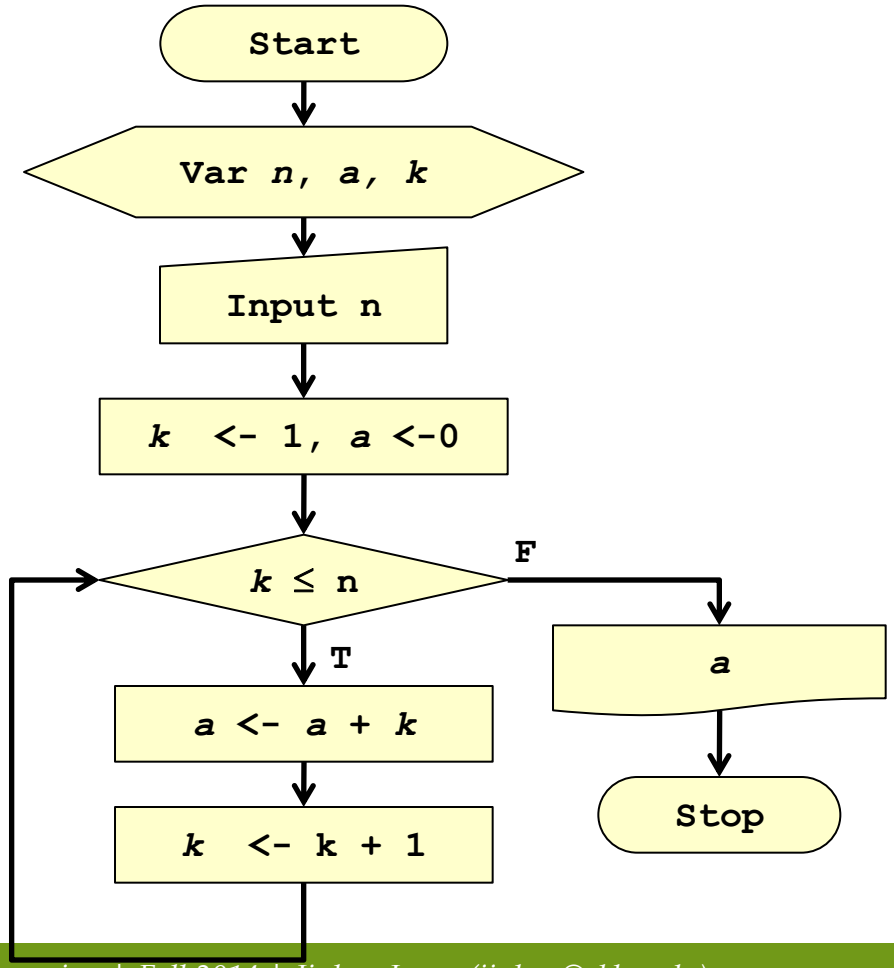
Example 8

- ***n*-th term of Fibonacci sequence**
  - $a_1=1$, $a_2=1$, $a_n=a_{n-1}+a_{n-2}$



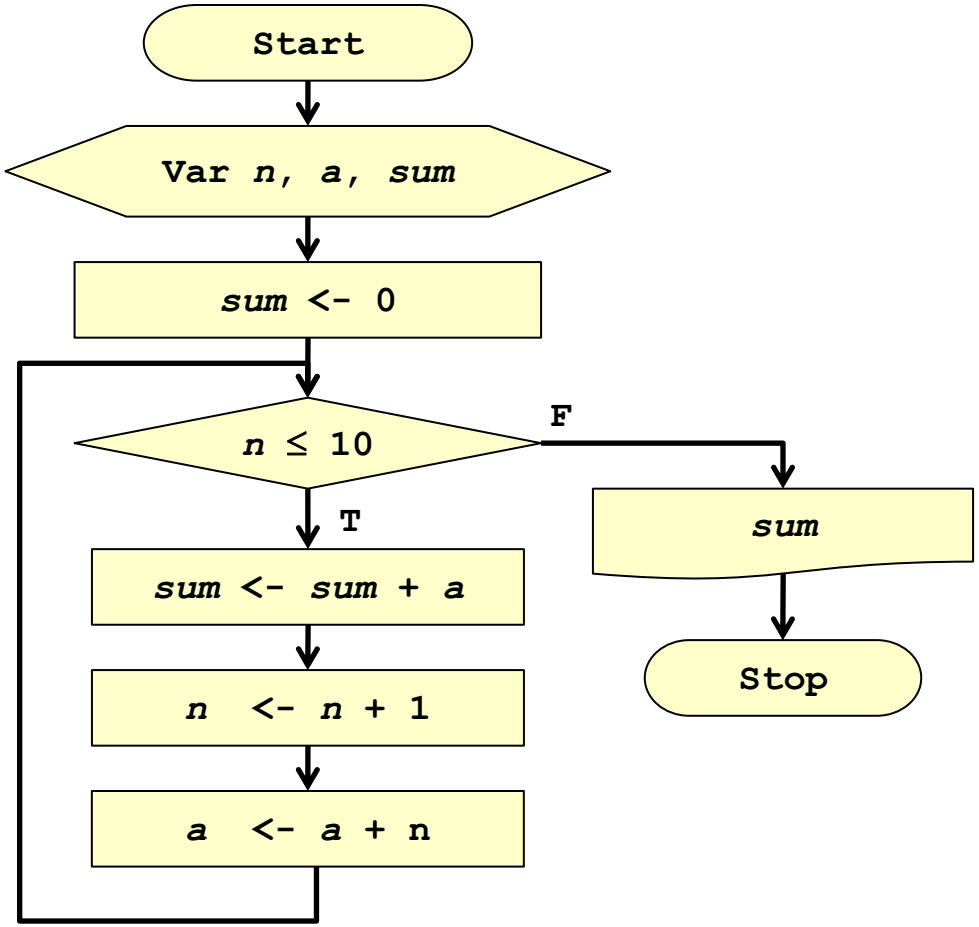
Example 9

- Given n, 1+2+3+....+n



Example 10

- 1+(1+2)+(1+2+3)+.....+(1+....+10)



Flow Chart and C

Basic Structure of C program

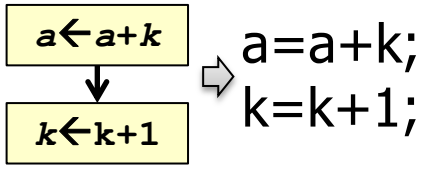
- **In the beginning, there are a couple of #include <..>**
  - Include standard library (routines) for basic functionalities
- **All programs have one main() function which is an entry of each program**
  - **void main(void)**
    {
    statements…
    }
- **Most of the statements should be terminated with ';'**
- **Case-sensitive**
  - E.g., Printf and printf are different

```
#include <stdio.h>

void main()
{
    int x;

    scanf( "%d", &x );
    printf( "%d\n", x*x );

    return;
}
```

statements

1. Conversion of Input/Output

- **Arrows :** ⟶
  - Nothing necessary to the next statement
  - Otherwise, convert it to "`goto label;`"



- **Start/End :** Start Stop
  - Start: no conversion
  - Stop: convert to `return;`

- **Declaration of variables:** Var i, j, k
  - Convert to `int i, j, k;`
  - 'int' is a variable type (abbreviation of 'integer').
  - The variable name is made of alphanumeric characters, but should start with alphabet.

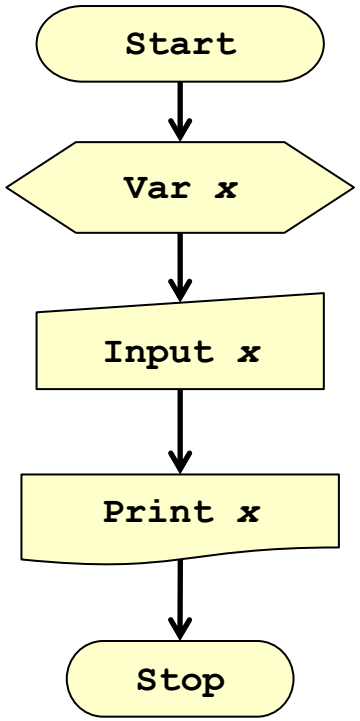
1. Conversion of Input/Output

- **Input:** `Input x` `Input x, y`
  - `scanf( "%d", &x);`
  - `scanf( "%d%d", &x, &y);`

- **Output:** Print x Print x, y
  - Printing numbers:
    - `printf( "%d\n", x);`
    - `printf( "%d %d\n", x, y);`
  - Printing characters: Print "abc" "abc"
    - `printf( "abc\n" );`

Example 1

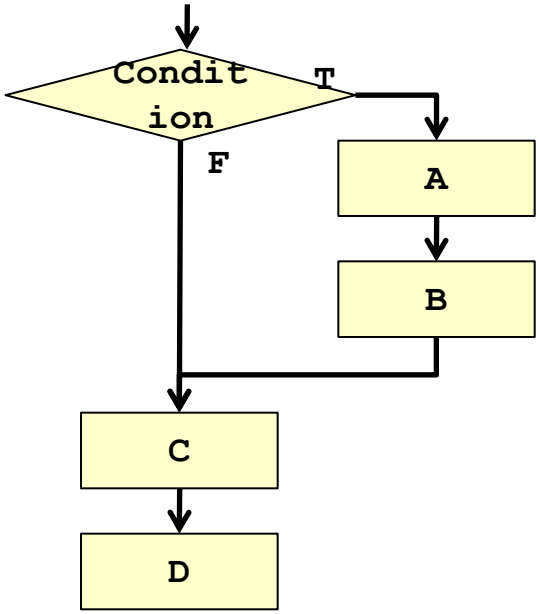
- **Read a number and print it**




```
#include <stdio.h>

main()
{
    int x;

    scanf( "%d", &x );

    printf( "%d\n", x );

    return;
}
```

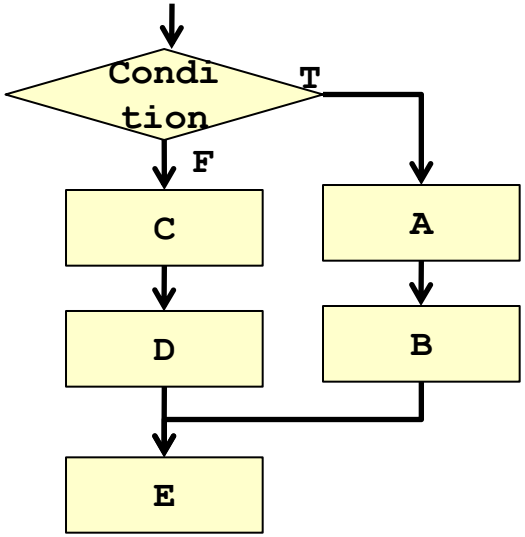
2. Conversion of Conditional (1)

- Type 1




```
if(condition)
{
     A;
     B;
}
C;
D;
```

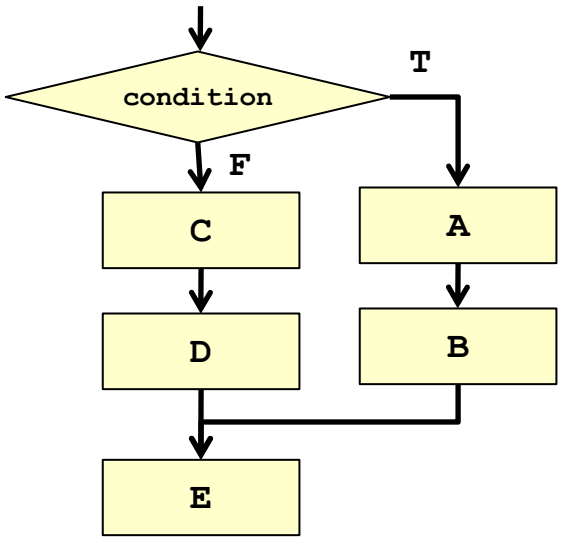
- Type 2




```
if(condition)
{
     A;
     B;
}
else
{
     C;
     D;
}

E;
```

2. Conversion of Conditional (2)

- if statement




```
if( condition )
{
        A;
        B;
}
else
{
        C;
        D;
}
E;
```

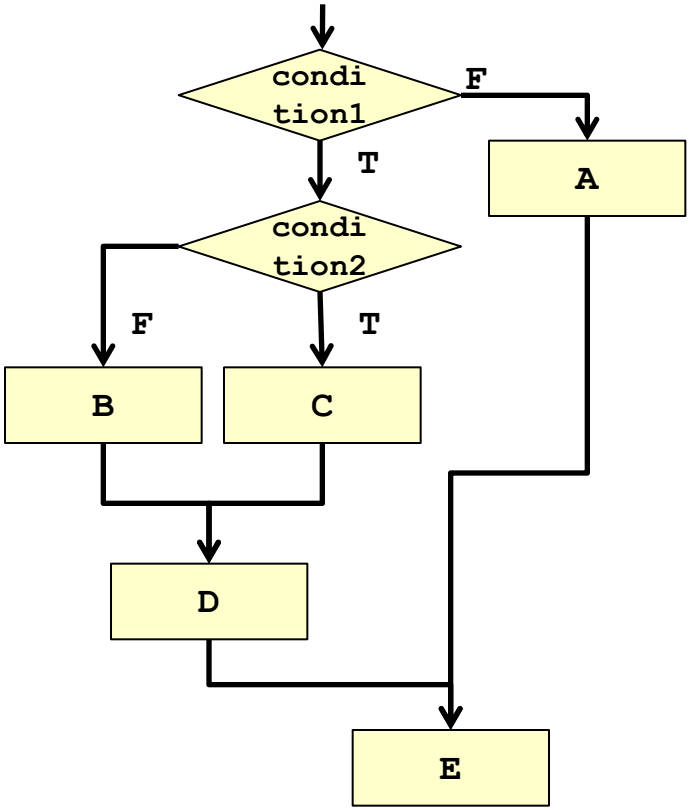
Things to do if the condition is true

Things to do if the condition is false

Things to do after the `if` statement

2. Conversion of Conditional (3)

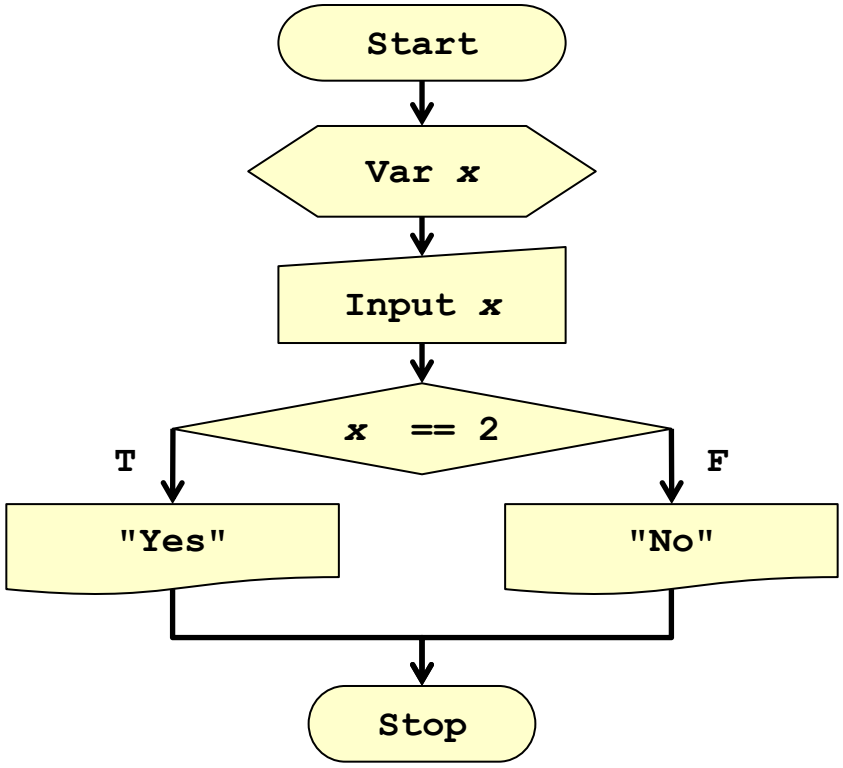
- Example




```
if(condition1 )
{
     if(condition2 )
     {
          C;
      }
      else
      {
           B;
      }
      D;
}
else
{
      A;
}

E;
```

Example 2

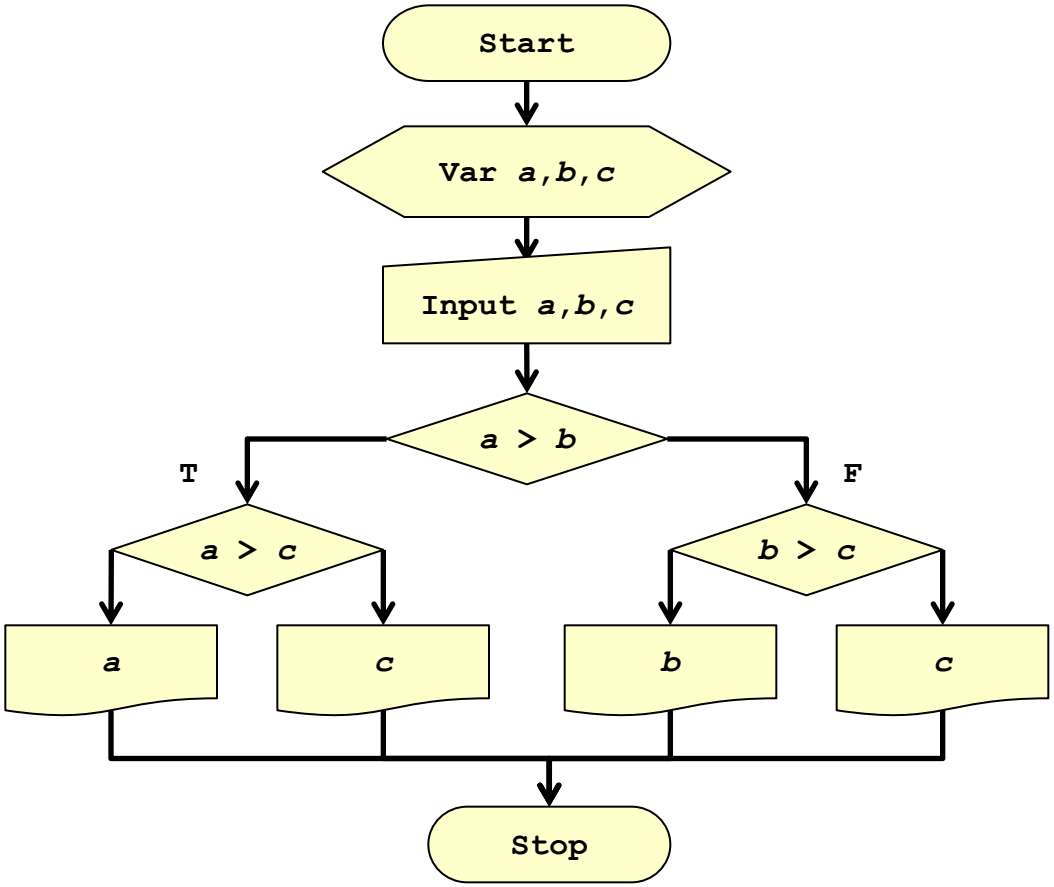
- Read a number and print "Yes" if it is 2 or "No"




```
#include <stdio.h>

main() {

    int x;

    scanf( "%d", &x );

    if( x == 2 )
    {
        printf( "Yes\n" );
    }
    else
    {
        printf( "No\n" );
     }

    return;
}
```

Example 3

- Find the maximum among 3 numbers




```
#include <stdio.h>

main() {

    int a, b, c;

    if(a > b )
    {
        if(a > c ) {
            printf( "%d\n", a );
        } else {
            printf( "%d\n", c );
        }
    }
    else
    {
        if( b > c ) {
            printf( "%d\n", b );
        } else {
            printf( "%d\n", c );
        }
    }

    return;
}
```

2. Conversion of Conditional (4)

- **Comparison symbols:**
  - Comparison operators for conditional statements

Meaning	C Operators	Examples
=	==	a == 2
≠	!=	a != 3
<	<	a < 3
>	>	a > 4
≤	<=	a <= 5
≥	>=	a >= 3
not	!	!(a < 3)
and	&&	(3 < a) && (a < 5)
or	||	(a < 3) || (5 < a)

```
3 < a < 5 ?
```

2. Conversion of Conditional (5)

- **Examples: try on your own**
  - a is greater than 3 and less than 5
  - a is greater than 3 and not less than 5
  - negation of "A is greater than 3 and not less than 5"
  - a is equal to or greater than 3
  - a is less than 3 or greater than 5
  - a is not equal to 3
  - a is not greater than 3

3. Conversion of Processing

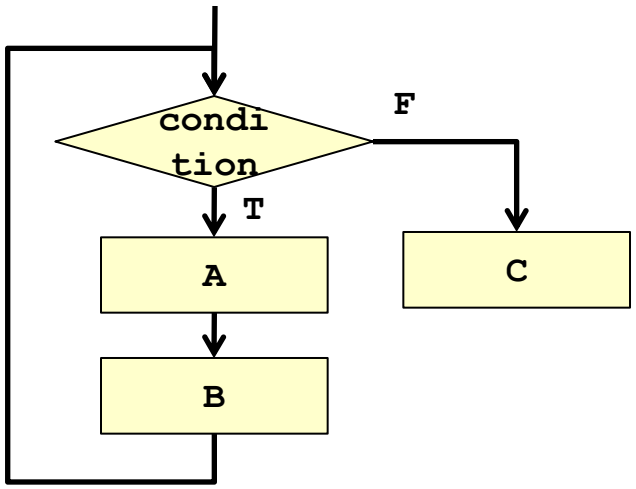
- **Processing symbol :**
  - Processing symbol can contain many statements.
  - Convert '<-' to '=' (assignment operator)
  - a ← a + 1 → a=a+1;

- **Arithmetic operators**

meaning	C operators	examples
+ (addition)	+	a + 2
– (subtraction)	–	a – 3
× (multiplication)	*	a * 3
/ (division)	/	a / 4
Modulo (remainder)	%	a % 5

4. Conversion of Loop

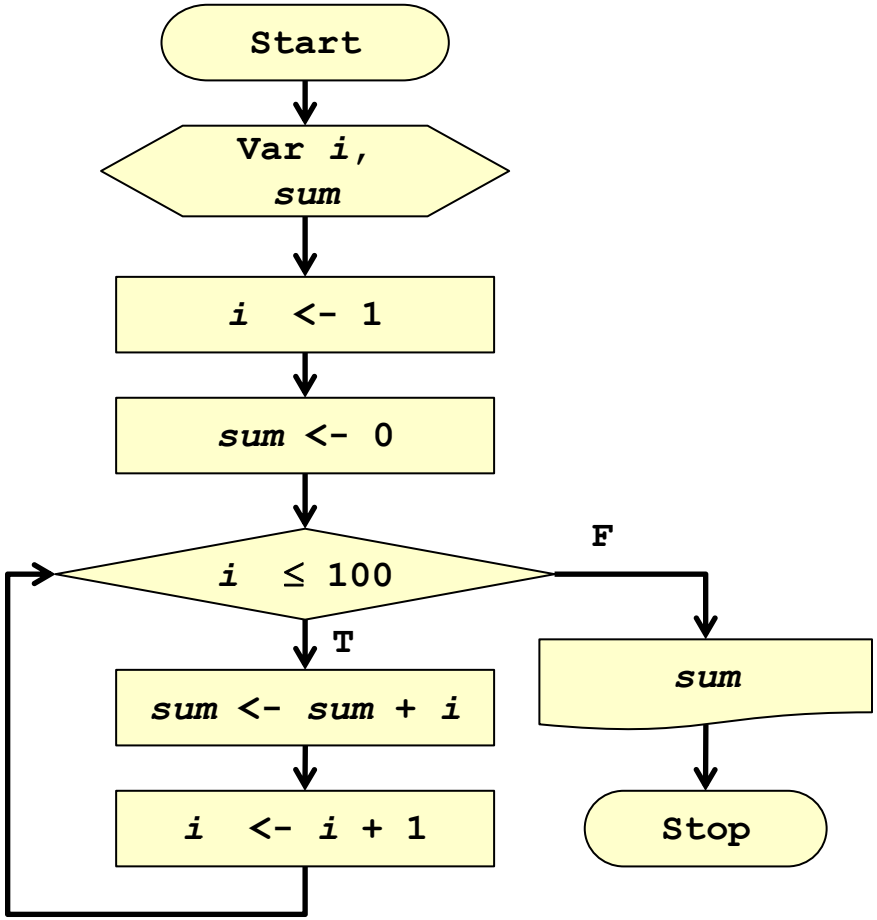
- **Type 3**
  - Repeat the statements while the condition is true




```
while( condition)
    {
        A;
        B;
    }

    C;
```

Example 6

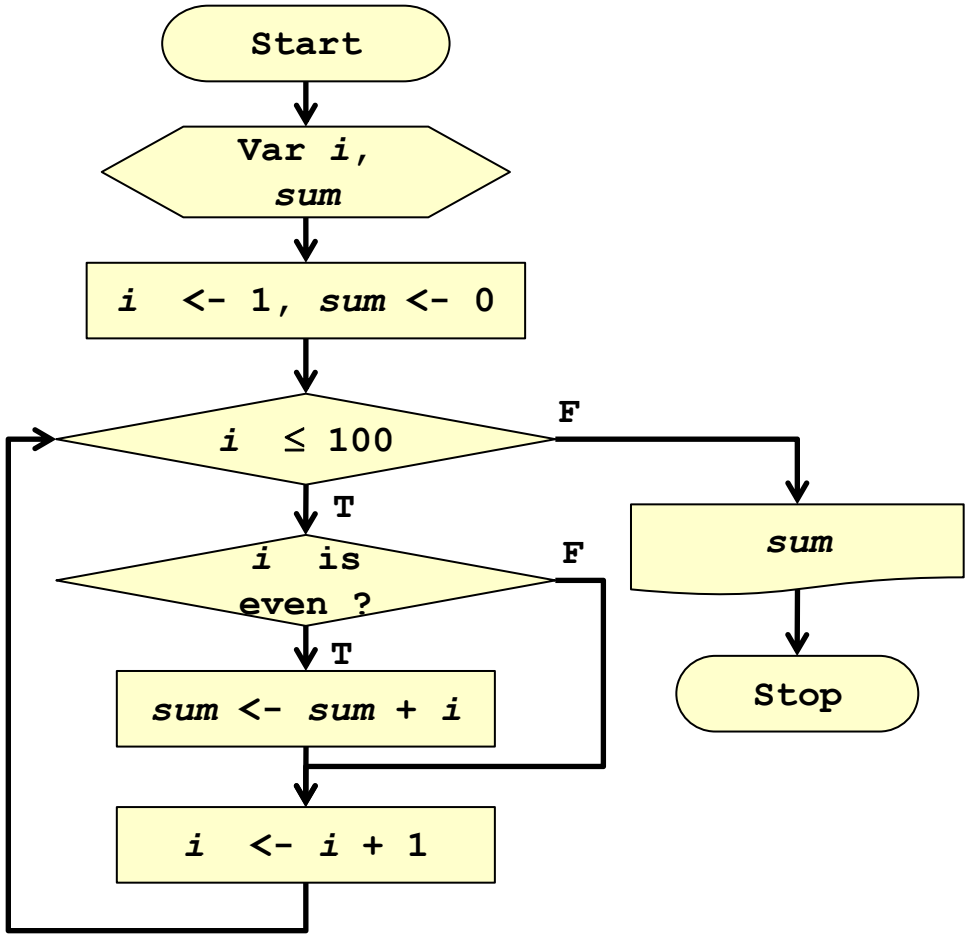
- Add all integers between 1 and 100




```c
#include <stdio.h>
#include <math.h>

main() {

    int i, sum;

    i   = 1;
    sum = 0;

    while( i <= 100 )
    {
          sum = sum + i;
          i = i + 1;
    }

    printf( "%d", sum );

    return;
}
```

Example 7

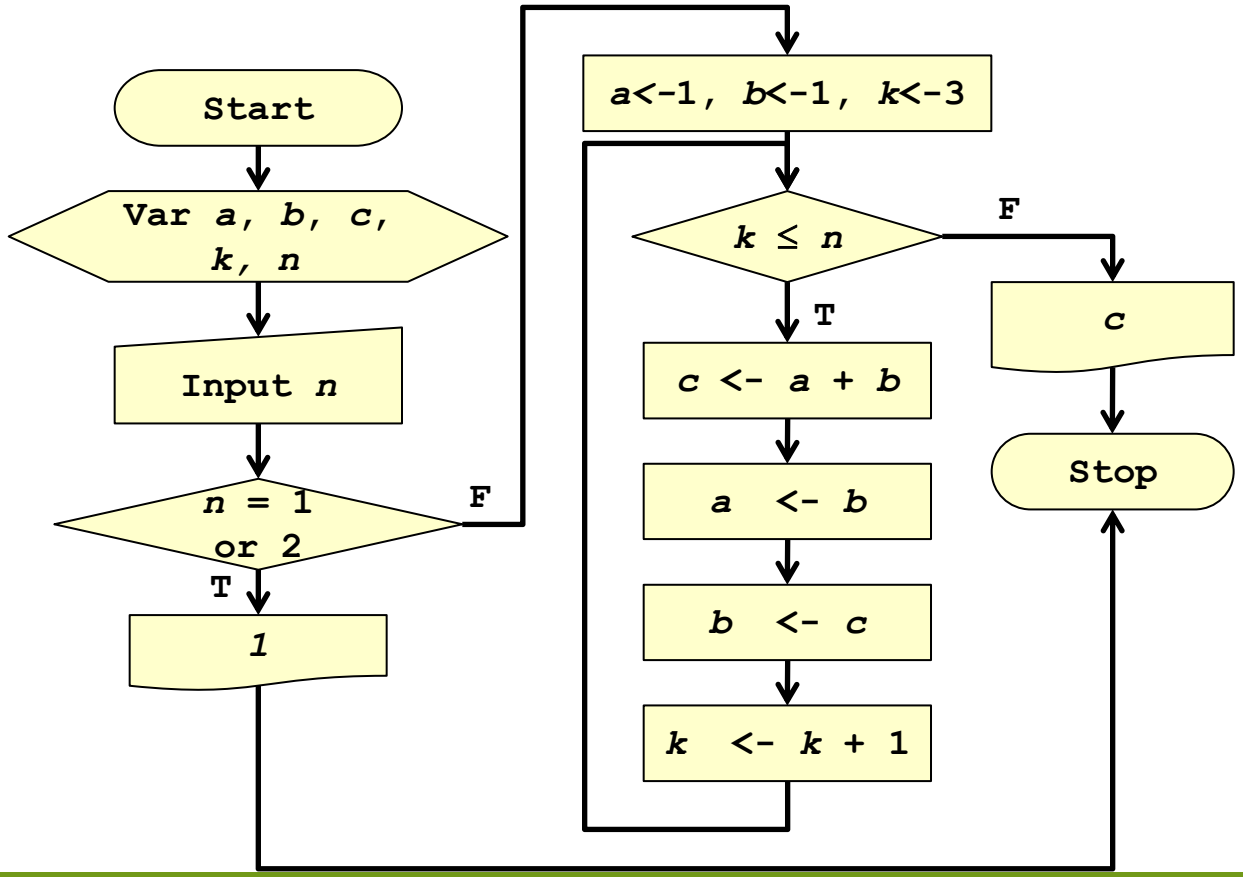
- **Add all even integers between 1 and 100**




```
#include <stdio.h>
#include <math.h>

main() {

    int i, sum;

    i   = 0;
    sum = 0;

    while( i <= 100 )
    {
          if( i % 2 == 0 )
          {
               sum = sum + i;
          }
          i = i + 1;
    }

    printf( "%d", sum );

    return;
}
```

Example 8

- ***n*-th term of Fibonacci sequence**
  - $a_1=1,\ a_2=1,\ a_n=a_{n-1}+a_{n-2}$




```
#include <stdio.h>
#include <math.h>

main()
{
  int a, b, c, k, n;

  scanf( "%d", &n );

  if( (n == 1) || (n == 2) )
  {
     printf( "%d", 1 );
   }
   else
   {
     a = 1;
     b = 1;
     k = 3;

     while( k <= n )
     {
        c = a + b;
        a = b;
        b = c;
        k = k + 1;
     }

      printf( "%d", c );
  }

    return;
}
```

Summary

- **Flowchart**
  - A diagram to represent a process a computer needs to perform to solve a problem
  - Easy to understand for human
  - Easy to be converted to programming languages including C
- **Draw a flowchart before coding a program**
- **Lab introduction**
  - In the room #400212