Computer programming

Lecture 3

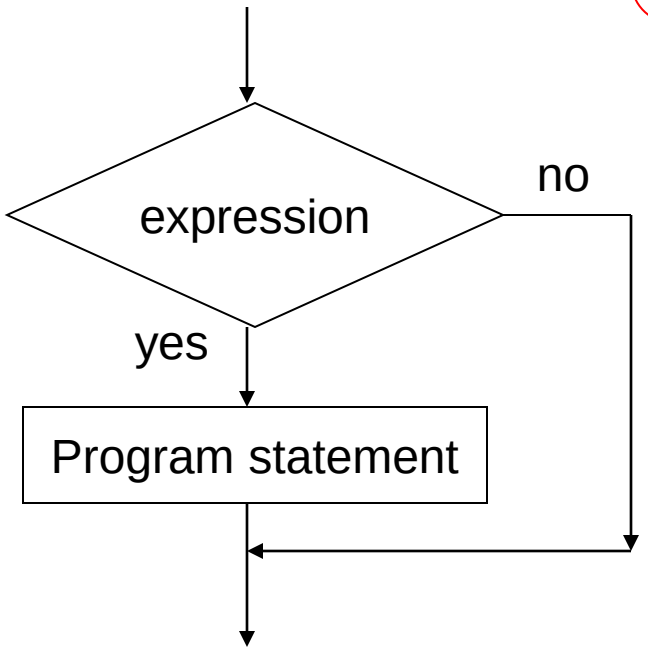
Lecture 3: Outline

- **The if Statement**
- **The if-else Construct**
- **Nested if Statements**
- **The else if Construct**
- **The switch Statement**

The `if` statement

single branch selection structure

if (*expression*)
program statement



If expression is true (non-zero), executes statement.
If gives you the choice of executing statement or skipping it.

Example - if

single branch selection structure

```
#include <stdio.h>
int main (void)
{
    float x,y;
    printf ("Type in your number: ");
    scanf ("%f", &x);
    if ( x < 0 )
        y = 3-x;
    printf ("%f\n", y);
    return 0;
}
```

```
ur number: -8


turned 0 (0x0)   execution time
key to continue.
```

```
ur number: 6666


turned 0 (0x0)   execution time
key to continue.
```

Example - `if`

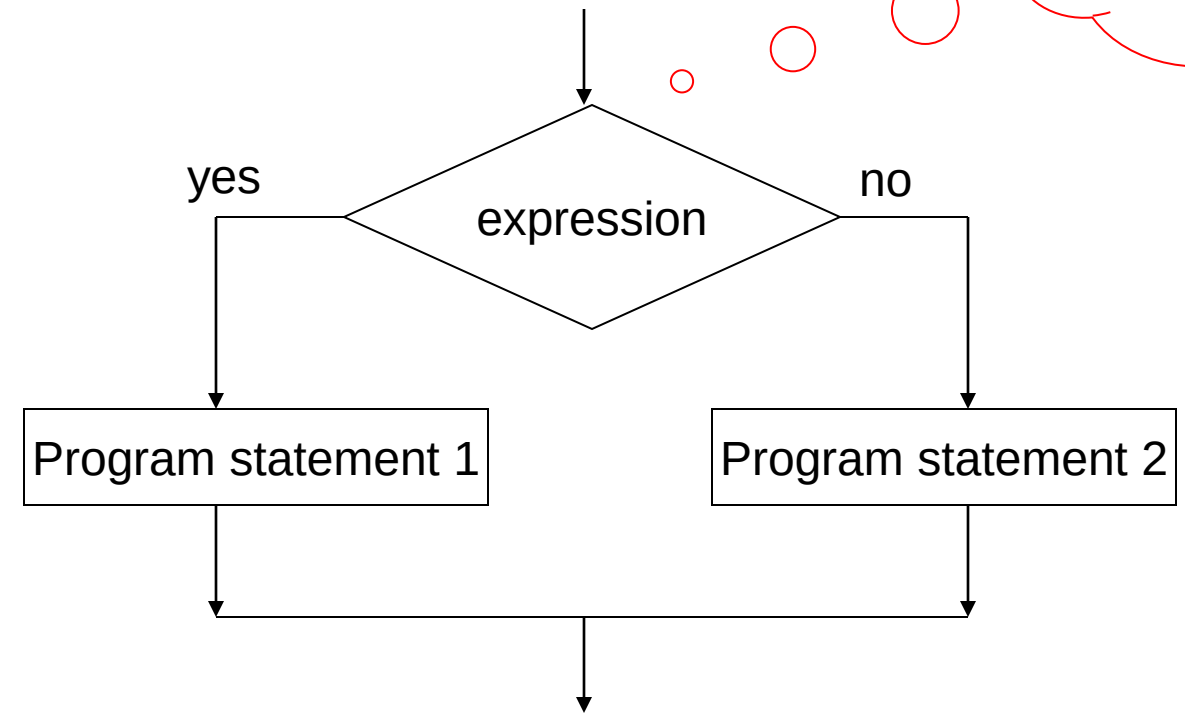
single branch selection structure

```
#include <stdio.h>
int main (void)
{
    float x,y;
    printf ("Type in your number: ");
    scanf ("%f", &x);
    y = x;
    if ( x < 0 )
        y = -x;
    printf ("%f\n", y);
    return 0;
}
```

The `if-else` statement

double branch selection structure

```
if ( expression )
        program statement 1
else
        program statement 2
```



if-else statement:
enables you to
choose between
two statements

The `if-else` statement

double branch selection structure

```
#include "stdio.h"
main()
{ float x , y;
   scanf("%f", &x);
   if(x>=0)
      y = 2 * x ;
   else
      y = 3 - x ;
   printf("y= %.2f\n", y );
}
```

$$y=\begin{cases}3-x & (x<0)\\ 2x & (x\geqslant 0)\end{cases}$$

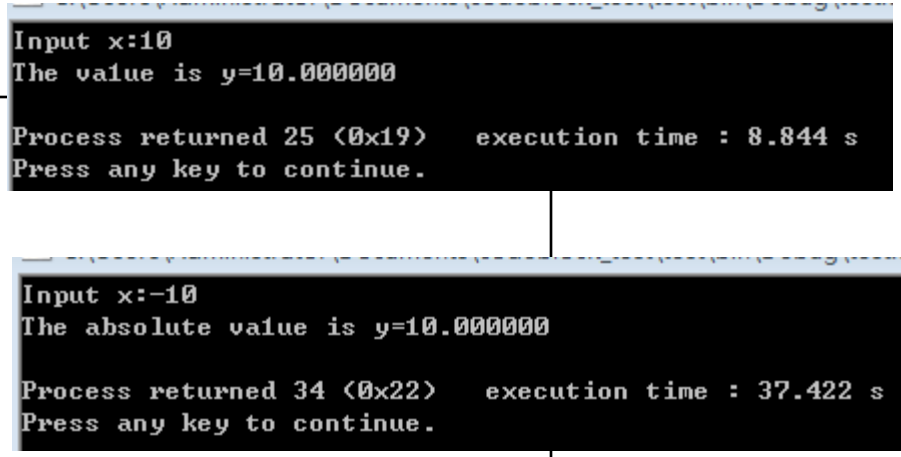
```
8
y= 16.00

Process returned 0 (0x0)   execution time : 10.969 s
Press any key to continue.
```

```
-8
y= 11.00

Process returned 0 (0x0)   execution time : 3.771 s
Press any key to continue.
```

The `if-else` statement

double branch selection structure

```
#include <stdio.h>
void main()
{  float x,y;
   printf("Input x:");
   scanf("%f",&x);
   if(x>=0)
      {y=x;
         printf("The va1ue is y=%f\n",y); }
   else
      {y=-x;
         printf("The absolute va1ue is y=%f\n",y);}
}
```



write a program: input an integer and output whether the integer is odd or even.

Example: `if-else`

```
// Program to determine if a number is even or odd
#include <stdio.h>
int main ()
{
    int number_to_test, remainder;
    printf ("Enter your number to be tested: ");
    scanf ("%i", &number_to_test);
    remainder = number_to_test % 2;
    if ( remainder == 0 )
        printf ("The number is even.\n");
    else
        printf ("The number is odd.\n");
    return 0;
}
```

Attention on `if-else` syntax !

if (*expression*)
    program statement 1
else
    program statement 2

In C, the **;** is part (end) of a statement ! You have to put it also before an `else` !

```
if ( remainder == 0 )
    printf ("The number is even.\n");
else
    printf ("The number is odd.\n");
```

```
if ( x == 0 );
    printf ("The number is zero.\n");
```

Syntactically OK (void statement on if) but probably a semantic error !

write a program: input an year and output whether the year is leap year.

leap year :one year in every four years when February has 29 days instead of 28

The condition for a certain year to be a leap year is that it can be divisible by 4 but not by 100, or by 400.

Example: compound relational test

```
// Program to determine if a year is a leap year
#include <stdio.h>
int main (void)
{
    int year, rem_4, rem_100, rem_400;
    printf ("Enter the year to be tested: ");
    scanf ("%d", &year);
    rem_4 = year % 4;
    rem_100 = year % 100;
    rem_400 = year % 400;
    if ( (rem_4 == 0 && rem_100 != 0) || rem_400 == 0 )
        printf ("It's a leap year.\n");
    else
        printf (“It's not a leap year.\n");
    return 0;
}
```

Logical operators

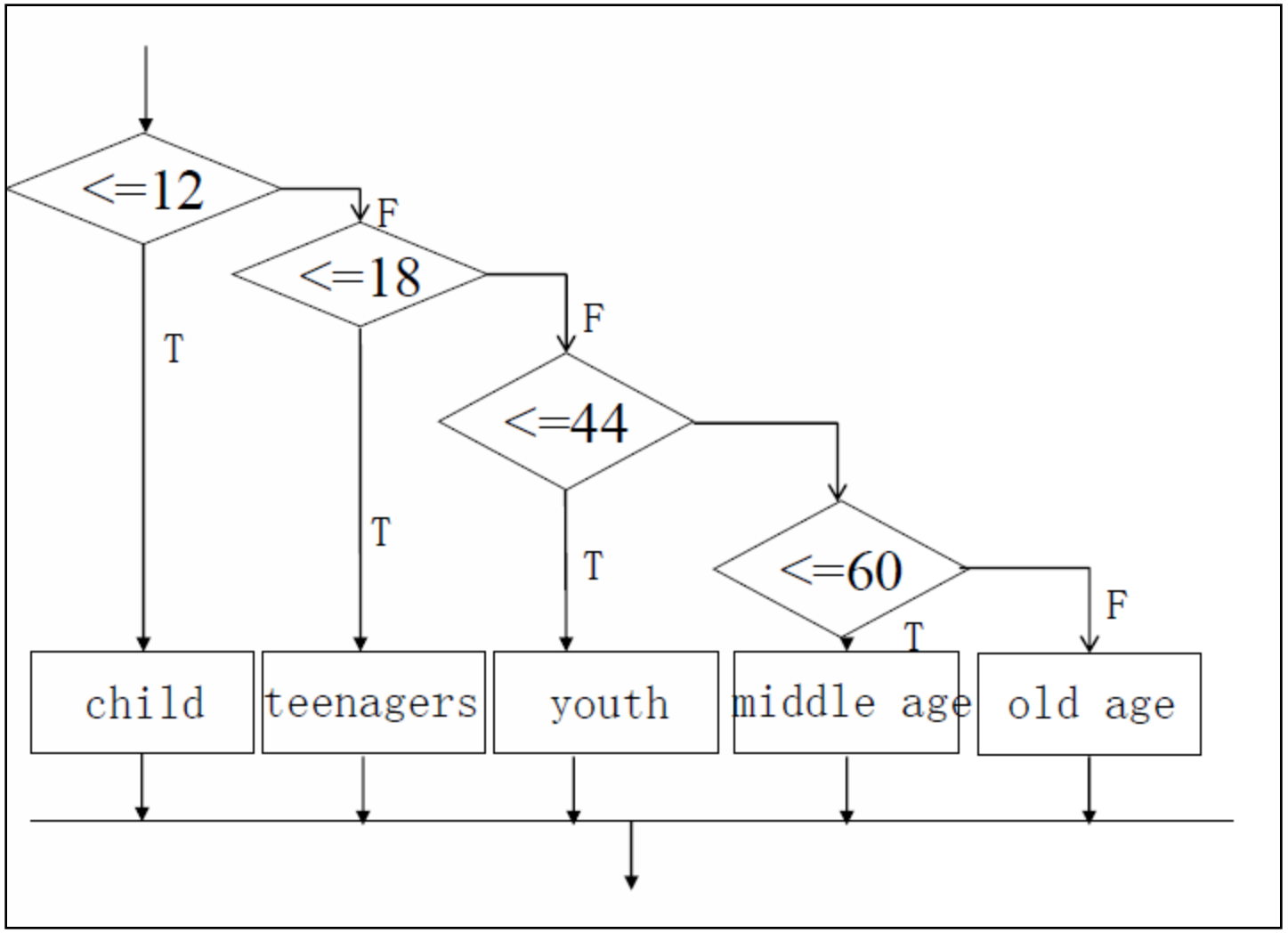
Operator	Symbol	Meaning
AND	&&	X && y is true if BOTH x and y are true
OR	||	X || y is true if at least one of x and y is true
NOT	!	!x is true if x is false

Logical values as operands or in tests: true = non-zero, false=zero

Logical values returned as results of expressions: true = 1, false=zero

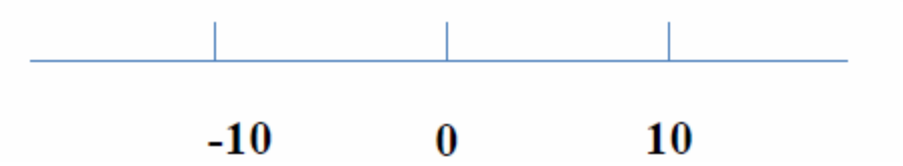
Example: `5 || 0` is `1`

multi branch selection structure

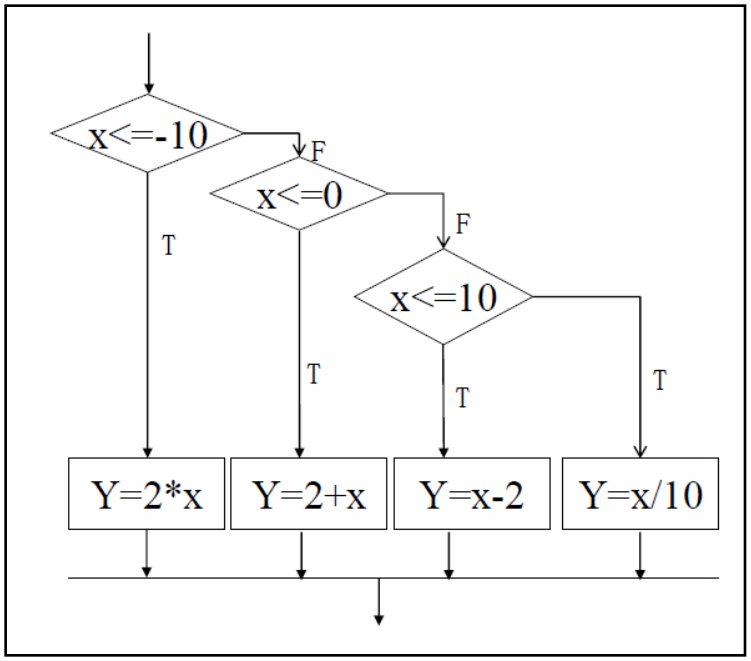


Calculate piecewise function

$$y=\begin{cases} 2*x & x<= -10 \\ 2+x & -10<x<=0 \\ x-2 & 0<x<=10 \\ x/10 & x>10 \end{cases}$$



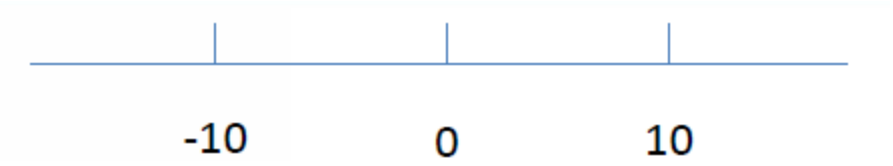

```
#include<stdio.h>
main( )
{  float x,y;
scanf("%f",&x);
if(x<=-10)              y=2*x;
if(-10<x&&x<=0)   y=2+x;
if(0<x&&x<=10)     y=x-2;
if(x>10)                 y=x/10;
printf("\ny=%f",y);
```




```
if ( expression 1)
    program statement 1
else if ( expression 2)
    program statement 2
else
    program statement 3
```

execution process is as follows: first calculate expression 1, and if the value of expression 1 is true, execute statement 1, Otherwise, calculate the value of expression 2. If the value of expression 2 is true, execute statement 2 and continue until the final result is obtained.

```
#include<stdio.h>
main( )
{
   float x,y;
   scanf("%f",&x);
   if(x<=-10)
        y=2*x;
   else   if(x<=0)
        y=2+x;
    else  if(x<=10)
        y=x-2;
    else  y=x/10;
    printf("\ny=%f",y);
}
```




```
-12
y=-24.000000
Press any key to continue
```

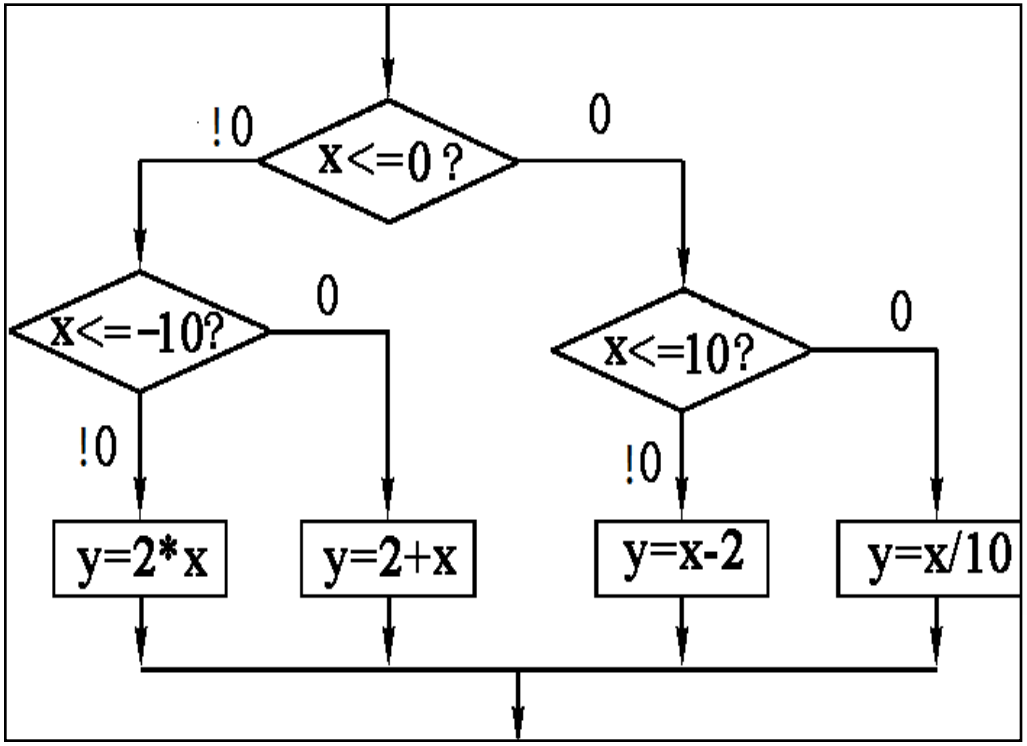
```
-8
y=-6.000000
Press any key to continue
```

```
5
y=3.000000
Press any key to continue
```

```
15
y=1.500000
Press any key to continue
```

nestification of “if”

implement statements 1 or 2 in the single branch if structure or double branch if structure, using if structure statements



$$y=\begin{cases} 2*x & x<=-10 \\ 2+x & -10<x<=0 \\ x-2 & 0<x<=10 \\ x/10 & x>10 \end{cases}$$

```
#include<stdio.h>
main( )
{ float x,y;
scanf("%f",&x);
 if(x<=0)
      if(x <=-10)
            y=2*x ;
     else
            y=2+x;
else
     if(x<=10)
             y=x-2;
     else
            y=x/10;
printf("\ny=%f ",y);
}
```

The structure of statement 1 and statement 2 using if statements as the upper level of if statements is called nesting of if statements

Nesting of If statements

```
if  (expression 1)
     if  (expression 1-1)
          statement 1-1;
    else
          statement 1-2;
else
    if (expression 2-1)
          statement 2-1;
    else
          statement 2-2;
```

Implement if~else correct pairing method

```
example:  if (a==b)
              if(b==c)
                    printf("a==b==c");
          else
              printf("a!=b");
```

```
modify:  if (a==b)
           {    if(b==c)
                    printf("a==b==c");
            }
          else
              printf("a!=b");
```

Implement if~else correct pairing method: add {}

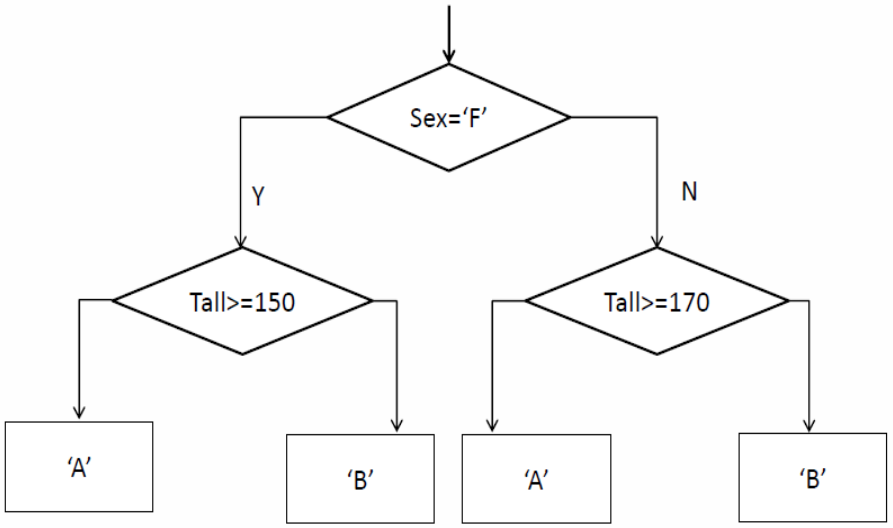
If~else pairing principle

Else is paired with an if statement that is closest to it on the same layer and does not have any other Else statements matching it

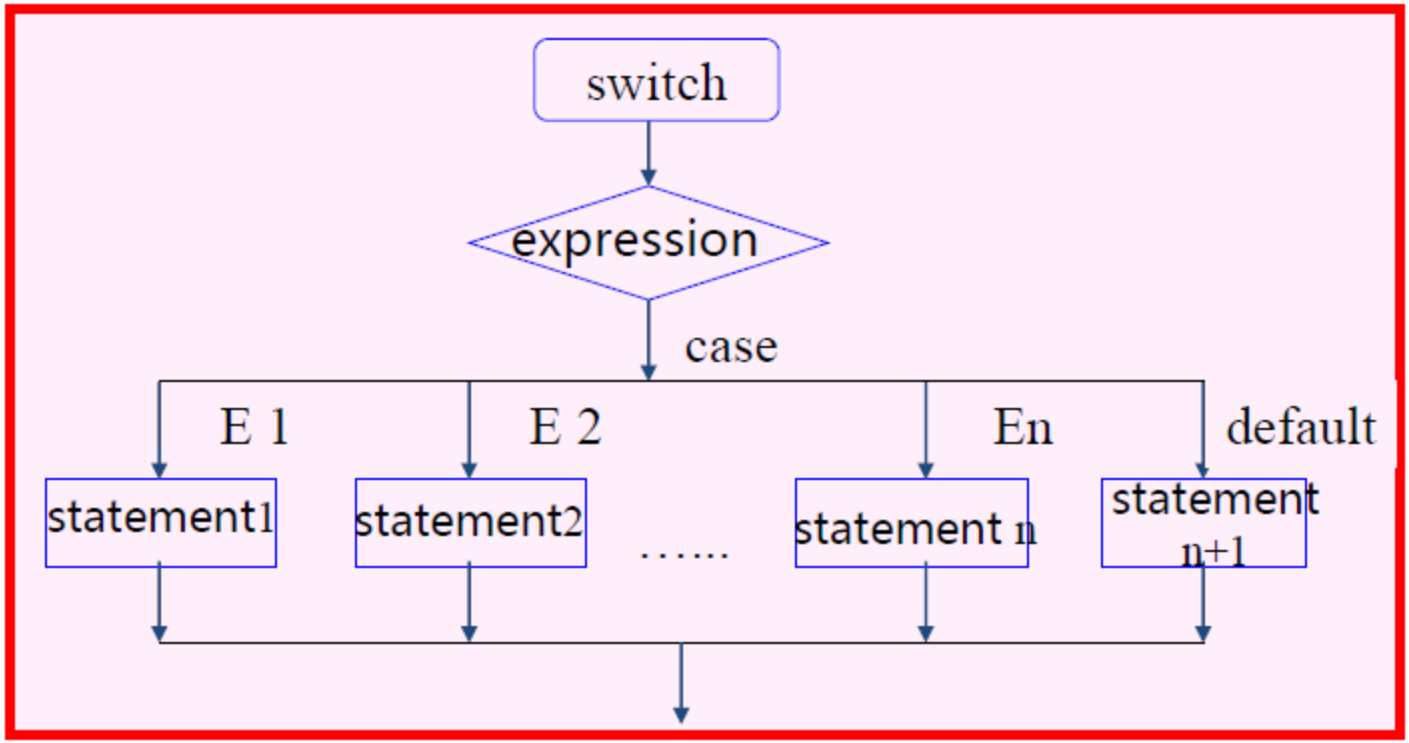
```
if(……)
    if(……)
        if(……)
        else…...
    else…...

else…...
```

Programming implementation:
Classify based on two numbers, sex and talk, if sex is' F ‘, When call is greater than or equal to 150, output A; otherwise, output B; If sex is not 'F', when total is greater than or equal to 170, output A; otherwise, output B.

```
#include<stdio.h>
main()
{
int tall;
char sex;
printf( "input sex and tall :  " );
scanf( "%c %d", &sex, &tall );
if  (sex=='F')
    if  (tall>=150)
        printf(" A\n ");
    else
        printf(" B\n ");
else
    if  (tall>=170)
        printf(" A\n ");
    else
       printf(" B\n ");
}
```



Switch statement



The `switch` statement

```
switch ( expression )
{
    case value1:
        program statement
        program statement
        ...
        break;
    case value2:
        program statement
        program statement
        ...
        break;
        ...
    case valuen:
        program statement
        program statement
        ...
        break;
    default:
        program statement
        program statement
        ...
        break;
}
```

The *expression* is successively compared against the values *value1*, *value2, ..., valuen*. If a case is found whose value is equal to the value of *expression*, the program statements that follow the case are executed.

The switch test expression must be one with an **integer value** (including type char) (No float !).

The case values must be integer-type **constants** or integer constant expressions (You can't use a variable for a case label !)

```
# include <stdio.h>
void main( )
{
   char grade;
   scanf("%c",&grade);
   printf("Your score:");
   switch(grade)
  { case 'A': printf("85～100\n");break;
     case 'B': printf("70～84\n");break;
     case 'C': printf("60～69\n");break;
     case 'D': printf("<60\n");break;
      default:  printf("enter data error!\n");
   }
```

```
A
Your score:85~100
70~84
60~69
<60
enter data error!
```