

Ch.6 Conditional Statements

What you will learn in this chapter



- What is a conditional statement ?
- if statement
- if, else statement
- nested if statements
- switch statement
- break statement
- continue statement
- goto statement

Ability to change the execution order of statements when conditions are met, as needed.



Conditional statement

- If a program does not have a selection structure, the program will always repeat the same actions .



Control statement

Control Statement

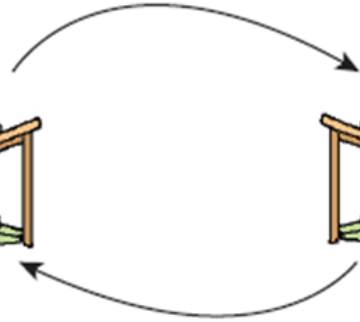
Conditional Statement

If you have good grades when you graduate, you can get a job, otherwise you have to take an exam.



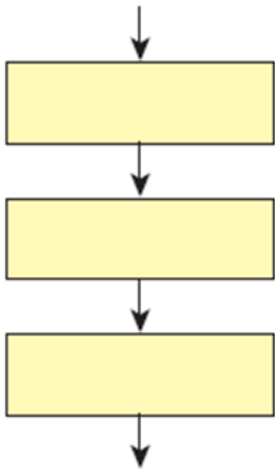
Loop Statement

You have to study repeatedly until your TOEIC score exceeds 600

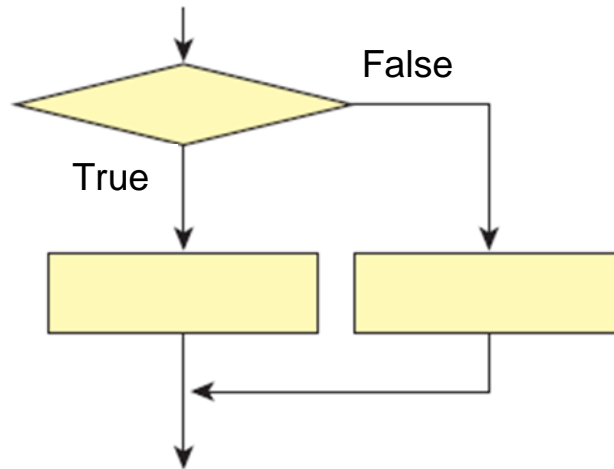


3 types of control structures

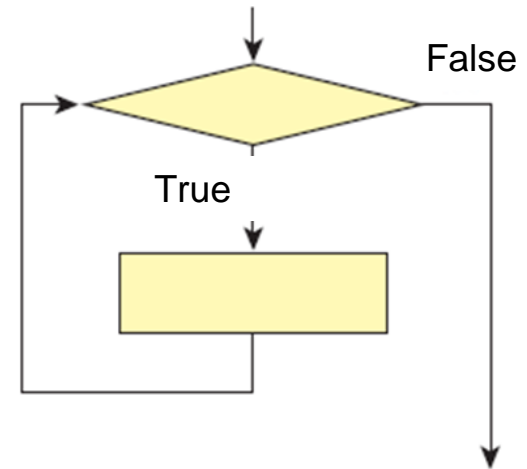
Sequential structure



Selection structure

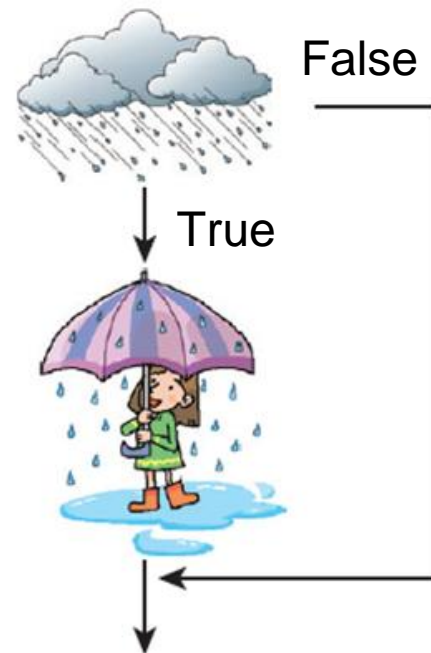
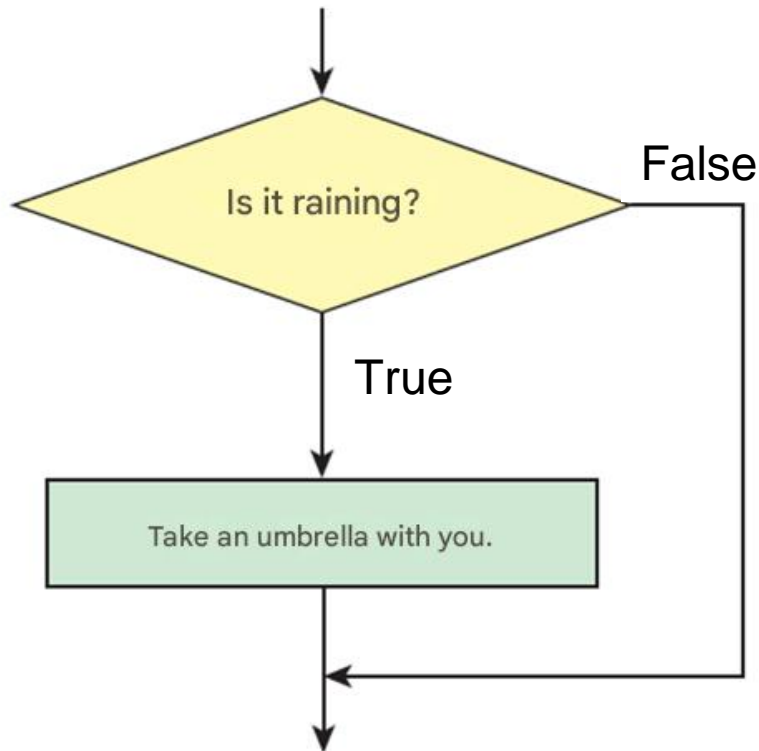


Repeating structure

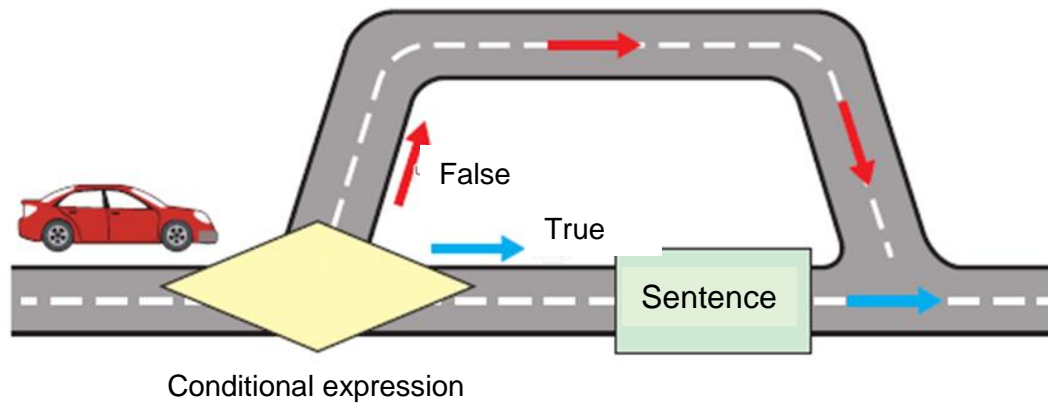
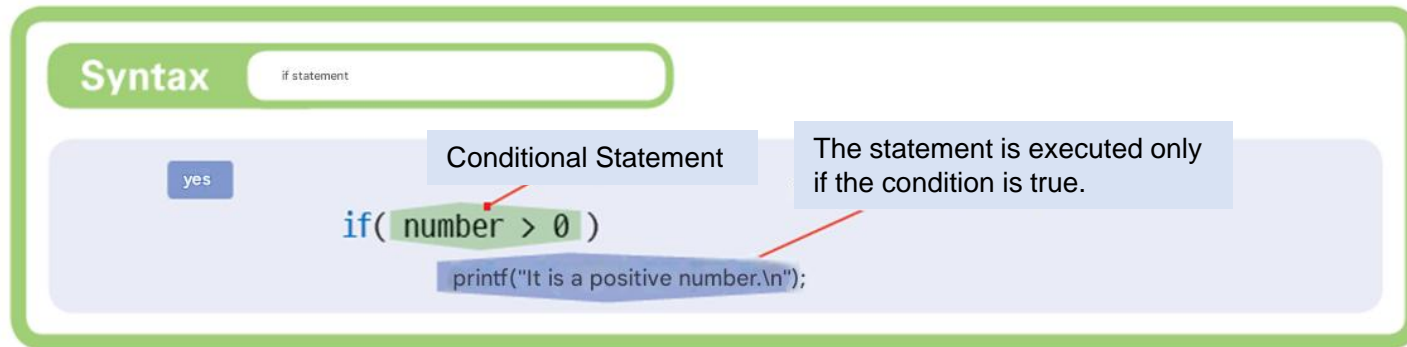


if statement

- In everyday life, there are many instances where decisions must be made based on conditions.



if statement structure



if statement example

If number is greater than 0

```
if ( number > 0 )  
    printf("It is a positive number \n ");
```

Prints " It is a positive number"

```
if ( temperature < 0 )  
    printf("It's below zero now.\n "); // Only run when the condition is true  
  
printf("Current temperature is % degrees.\n" , temperature); // Always run
```

if statement ends,
the following line of the if-statement
is executed .

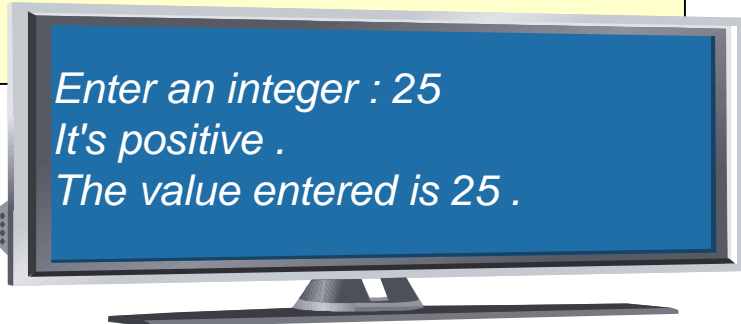
Example

```
#include <stdio.h>
int main( void )
{
    int number;

    printf ( " Enter an integer : " );
    scanf ( "%d" , &number);

    if ( number > 0 ) {
        printf ( " It is a positive number ." );
    }
    printf ( " The entered value is %d .", number );

    return 0;
}
```



*Enter an integer : 25
It's positive .
The value entered is 25 .*

Example

```
// Program to find absolute value using if statement
```

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

```
int main( void )
```

```
{
```

```
    int number;
```

```
    printf ( " Enter an integer : " );
```

```
    scanf ( "%d" , &number);
```

```
    if ( number < 0 ) {
```

```
        number = -number;
```

```
    }
```

```
    printf ( " The absolute value is %d .\n" , number);
```

```
    return 0;
```

```
}
```

If the user entered -5

-5 Since it is < 0 ,
the conditional statement
is executed.



Enter an integer : -5
The absolute value is 5 .

Compound sentence

- Compound statement
 - Grouping sentences using curly brackets,
 - Also called block .

```
if ( score >= 60 )  
{  
    printf (" You passed .\n");  
    printf (" You can also get a scholarship .\n ");  
}
```

If the condition is true,
two statements are
grouped and executed.

A brief notation for conditional statements

standard method	concise notation
<pre>if(x != 0) printf("x is not 0.\n");</pre>	<pre>if(x) printf("x is not 0.\n");</pre>
<pre>if(x == 0) printf("x is 0.\n");</pre>	<pre>if(!x) printf("x is 0.\n");</pre>

Error

Warning: Error Caution #1

You should not put a semicolon after the conditional expression of an if statement as follows. An if statement is a single sentence made up of a conditional expression and a statement. If you write it as follows, the if statement ends with `if(x > 0);` and the `printf` statement is executed regardless of the condition.

```
if( x > 0 );  
    printf("It is a positive number.\n");
```

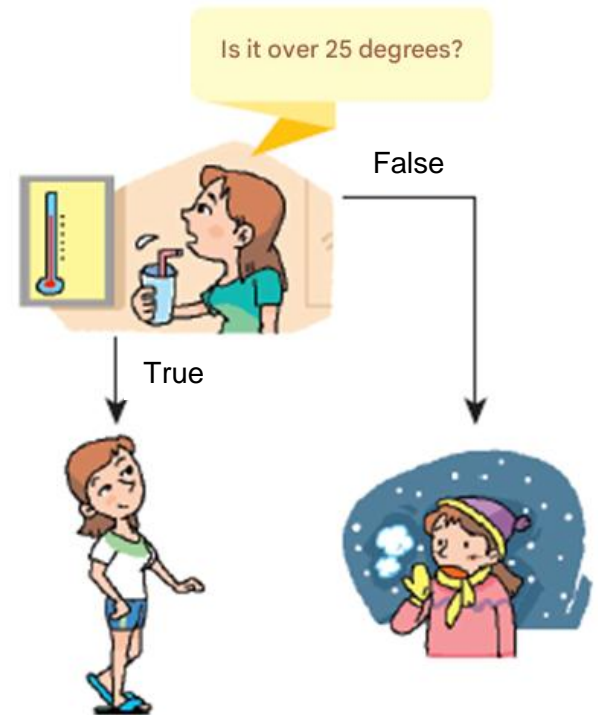
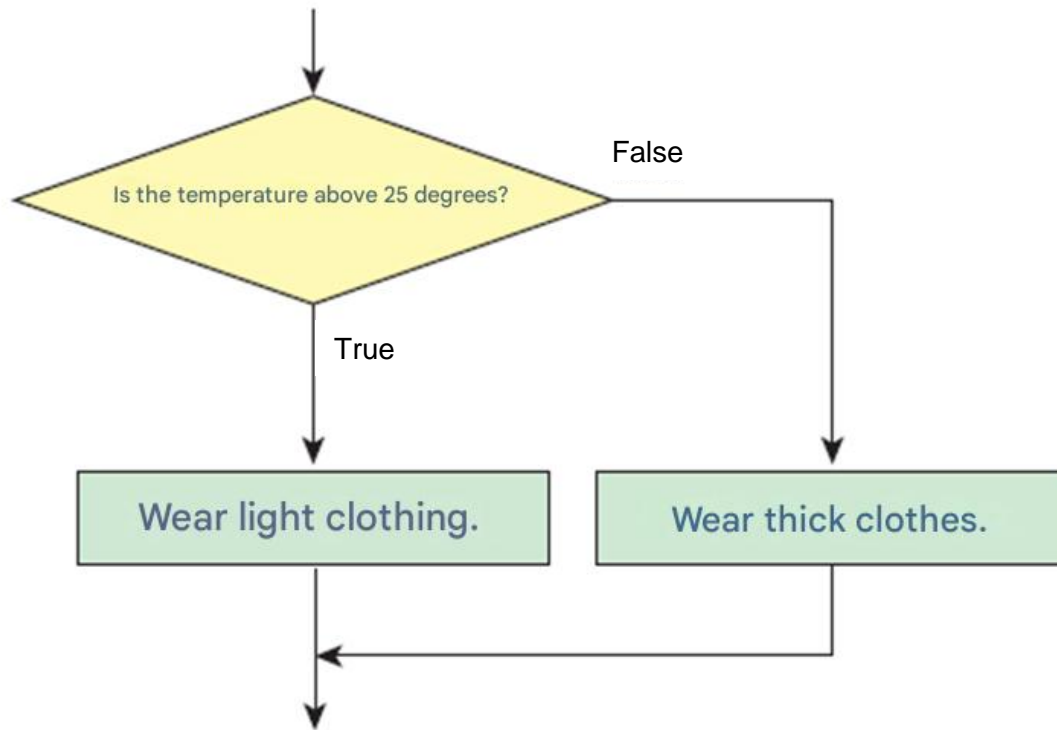
Warning: Error Note #2

A very common mistake is to use the `=` operator instead of the `==` operator when comparing two values. In this case, no comparison is made, and the values are simply assigned to the variable. True or false is determined based on the assigned value.

```
if( x = 0 )  
    printf("x is 0.");
```

In this case, 0 is assigned to x, so it is always false. You should write `x == 0`. To avoid this error, some people write `0 == x`. If `0 = x`, a syntax error occurs.

if-else statement



if-else statement

Syntax

if-else statement

yes

Conditional expression.
`if(number > 0)`

`printf("It is a positive number.\n");`

`else`

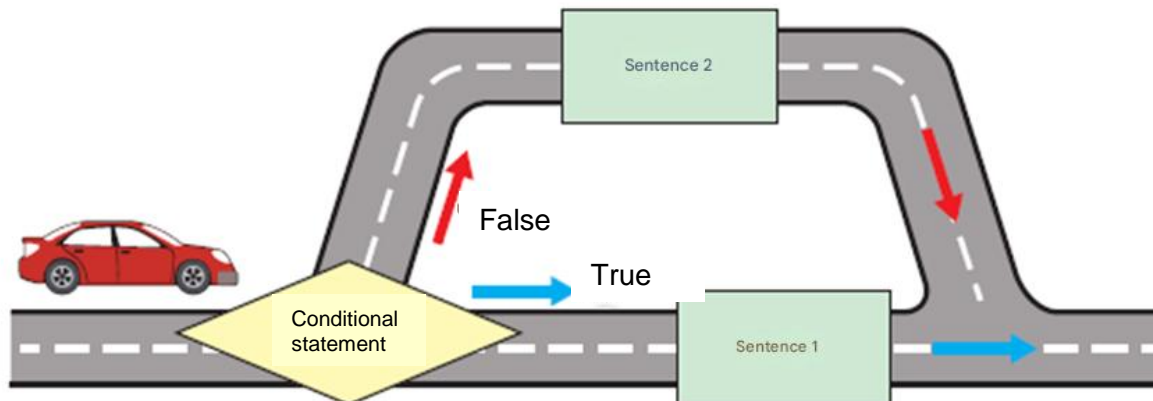
`printf("Not a positive number.\n");`

If the condition is true

Statement 1 is executed.

Otherwise, sentence 2 is true.

It is done.



if-else statement

```
if ( score >= 60 )  
    printf ( " Pass .\n" );  
else  
    printf ( " Failed .\n " );
```

Run if score is 60 or higher

Run if score is less than 60

```
if ( score >= 60 )  
{  
    printf ( " Pass .\n" );  
    printf ( " You can also get a scholarship .\n" );  
}  
else  
{  
    printf ( " Failed .\n " );  
    printf ( " Try again .\n" );  
}
```

Run if score is 60 or higher

Run if score is less than 60

Complex conditional expressions are also possible

- Multi comparison code

```
if ( score >= 80 && score < 90 )  
    grade = 'B' ;
```

- Code to count the number of white-space characters

```
if ( ch == ' ' || ch == '\n' || ch == '\t' )  
    white_space ++;
```

Conditional operator

- A simple if-else statement can also be expressed using the conditional operator we learned in Chapter 4.

```
(score >= 60 ) ? printf ( " Pass.\n " ) : printf ( " Fail.\n " );
```

```
bonus = (( years > 30 ) ? 500 : 300 );
```

if-else statement style

Style

If-else statements usually use one of the following two styles.

Compound sentence are easier to read if indented.

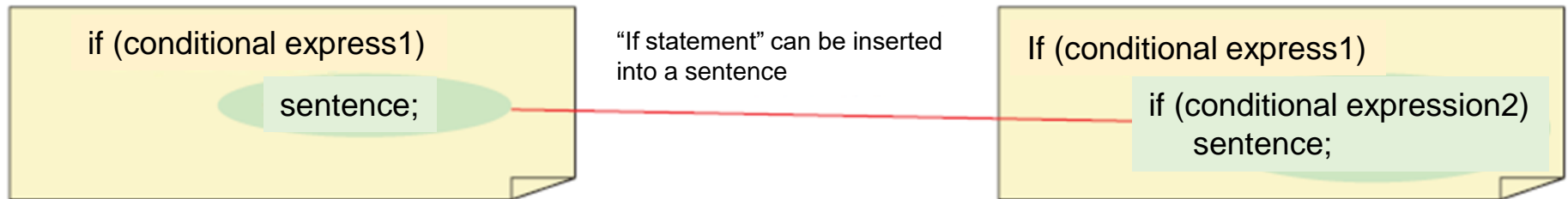
```
if( expression )  
{  
    → statement11;  
    statement12;  
    ...  
}  
else  
{  
    → statement21;  
    statement22;  
    ...  
}
```

It is sometimes written in this format To save space.

```
if( expression ){  
    statement11;  
    statement12;  
    ...  
}  
else {  
    statement21;  
    statement22;  
    ...  
}
```

Nested if

- if statement contains another if statement



Nested if

```
if ( score >= 80 )  
    if ( score >= 90 )  
        printf ( " Your grade is A. \n" );
```

```
if ( score >= 80 )  
    if ( score >= 90 )  
        printf ( " Your grade is A. \n" );  
    else  
        printf ( " Your grade is B. \n" );
```

Matching problem of if and else

The else clause matches the nearest if clause .

if(score > 80)

X

if(score >= 90)

else

printf (" Your grade is A\n");

printf (" Your grade is B\n")

if (score >= 80)

{

if (score >= 90)

printf (" Your grade is A. \n ");

}

else

printf (" Your grade is neither an A nor a B. \n");

If you want to match different if-clauses with different else-clauses, use curly brackets to group them into a block .

Consecutive if

Syntax

consecutive if statements

grammar

If(conditional expression 1)

Sentence 1;

else if(condition2)

Sentence 2;

else if(condition 3)

Sentence 3;

else

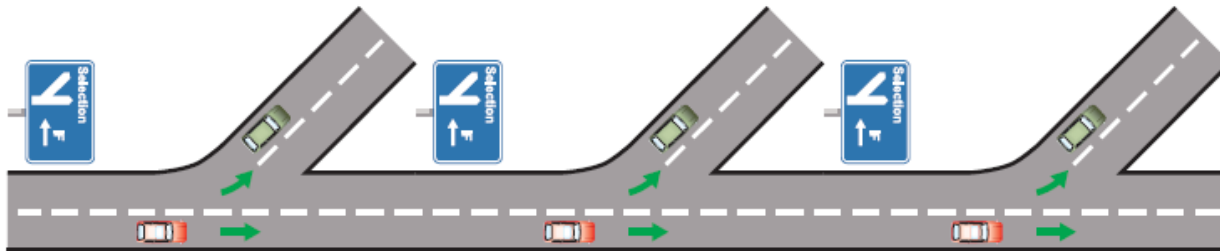
Sentence 4;

If condition 1 is true, statement 1 is executed.

Otherwise, if condition 2 is true, statement 2 is executed.

Otherwise, if condition 3 is true, statement 3 is executed.

Otherwise, sentence 4 is executed.



Example of determining grades

- Let's write and run a program that receives students' grades and prints out their grades.
1. Receive the points from user
 2. If Points ≥ 90 , then 'A'
 - If Points ≥ 80 , then 'B'
 - If Points ≥ 70 , then 'C'
 - If Points ≥ 60 , then 'D'
 - If Points < 60 , then 'F'



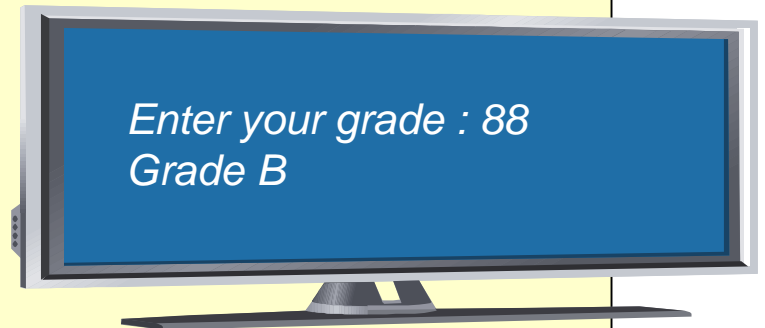
Example of determining grades

```
#include <stdio.h>

int main( void )
{
    int score;

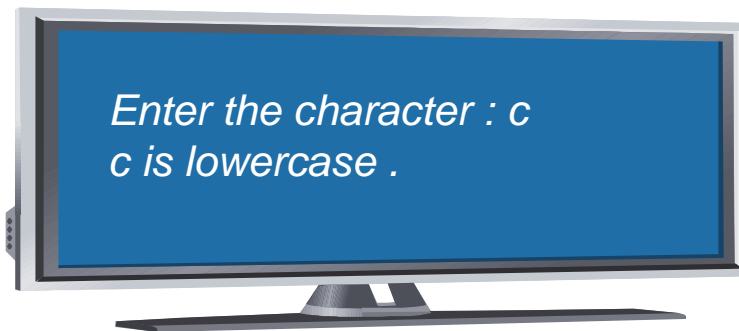
    printf ( " Enter your grades : " );
    scanf ( "%d" , &score);

    if (score >= 90)
        printf("Passed : Grade A\n" );
    else if (score >= 80)
        printf("Passed : Grade B\n" );
    else if (score >= 70)
        printf("Passed : Grade C\n" );
    else if (score >= 60)
        printf("Passed : Grade D\n" );
    else
        printf("Failed : Grade F\n" );
    return 0;
}
```



Character classification example

- Take characters from the keyboard and separate them into uppercase letters (A-Z), lowercase letters (a-z), numbers (0-9), and other characters.
- Let's use `getchar()` as a function that accepts characters.



10진수	16진수	문자	10진수	16진수	문자	10진수	16진수	문자	10진수	16진수	문자
0	0x00	NUL	32	0x20	Space	64	0x40	@	96	0x60	`
1	0x01	SOH	33	0x21	!	65	0x41	A	97	0x61	a
2	0x02	STX	34	0x22	"	66	0x42	B	98	0x62	b
3	0x03	ETX	35	0x23	#	67	0x43	C	99	0x63	c
4	0x04	EOT	36	0x24	\$	68	0x44	D	100	0x64	d
5	0x05	ENQ	37	0x25	%	69	0x45	E	101	0x65	e
6	0x06	ACK	38	0x26	&	70	0x46	F	102	0x66	f
7	0x07	BEL	39	0x27	'	71	0x47	G	103	0x67	g
8	0x08	BS	40	0x28	(72	0x48	H	104	0x68	h
9	0x09	TAB	41	0x29)	73	0x49	I	105	0x69	i
10	0x0A	LF	42	0x2A	*	74	0x4A	J	106	0x6A	j
11	0x0B	VT	43	0x2B	+	75	0x4B	K	107	0x6B	k
12	0x0C	FF	44	0x2C	,	76	0x4C	L	108	0x6C	l
13	0x0D	CR	45	0x2D	-	77	0x4D	M	109	0x6D	m
14	0x0E	SO	46	0x2E	.	78	0x4E	N	110	0x6E	n
15	0x0F	SI	47	0x2F	/	79	0x4F	O	111	0x6F	o
16	0x10	DLE	48	0x30	0	80	0x50	P	112	0x70	p
17	0x11	DC1	49	0x31	1	81	0x51	Q	113	0x71	q
18	0x12	DC2	50	0x32	2	82	0x52	R	114	0x72	r
19	0x13	DC3	51	0x33	3	83	0x53	S	115	0x73	s
20	0x14	DC4	52	0x34	4	84	0x54	T	116	0x74	t
21	0x15	NAK	53	0x35	5	85	0x55	U	117	0x75	u
22	0x16	SYN	54	0x36	6	86	0x56	V	118	0x76	v
23	0x17	ETB	55	0x37	7	87	0x57	W	119	0x77	w
24	0x18	CAN	56	0x38	8	88	0x58	X	120	0x78	x
25	0x19	EM	57	0x39	9	89	0x59	Y	121	0x79	y
26	0x1A	SUB	58	0x3A	:	90	0x5A	Z	122	0x7A	z
27	0x1B	ESC	59	0x3B	;	91	0x5B	[123	0x7B	{
28	0x1C	FS	60	0x3C	<	92	0x5C	\	124	0x7C	
29	0x1D	GS	61	0x3D	=	93	0x5D]	125	0x7D	}
30	0x1E	RS	62	0x3E	>	94	0x5E	^	126	0x7E	~
31	0x1F	US	63	0x3F	?	95	0x5F	_	127	0x7F	DEL

Character classification example

```
// Program to classify characters
#include <stdio.h>
int main( void )
{
    char ch ;
    printf ( " Enter a character : " );

    ch = getchar ();
    if ( ch >= 'A' && ch <= 'Z' )
        printf ( "%c is uppercase .\n" , ch );
    else if ( ch >= 'a' && ch <= 'z' )
        printf ( "%c is a lowercase letter .\n" , ch );
    else if ( ch >= '0' && ch <= '9' )
        printf ( "%c is a number .\n" , ch );
    else
        printf ( "%c is a miscellaneous character .\n" , ch );

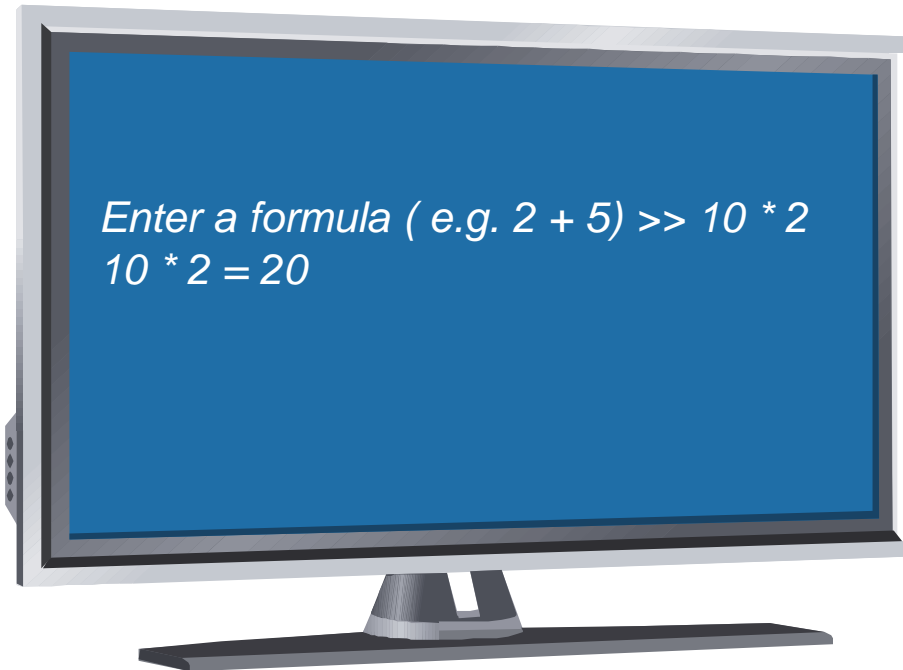
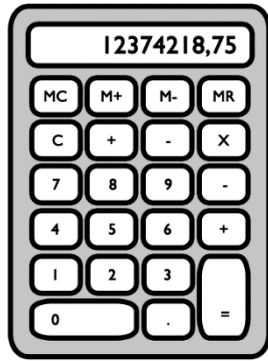
    return 0;
}
```



*Enter the character : c
c is lowercase .*

practice

Lab: Arithmetic Calculator



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

```
int main( void )
```

```
{
```

```
    char op;
```

```
    int x, y, result;
```

```
    printf ( " Enter a formula ( e.g. 2 + 5 ) >> " );
```

```
    scanf ( "%d %c %d" , &x, &op, &y);
```

```
    if ( op == '+' )
```

```
        result = x + y;
```

```
    else if ( op == '-' )
```

```
        result = x - y;
```

```
    else if ( op == '*' )
```

```
        result = x * y;
```

```
    else if ( op == '/' )
```

```
        result = x / y;
```

```
    else if ( op == '%' )
```

```
        result = x % y;
```

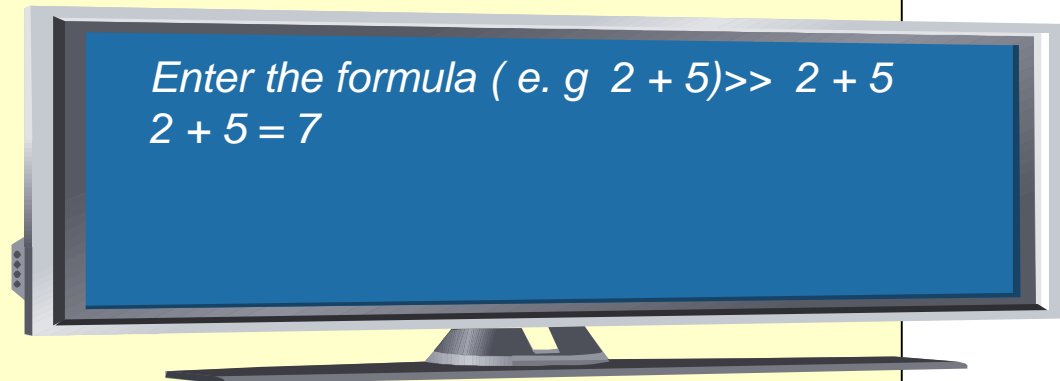
```
    else
```

```
        printf ( " Unsupported operator ." );
```

```
    printf( "%d %c %d = %d \n" , x, op, y, result);
```

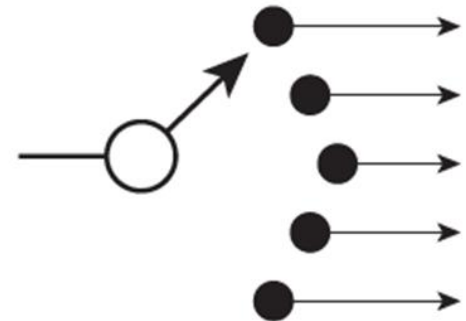
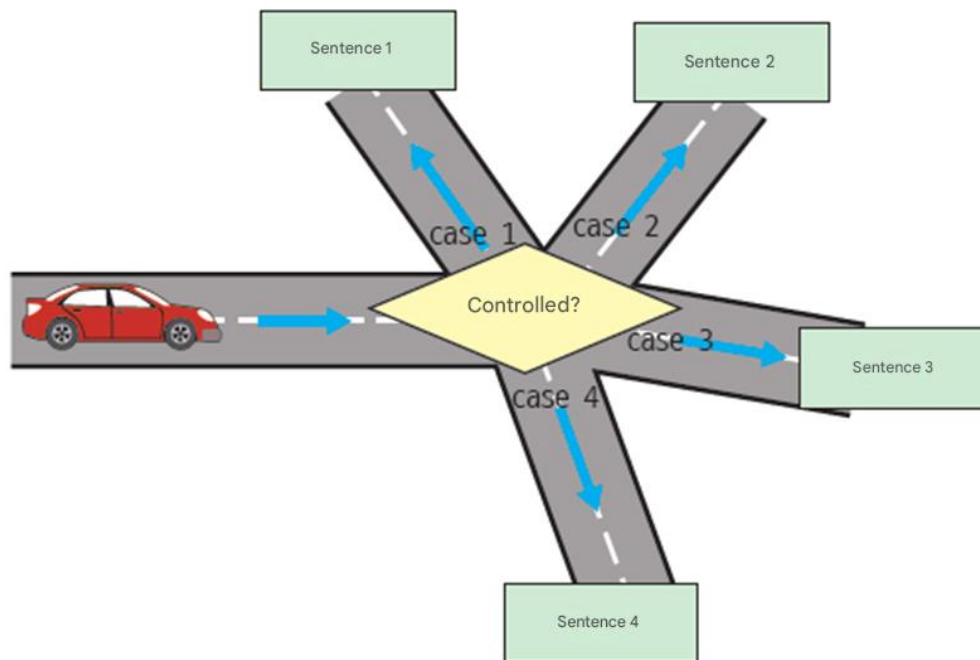
```
    return 0;
```

```
}
```



switch statement

- A control structure that can select one of several paths depending on the value of the control expression.



switch statement

Syntax

switch statement

grammar

switch (controlled)

{

case c1:

 sentence 1

 break;

If the value of the control expression is C1, It is executed

case c2:

 sentence 2

 break;

If the value of the control expression is C2, It is executed

...

default:

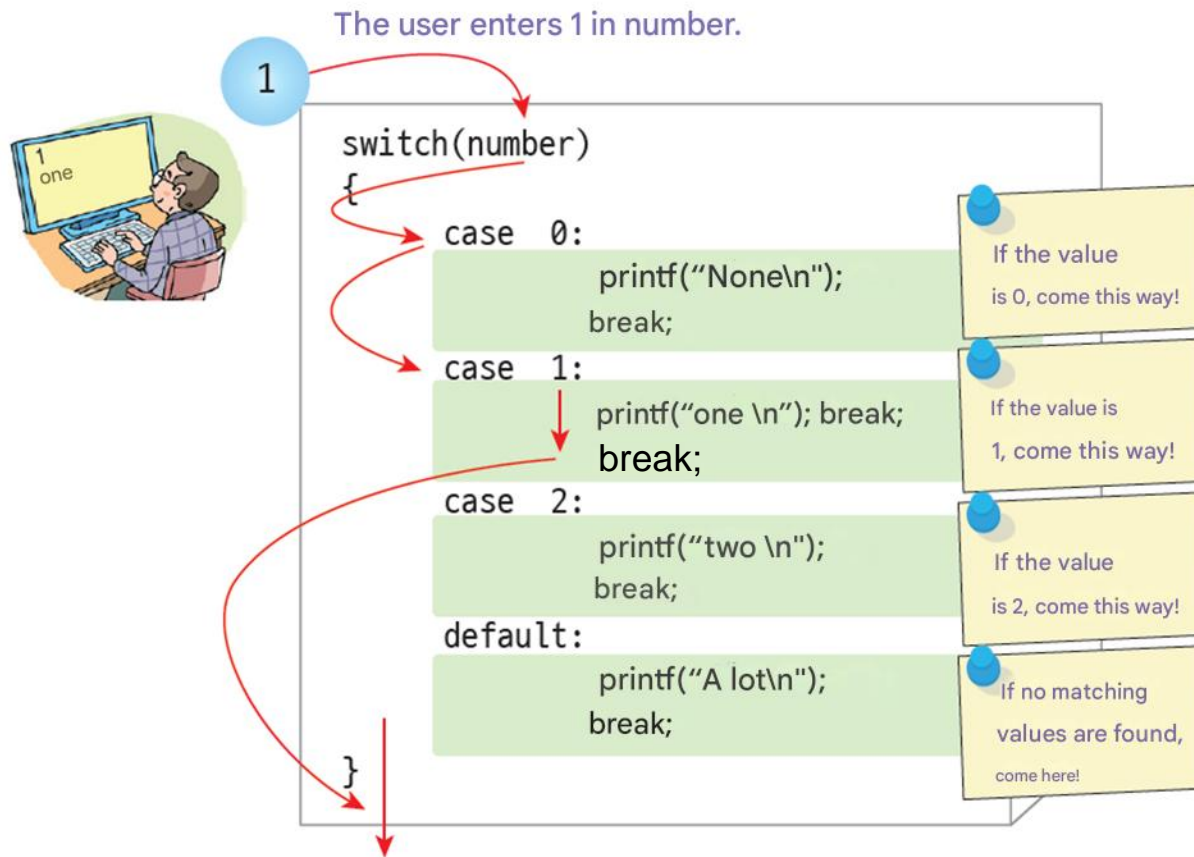
 sentence 3

 break;

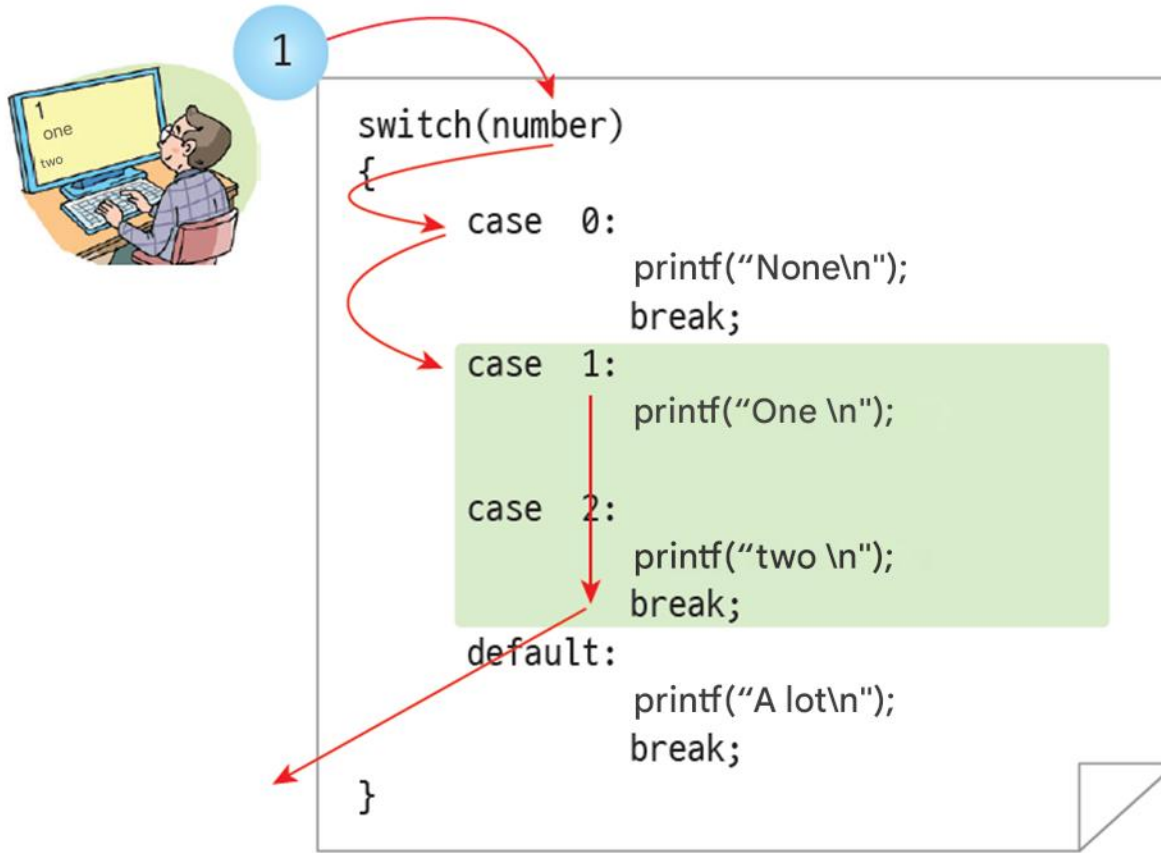
If no matching value is found, It is executed

}

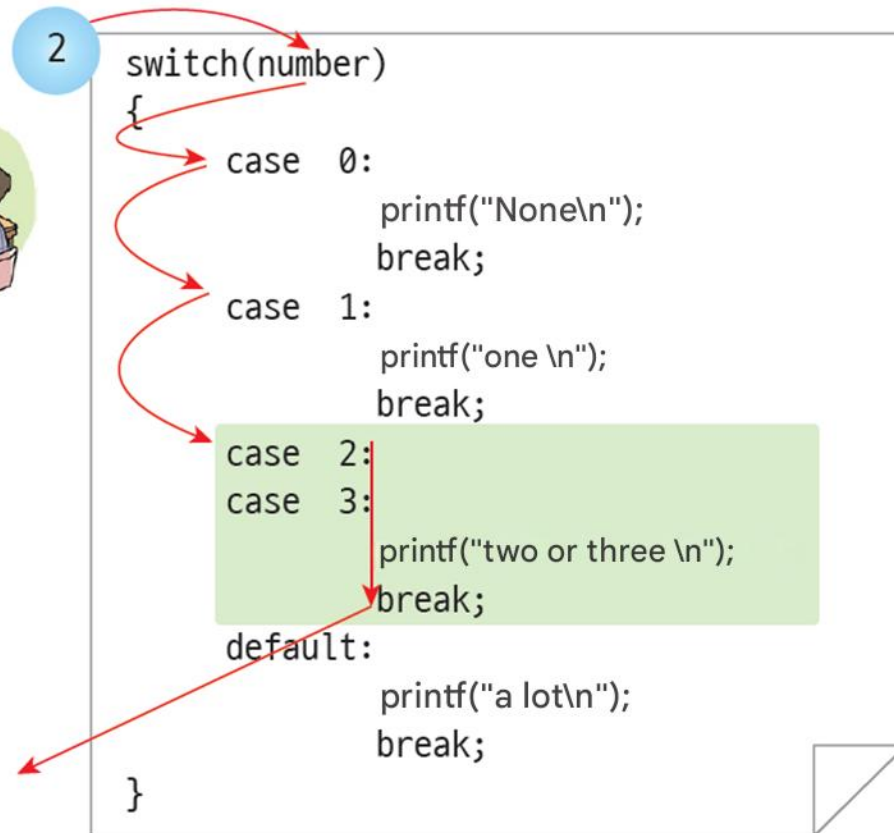
In case user enters 1



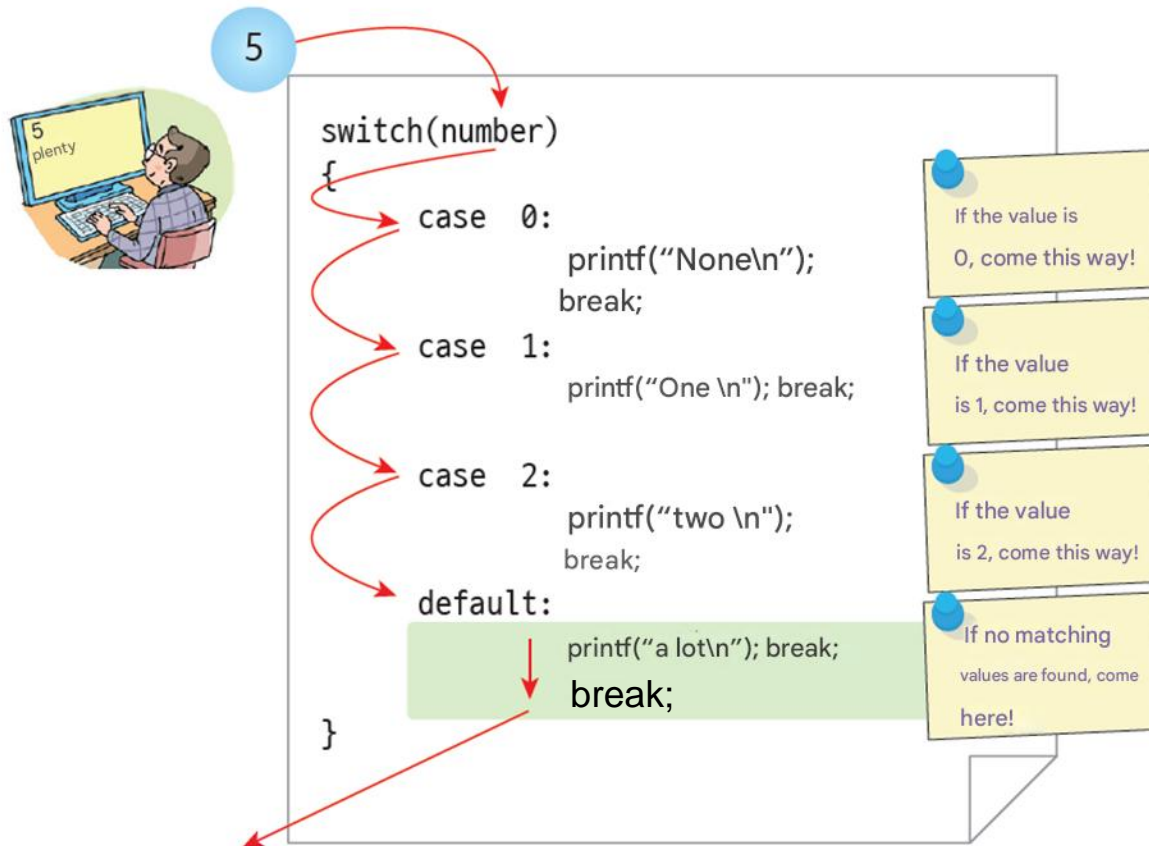
In case break is omitted



Intentional omission of break

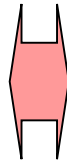


default statement



switch statement and if-else statement

```
switch (number) {  
    case 0:  
        printf(" None \n" );  
        break;  
    case 1:  
        printf(" one \n" );  
        break;  
    case 2:  
        printf(" two \n" );  
        break;  
    default :  
        printf(" many \n" );  
        break;  
}
```



```
if ( number == 0 )  
    printf(" None \n" );  
else if( number == 1 )  
    printf(" one \n" );  
else if( number == 2 )  
    printf(" two \n" );  
else  
    printf(" many \n" );
```

Caution: In the switch statement

```
switch (number)
{
    case x:    // Variable cannot be used .
        printf ( " Matches x . \n " );
        break ;
    case (x+2): // Formulas containing variables cannot be used .
        printf ( " Matches formula . \n " );
        break ;
    case 0.001: // Real numbers cannot be used .
        printf ( " error \n " );
        break ;
    case 'a' : // OK! The character, int, enum is can be used .
        printf ( " character \n " );
        break ;
    case "abc": // String cannot be used .
        printf ( " string \n " );
        break ;
}
```

When indicating a range of integer

```
switch (score) {  
    case 100:  
    case 99:  
    case 98:  
    ...  
    case 90:  
        printf("It's an A.\n");  
        break;  
    ...  
}
```



```
if( score >= 90 && score <= 100 )  
    printf("It's an A.\n");
```

You can also express the range of integers, but it is cumbersome.

When indicating a range of integer

```
int iscore;
...
iscore = score/10;           // In case of integer division, the remainder is lost.
switch (iscore) {
    case 9: grade = 'A'; break; // 90-100 is an A grade
    case 8: grade = 'B'; break; // 80-89 is a B grade
    case 7: grade = 'C'; break; // 70-79 is a C grade
    case 6: grade = 'D'; break; // 60-69 is a D grade
    default: grade = 'F'; break; // 59 points or less is an F grade
}
```

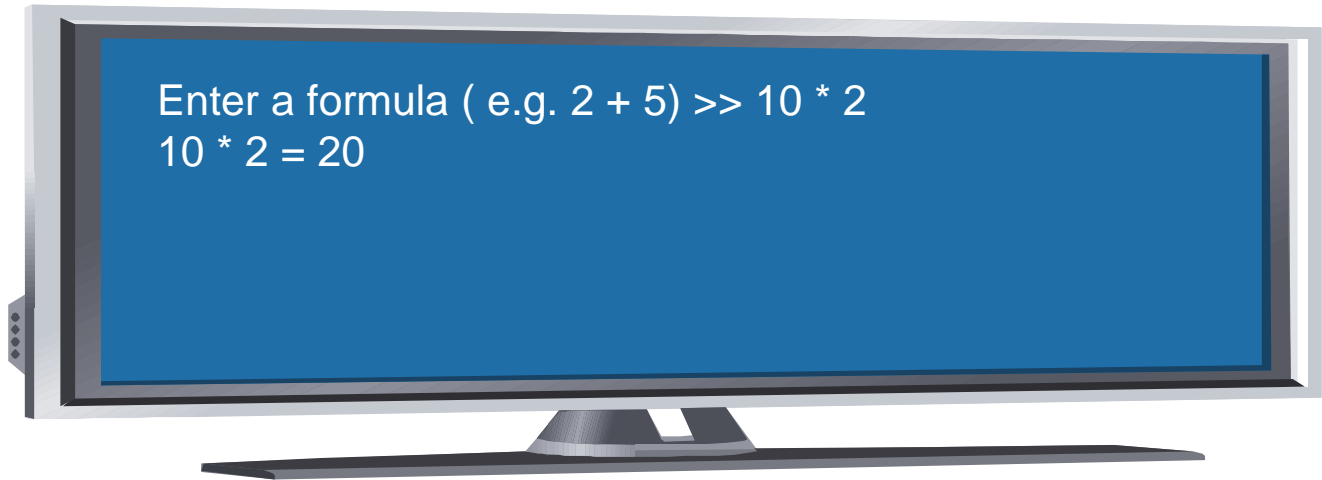


Which is more efficient: a switch statement or an if/else chain?

The difference is subtle, the switch statement is designed to be implemented efficiently with a simple jump table. Therefore, in most cases, it is better to use switch.
The code is concise and probably a little more efficient.

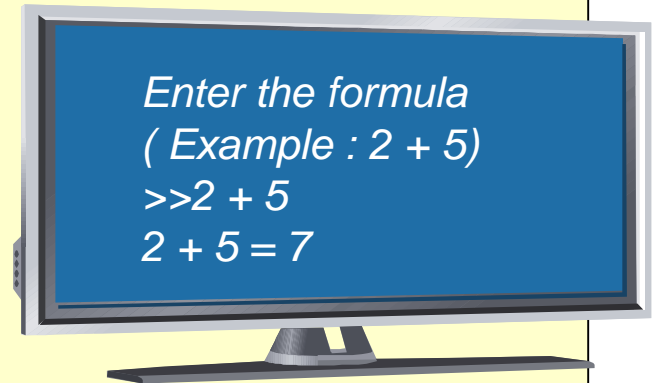
Lab: Arithmetic Calculator (switch version)

- Let's rewrite the previous arithmetic calculator example using a switch statement.




```
// Simple arithmetic calculator program
#include <stdio.h>
int main( void )
{
    char op;
    int x, y, result;
    printf ( " Enter a formula ( e.g. 2 + 5 ) >> " );
    scanf ( "%d %c %d" , &x, &op, &y);

    switch (op) {
        case '+' : result = x + y; break ;
        case '-' : result = x - y; break ;
        case '*' : result = x * y; break ;
        case '/' : result = x / y; break ;
        case '%' : result = x % y; break ;
        default : printf ("Unsupported operator.\n"); break ;
    }
    printf( "%d %c %d = %d \n" , x, op, y, result);
    return 0;
}
```

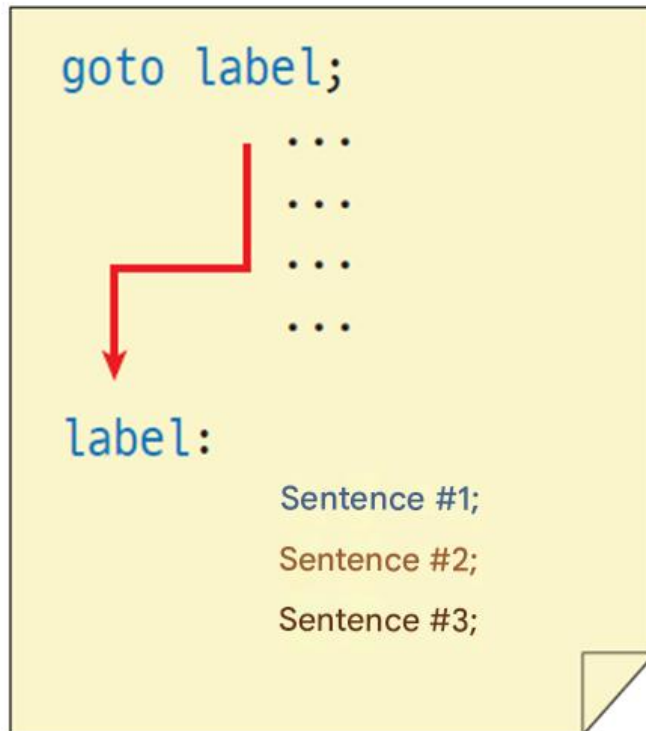


goto statement

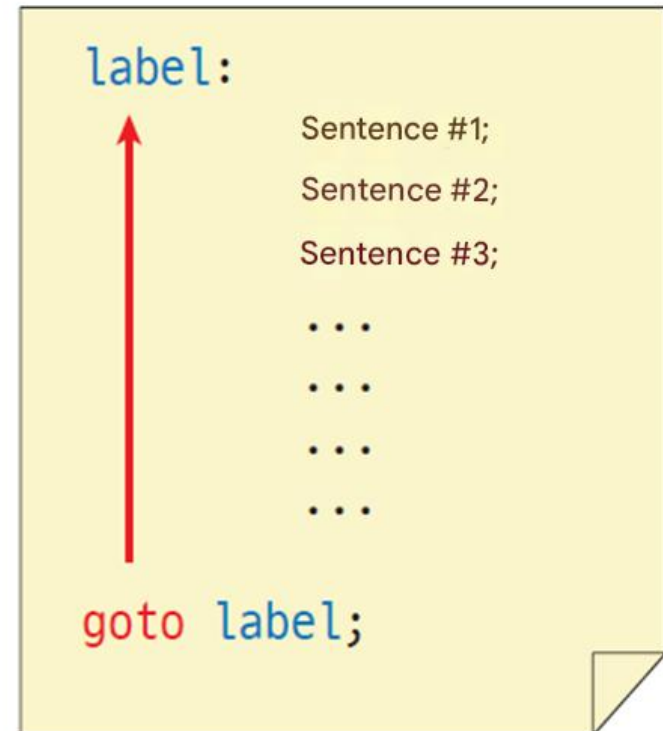
- Jump to any location unconditionally
- It is not recommended to use it



goto door



forward looking



retrospective reference

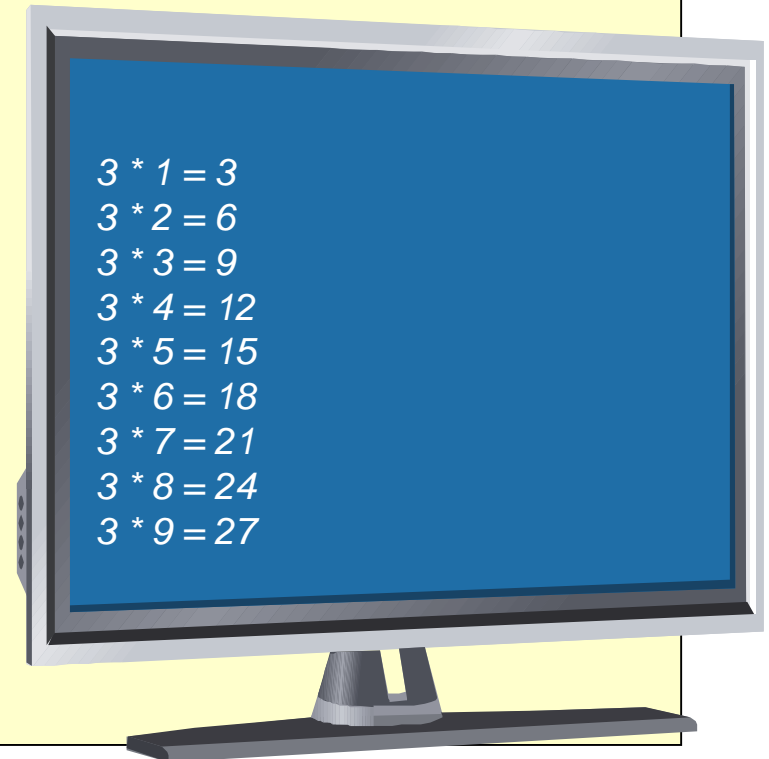
Example

```
// Multiplication table output program
#include <stdio.h>

int main( void )
{
    int i = 1;

loop:
    printf ( "%d * %d = %d \n" , 3, i , 3 * i );
    i ++;
    if ( i == 10 ) goto end;
    goto loop;

end:
    return 0;
}
```



Q & A

