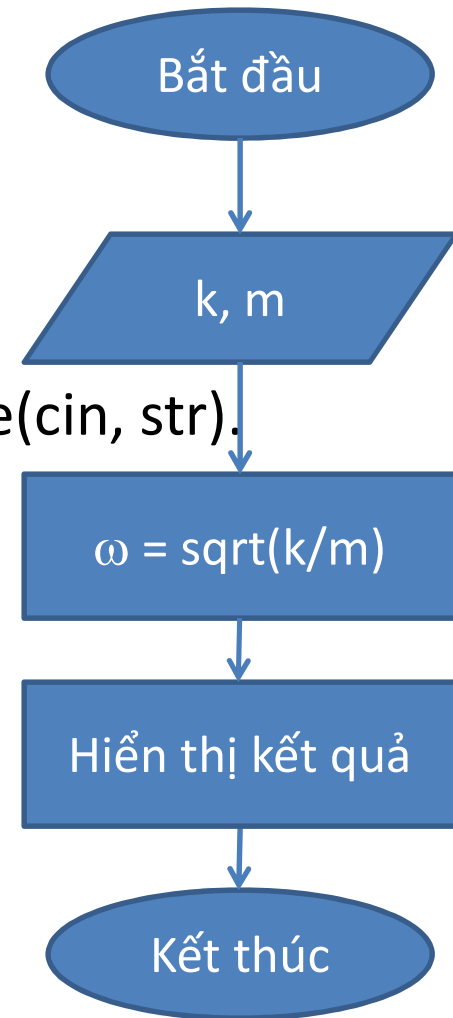


Nội dung 6:
Cấu trúc điều khiển

TS. Trần Thanh Hải

Tổng kết tuần 1-4

- `#include<iostream>`
- `using namespace std;`
- `cout<<"Hello "<<"INT006"<<"\n"<<endl;`
- `cin>>bien>>kytu>>xau;`
- `// một số hàm`
- `system("PAUSE"); rand()%(b-a+1)+a`
- `cin.getline(char *ten, 80), cin>>, gets, getline(cin, str).`
- `int main() // chương trình chạy tuần tự`
`{`
`cout<<"Tong ket 1 – 4 \n";`
`int a;`
`cin>>a;`
`}`



Xét bài toán

- Giải phương trình bậc 2:

$$y = ax^2 + bx + c = 0 \quad (a \neq 0)$$

- Nếu: $\Delta = b^2 - 4ac > 0$

- Thì có 2 nghiệm : $x_1 = \frac{-b + \sqrt{\Delta}}{2a}, x_2 = \frac{-b - \sqrt{\Delta}}{2a}$

- Nếu: $\Delta = b^2 - 4ac = 0$

- Thì có 1 nghiệm: $x = -\frac{b}{2a}$

- Nếu : $\Delta = b^2 - 4ac < 0$ thì phương trình vô nghiệm.

Xét bài toán

- **Bài 1:** Viết chương trình thực hiện các công việc sau:
 - a) In ra màn hình 3 dòng “Tuan 1”, 5 dòng “Tuan 2” và 10 dòng “Tuan 3”. (cout)
 - b) Nhập từ bàn phím 3, đến 10 số kiểu (int hoặc float, double) bằng lệnh cin>>.

Nội dung

- Giới thiệu.
- Câu lệnh if, if...else, if....else if,
- Câu lệnh for, while..., do...while, switch
- Break, continue, goto.
- Tổng kết.
- Mở rộng.

Giới thiệu

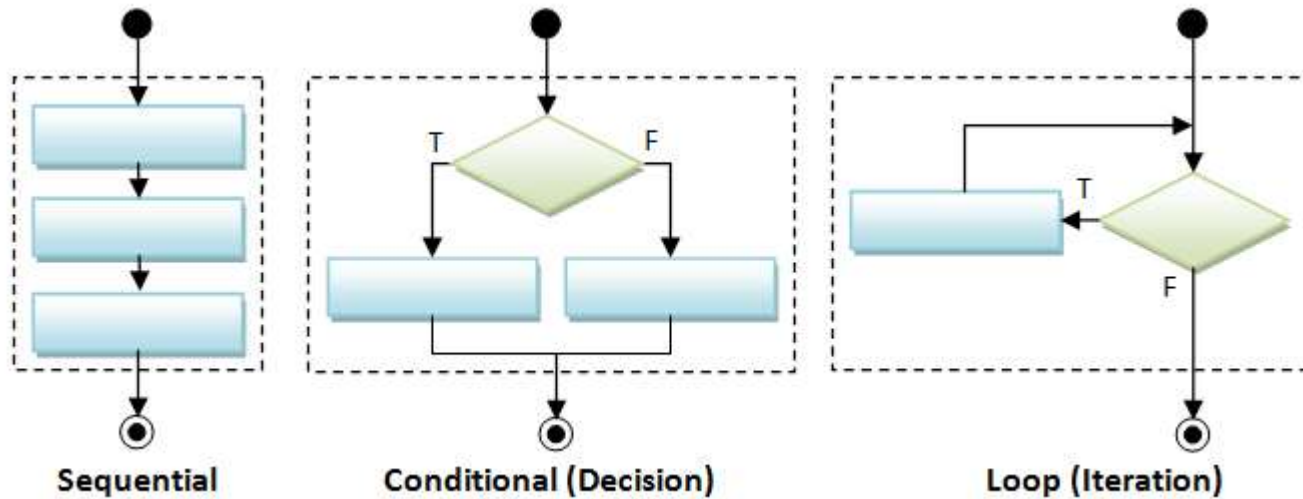
- Một câu lệnh C++ là: một chỉ dẫn riêng lẻ của chương trình, giống như khai báo biến và biểu thức luôn kết thúc bằng một dấu chấm phẩy ;
- Một chương trình tuần tự là: một dãy các lệnh được thực hiện kế tiếp nhau thứ tự nó xuất hiện trong chương trình.
- Chương trình không giới hạn số dòng lệnh.
- Tuy nhiên, trong phần lớn các chương trình, việc đi đến các quyết định còn phụ thuộc vào các giá trị được nhập vào hoặc các giá trị đã xử lý. Quá trình này được thực hiện bằng các lệnh điều khiển.

Giới thiệu

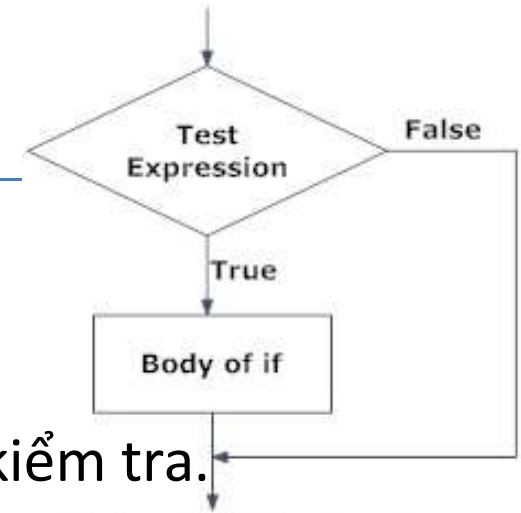
- Một chương trình có thể lặp lại một đoạn mã, hoặc đưa ra một quyết định hoặc rẽ nhánh. Với mục đích này, C++ cung cấp câu lệnh điều khiển phục vụ để hình thành cái gì phải làm trong chương trình, khi nào và trong điều kiện nào.
- Hai lệnh dùng để đi đến quyết định là: `if..else` và `switch`. Có 3 dạng lặp: `while`, `do`, và `for`.
- Tạo ra một chương trình không tuần tự.
- Toàn bộ khối lệnh được xem như là một câu lệnh đơn.....

Giới thiệu

- 3 Kiểu điều khiển cấu trúc cơ bản



1.Câu lệnh lựa chọn: if

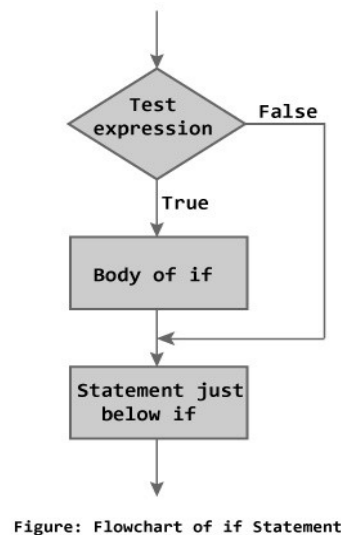


- **Cú pháp:** `if (condition) true_block`
 - + *if* là: từ khóa.
 - + **condition** (điều kiện) là: biểu thức được kiểm tra.
- **Cách làm việc:**
- Nếu **condition** là: true (đúng) thì thực thi câu lệnh **true_block**.
(condition – boolean Expression)
- Nếu **condition** là: false (sai) thì câu lệnh **true_block** không được thực thi (đơn giản chỉ bỏ qua nó) và chương trình tiếp tục thứ tự sau toàn bộ câu lệnh lựa chọn này.
- **Ví dụ:** `if (x == 100) // toan tu quan he condition = (x==100)`
`cout << "x is 100"; // (câu lệnh đơn không cần {})`
- Nếu x khác 100 thì câu lệnh này được bỏ qua và không thực hiện in ra màn hình.

1. Câu lệnh lựa chọn - if

- Để chạy gộp nhiều câu lệnh đơn nếu điều kiện đúng, thì câu lệnh này sẽ nằm trong ngoặc kín {}. Cú pháp:

```
if (x == 100)
{
    cout << "x is
    cout << x;
}
```



Test expression is true

```
int test = 5;
```

```
if (test < 10)
{
    // codes
}
```

```
// codes after if
```

Test expression is false

```
int test = 5;
```

```
if (test > 10)
{
    // codes
}
```

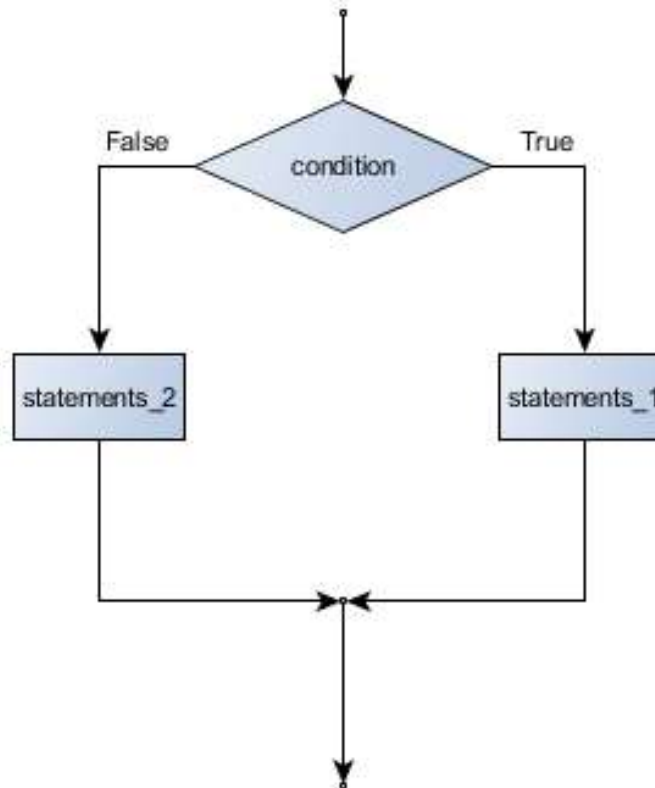
```
// codes after if
```

- Sự thụt dòng và sự xuống dòng không bị ảnh hưởng, do đó khối lệnh trên tương đương:

```
if (x == 100) { cout << "x is "; cout << x; }
```

1. Câu lệnh lựa chọn - if and else

- Cú pháp : `if (condition) true_block else false_block`
trong đó : `true_block` được thực thi nếu `condition` là đúng (true),
và thực thi `false_block` khi `condition` false (sai).



1. Câu lệnh lựa chọn - if and else

- Ví dụ.

```
if (x == 100)
    cout << "x is 100";
else
    cout << "x is not 100";
```

- In ra “x is 100”, nếu x có giá trị 100, nếu không nó sẽ in ra dòng “x is not 100” lên màn hình.

1. Câu lệnh lựa chọn - if and else

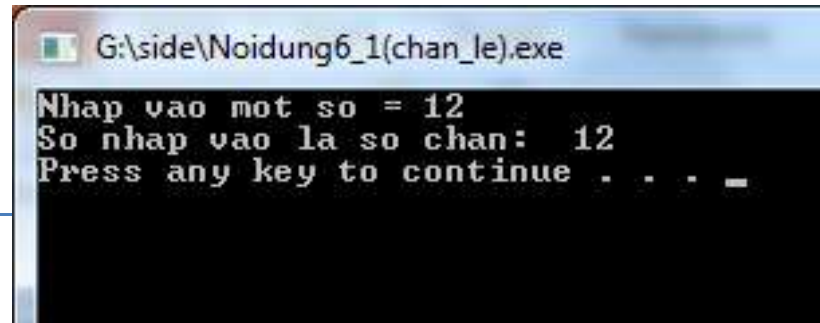
- Cấu trúc if + cấu trúc else có thể được ghép với mục đích kiểm tra dải giá trị. Ví dụ

```
if (x > 0)
    cout << "x is positive";
else if (x < 0)
    cout << "x is negative";
else // = 0
    cout << "x is 0";
```

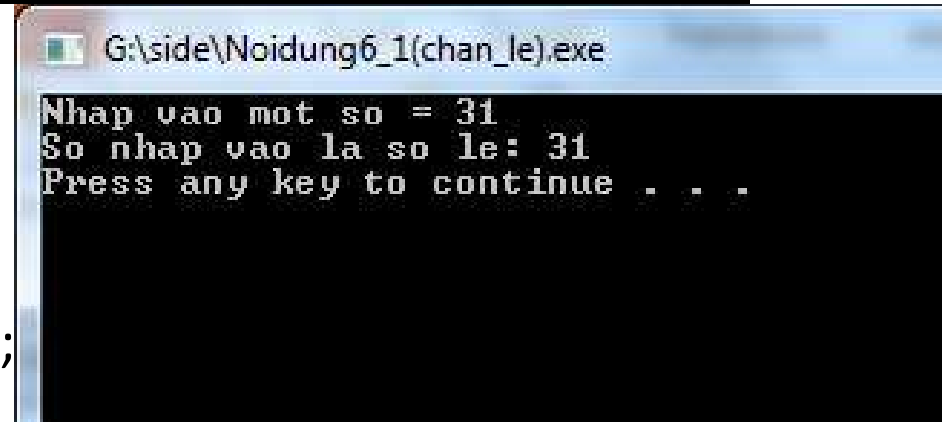
- in ra: "x is positive" nếu $x > 0$. Nếu âm hoặc bằng 0 bằng hai ghép nối if else.
- Chạy nhiều hơn một lệnh bằng cách gộp trong ngoặc {}.

Ví dụ 1: số chẵn – số lẻ

```
#include<iostream>
#include<stdlib.h>
using namespace std;
int main() {
    int n;
    cout<<"Nhập vào một số = ";
    cin>>n;
    if(n%2 == 0) // phân du = 0.
        cout<<"Số nhập vào là số chẵn: "<<endl;
    else
        cout<<"Số nhập vào là số lẻ: "<<endl;
    system("pause");
    return 0;
} // Noidung6_1.cpp
```



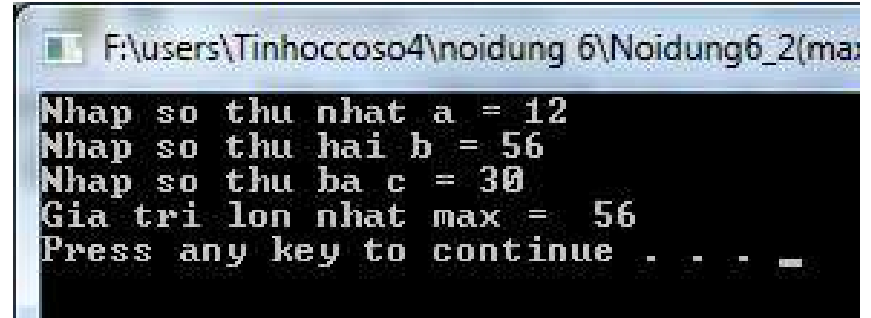
```
G:\side\Noidung6_1(chan_le).exe
Nhập vào một số = 12
Số nhập vào là số chẵn: 12
Press any key to continue . . .
```



```
G:\side\Noidung6_1(chan_le).exe
Nhập vào một số = 31
Số nhập vào là số lẻ: 31
Press any key to continue . . .
```

Ví dụ 2: Số lớn nhất từ số đã cho

```
#include<iostream>  #include<stdlib.h>
using namespace std;
int main() {
    int a, b, c; // float, double
    int max;
    cout<<"Nhap so thu nhat a = ";
    cin>>a;
    cout<<"Nhap so thu hai b = ";
    cin>>b;
    cout<<"Nhap so thu ba c = ";
    cin>>c;
    max = (a>b)?a:b; // Toán tử điều kiện tổ hợp bộ 3 (?)
    if (c>max)
        max = c;
    cout<<"Gia tri lon nhat max = "<<max<<endl;
    return 0;
}
```

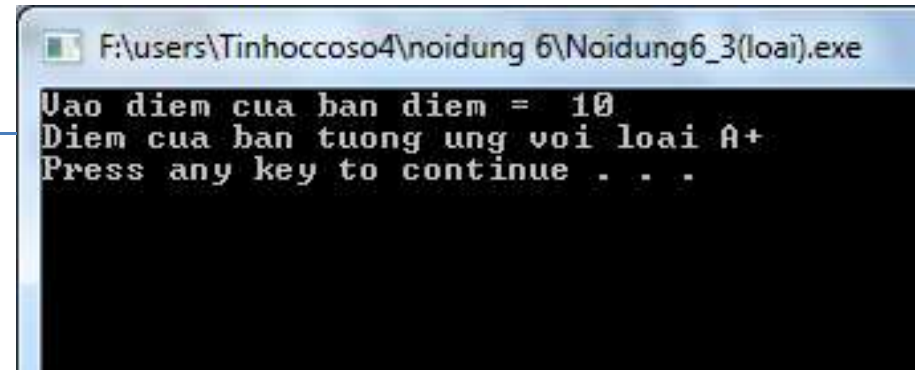


The screenshot shows a Windows command prompt window with the title bar "F:\users\Tinhoccoso4\noidung 6\Noidung6_2(max)". The program's output is as follows:

```
Nhap so thu nhat a = 12
Nhap so thu hai b = 56
Nhap so thu ba c = 30
Gia tri lon nhat max = 56
Press any key to continue . . . _
```

Ví dụ 3. Điểm - loại A, B...

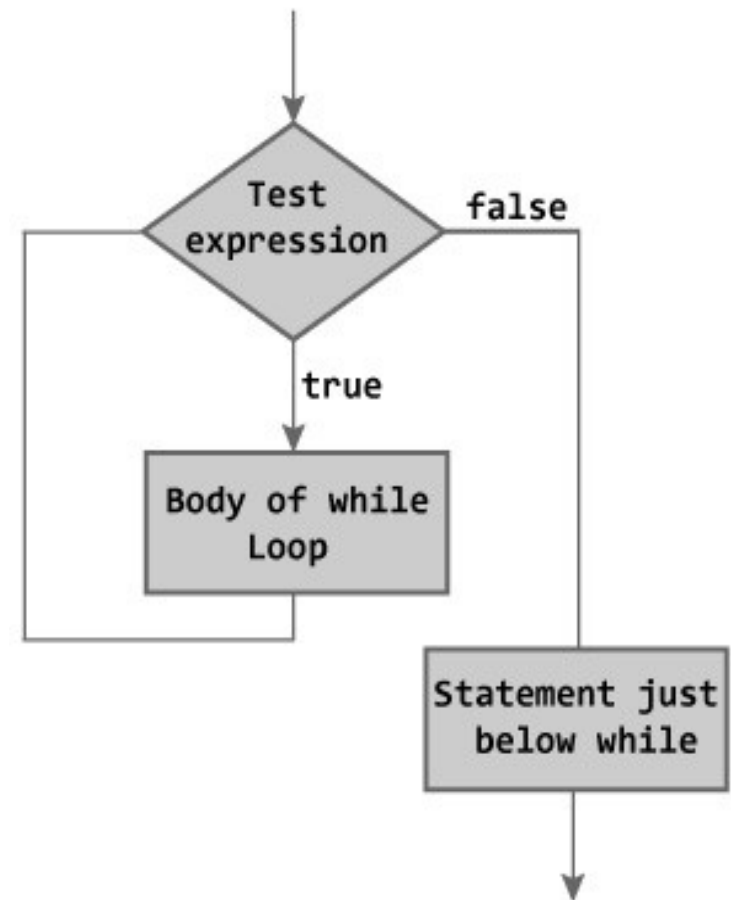
```
int main() {  
    float diem;  
    cout<<"Vao diem cua ban diem = "; cin>>diem;  
    if(diem>=9)  
        cout<<"Diem cua ban tuong ung voi loai A+ \n";  
    else if(diem>=8.5 && diem<=8.9)  
        cout<<"Diem cua ban tuong ung voi loai A \n";  
    else if(diem>=8.0 && diem<=8.4)  
        cout<<"Diem cua ban tuong ung voi loai B+ \n";  
    else if(diem>=7.0 && diem<=7.9)  
        cout<<"Diem cua ban tuong ung voi loai B \n";  
    else if(diem>=6.5 && diem<=6.9)  
        cout<<"Diem cua ban tuong ung voi loai C+ \n";  
    else if(diem>=5.5 && diem<=6.4)  
        cout<<"Diem cua ban tuong ung voi loai C \n";  
    else if(diem>=5.0 && diem<=5.4)  
        cout<<"Diem cua ban tuong ung voi loai D+ \n";  
    else if(diem>=4.0 && diem<=4.9)  
        cout<<"Diem cua ban tuong ung voi loai D \n";  
    else if(diem>=0 && diem<=3.9) {  
        cout<<"\a"<<"\007"; // hang ky tu dac biet tieng bell  
        cout<<"Diem cua ban tuong ung voi loai F - KHONG DAT \n"; }  
}
```



```
F:\users\Tinhocoso4\noidung 6\Noidung6_3(loai).exe  
Vao diem cua ban diem = 10  
Diem cua ban tuong ung voi loai A+  
Press any key to continue . . .
```


2. Vòng lặp while (while loop)

- **Cú pháp:**
`while (expression) statement`
- **Cách hoạt động:**
- Lặp **statement** khi biểu thức **expression** là true.
- Vòng lặp sẽ kết thúc nếu **expression** không đúng nữa (false).
- Như vậy, kiểm tra điều kiện trước khi thực thi lệnh (**statement**).



2. Vòng lặp while (while loop) – Ví dụ: đếm ngược

```
// custom countdown using while
int main () {
    int n = 10;
    while (n > 0) {
        cout << n << ", ";
        --n; // n = n - 1;
        m++; // m = m+1
    }
    cout << "Ket thuc vong while\n";
}
```

Output: [Noidung6 6\(while\)](#)

10, 9, 8, 7, 6, 5, 4, 3, 2, 1, Ket thuc vong while!

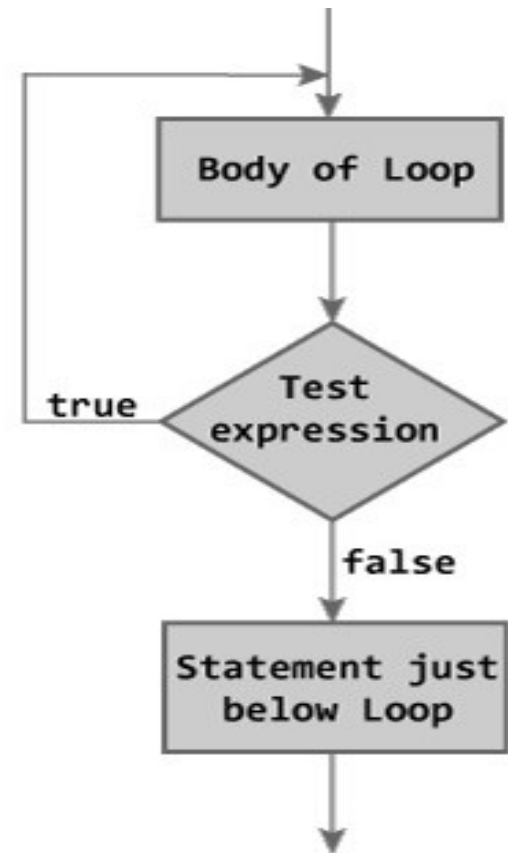
2. Vòng lặp while (while loop)

- Chương trình được thông dịch :
 1. n được gán một giá trị. ($n = 10$)
 2. Điều kiện của while được kiểm tra ($n > 0$). Có hai khả năng
 - + Điều kiện là đúng (true: $n > 0$): câu lệnh được thực thi (chạy tới bước 3)
 - + Điều kiện là sai (false: $n < 0$): câu lệnh được bỏ qua và tiếp tục lệnh sau nó (chạy tới bước 5).
 3. Chạy chương trình:

```
cout << n << ", ";  
--n; // n = n - 1;
```
 4. Kết thúc khối lệnh while và tự động trở lại bước 2.
 5. Chương trình tiếp tục ngay sau khối lệnh lặp
in: Ket thuc vong while! Và kết thúc chương trình

3. Vòng lặp do-while

- Cú pháp: **do statement while (condition);**
- Ứng xử như vòng lặp while ngoại trừ điều kiện (**condition**) được tính sau khi chạy **statement** trước.
- Có ít nhất một **statement** được tính, ngay cả khi **condition** là false.
- Ví dụ: chương trình **lặp lại** yêu cầu người sử dụng nhập tên **đến khi** người sử dụng vào đúng dòng chữ **goodbye**.



3. Vòng lặp do-while

```
// echo machine
#include <iostream>
#include <string>
using namespace std;

int main ()
{
    string str;
    do {
        cout << "Enter text: ";
        getline (cin,str); // cin>> str;
        cout << "You entered: " << str << '\n';
    } while (str != "goodbye");
    return 0;
} Noidung6\(do\)
```

- Output

```
-----
-----

Enter text: hello
You entered: hello
Enter text: who's there?
You entered: who's there?
Enter text: goodbye
You entered: goodbye
```

Vòng lặp do-while

- Vòng lặp **do – while** được ưa dùng hơn vòng lặp **while** khi **statement** (câu lệnh) được tính ít nhất một lần, chẳng hạn khi điều kiện được kiểm tra cuối vòng lặp được xác định, trong khi câu lệnh lặp chính nó.
- Trong ví dụ trên người sử dụng vào trong khối lệnh là những gì sẽ xác định nếu vòng lặp kết thúc. Và do vậy, người sử dụng muốn kết thúc vòng lặp sớm bằng cách nhập vào goodbye.
- Khối trong vòng lặp cần chạy ít nhất một lần lời nhắc đầu vào, và điều kiện có thể chỉ được xác định sau khi chạy.

Ví dụ : lệnh bắt phím sử dụng getch()-c - #include<conio.h>

```
char k;  
do  
{  
    //your code here  
    .....  
    cout<<"Nhấn phím ESC để kết thúc";  
    k = getch(); //cin>>k; - khác nhau  
} while(k != 27); // điều kiện dùng chương trình khi ấn phím ESC
```

- Ví dụ: [mauESC](#)

Ví dụ mẫu 2 – toupper(ch)

- Chúng ta thực hiện công việc khi nào chúng ta chọn phím khác chữ y, hoặc Y.

```
char ch;  
do  
{  
    //your code here  
    .....  
    cout<<"\nBan co muon thu lai khong [Y/N]: ";  
    cin>>ch;  
}while(toupper(ch) == 'Y');
```

Ví dụ: mauYN

Hàm bắt phím: cú pháp `int kbhit(void);`

- Hàm này sẵn có trong : `#include<conio.h>` hoặc `#include<graphics.h>`
- Hàm này trả về (true) nếu có sẵn một ký tự đầu vào bộ đệm (buffer). Trái lại sẽ (false) .

```
for (int i = 0; !kbhit( ); i++)  
{  
    cout<<"Hello functio kbhit \n";  
    // delay(4000);  
}
```

- Hàm này in ra dòng chữ,
và chỉ dừng, thoát khi ấn một phím bất kỳ

A screenshot of a Windows command prompt window titled "E:\tincoso4\side\ASCII_file.exe". The window displays a series of 20 lines, each containing the text "Hello functio kbhit". At the bottom of the window, the text "Press any key to continue . . ." is visible, indicating that the program is waiting for a key press to terminate. The text is displayed in a monospaced font on a black background.

Ví dụ: hàm kbhit()

```
#include <conio.h>
int main() {
    while(1)
    {
        if(kbhit()) {
            break;
        }
    }
}
```



- Khi ấn một phím bất kỳ trên bàn phím, hàm kbhit() trả về giá trị true.

Bảng mã ASCII

- Trường hợp 1:

Cho số thứ tự: 65

Thì ký tự là: char(65).

- Trường hợp 2:

Cho ký tự: z

Thì số thứ tự là: int(z);

Ascii	Char	Ascii	Char	Ascii	Char	Ascii	Char
0	Null	32	Space	64	@	96	`
1	Start of heading	33	!	65	A	97	a
2	Start of text	34	"	66	B	98	b
3	End of text	35	#	67	C	99	c
4	End of transmit	36	\$	68	D	100	d
5	Enquiry	37	%	69	E	101	e
6	Acknowledge	38	&	70	F	102	f
7	Audible bell	39	'	71	G	103	g
8	Backspace	40	(72	H	104	h
9	Horizontal tab	41)	73	I	105	i
10	Line feed	42	*	74	J	106	j
11	Vertical tab	43	+	75	K	107	k
12	Form feed	44	,	76	L	108	l
13	Carriage return	45	-	77	M	109	m
14	Shift in	46	.	78	N	110	n
15	Shift out	47	/	79	O	111	o
16	Data link escape	48	0	80	P	112	p
17	Device control 1	49	1	81	Q	113	q
18	Device control 2	50	2	82	R	114	r
19	Device control 3	51	3	83	S	115	s
20	Device control 4	52	4	84	T	116	t
21	Neg. acknowledge	53	5	85	U	117	u
22	Synchronous idle	54	6	86	V	118	v
23	End trans. block	55	7	87	W	119	w
		56	8	88	X	120	x
		57	9	89	Y	121	y
		58	:	90	Z	122	z
		59	;	91	[123	{
		60	<	92	\	124	
		61	=	93]	125	}
		62	>	94	^	126	~
		63	?	95	_	127	Forward

```

E:\tincoso4\side\Bai tap\tuan 3\char.exe
Chương trình chuyển số thứ tự trong bảng mã ASCII thành ký tự
STT 65 - 90 = A - Z, STT 97 - 121 = a - z

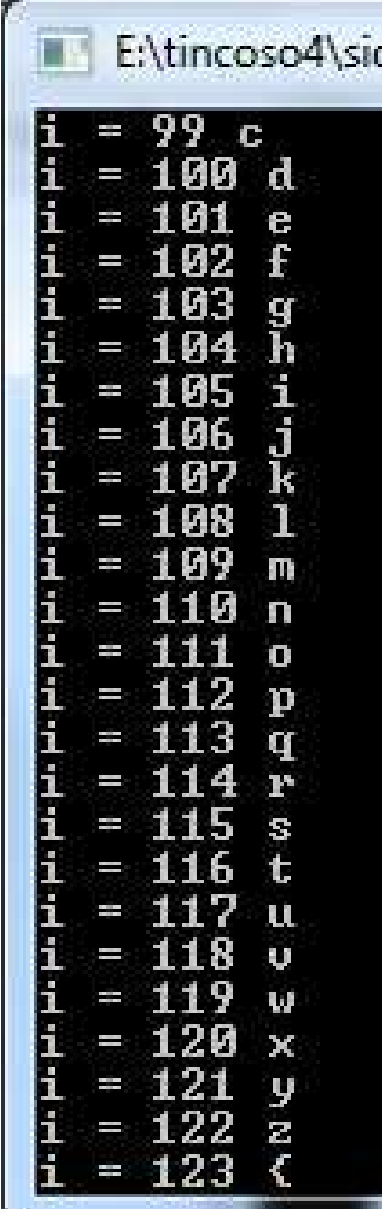
Nhập số thứ tự trong bảng mã ASCII: 65
Ký tự tương ứng với số thứ tự 65 là: A

Nhập ký tự trong bảng mã ASCII: z
Số thứ tự tương ứng với ký tự z là: 122
Press any key to continue . . .
  
```

Vấn đề: ấn phím theo ý ?

```
char ch;
while (1) {
    if (kbhit) {
        ch = getch(); // Stores the pressed key in ch
        // cin>>ch;
        cout << "\nKey pressed= " << ch;
        // Terminates the loop when escape is pressed
        if (int(ch) == 27)
            break;
        if(int(ch) == 105) // i
            cout<<" phim mui ten len \n ";
        if(int(ch) == 107)// k
            cout<<" phim mui ten xuong \n ";
        if(int(ch) == 108)// l
            cout<<" phim mui ten sang phai \n ";
        if(int(ch) == 106)// j
            cout<<" phim mui ten sang trai \n ";
    }
}
// Có thể sử dụng switch()
```

[mauSudungPhim](#)



```
E:\tincoso4\sic
i = 99 c
i = 100 d
i = 101 e
i = 102 f
i = 103 g
i = 104 h
i = 105 i
i = 106 j
i = 107 k
i = 108 l
i = 109 m
i = 110 n
i = 111 o
i = 112 p
i = 113 q
i = 114 r
i = 115 s
i = 116 t
i = 117 u
i = 118 v
i = 119 w
i = 120 x
i = 121 y
i = 122 z
i = 123 [
```

Ví dụ đọc phím mũi tên

- "When reading a function key or an arrow key, each function must be called twice; the first call returns 0 or 0xE0, and the second call returns the actual key code."
- [http://msdn.microsoft.com/en-us/library/078sfkak\(v=vs.110\).aspx](http://msdn.microsoft.com/en-us/library/078sfkak(v=vs.110).aspx)

unsigned char a;

```
while(1){
```

```
    a = getch();
```

```
    //for detect the function\arrow keys we must call the getch() again
```

```
    if (a == 0 || a == 0xE0) a = getch(); //testing if a is '0' or '0xE0'
```

```
    if (a == 27) //ESC for exit the 'while'
```

```
        break;
```

```
    else if (a == 72)
```

```
        cout<<"UP \n";
```

```
    else if (a == 80)
```

```
        cout<<"DOWN \n";
```

```
    else if (a == 75)
```

```
        cout<<"LEFT \n";
```

```
    else if (a == 77)
```

```
        cout<<"RIGHT \n";
```

```
    else
```

```
        cout<<int(a);
```

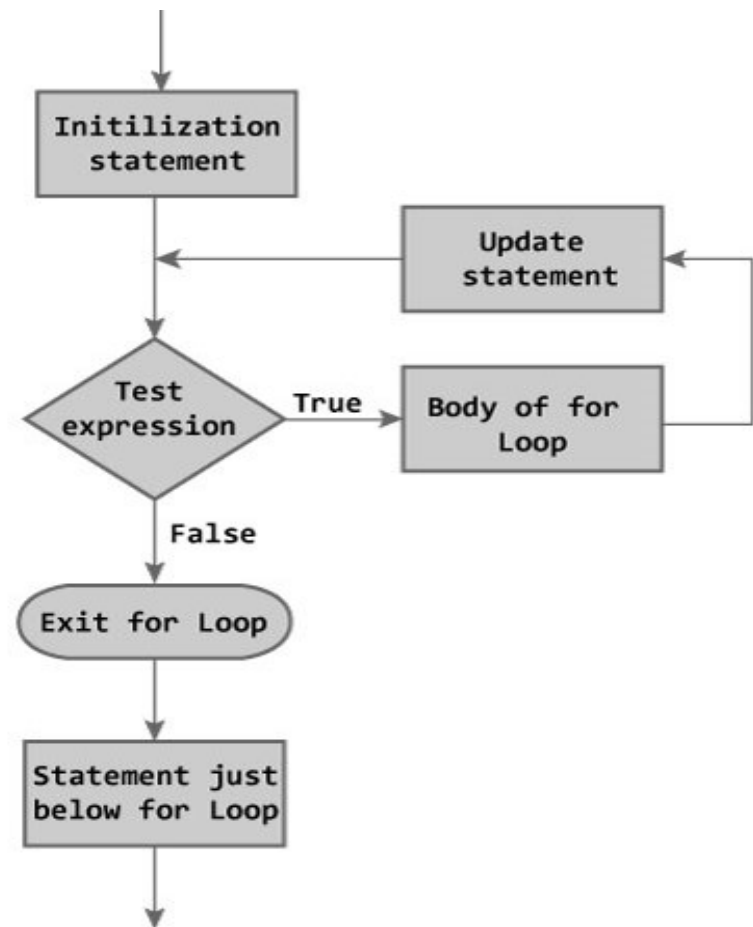
```
} mauSudungMuiTen
```



```
getchar() = cin.get() // c++
```

4. Vòng lặp for

- Thiết kế lặp một số lần hữu hạn.
- Cú pháp:** for (initialization; **condition**; increase or decrease)
{
 statement; // codes to process
}
- Vòng lặp:** lặp đi lặp lại **statement** khi điều kiện (**condition**) đúng (true).
- Ví dụ:**
 //for (initialization; condition; decrease)
 for (int n = 10; n>0; n--)
 { // có dấu;
 cout << n << ", ";
 }



4. Vòng lặp for

- Làm việc theo cách sau
 1. **initialization** được thực thi. Đây là khai báo biến đếm, và đặt cho nó giá trị ban đầu, tại thời điểm bắt đầu của vòng lặp.
 2. **condition** được kiểm tra. Nếu true thì vòng lặp tiếp tục, trái lại vòng lặp kết thúc, và statement bị bỏ qua và đi đến bước 5.
 3. **statement** được thực thi, nó là một dòng đơn hoặc là một khối đặt trong ngoặc {}.
 4. **increase** được thực thi, và vòng lặp trở lại bước 2.
 5. Vòng lặp kết thúc: chạy tiếp tục bởi câu lệnh tiếp theo sau vòng lặp.

4. Vòng lặp for

```
// Đếm ngược sử dụng vòng for
#include <iostream>
using namespace std;

int main ()
{
    for (int n = 10; n>0; n--)
    {
        cout << n << ", ";
    }
    cout << "Ket thuc vong for!\n";
    return 0;
}
```

Output:

10, 9, 8, 7, 6, 5, 4, 3, 2, 1,
Ket thuc vong for!

4. Vòng lặp for

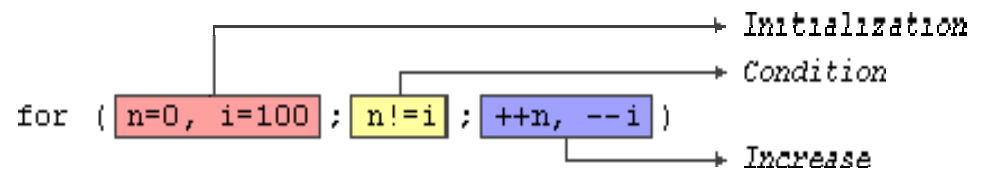
- Ba trường trong vòng lặp for là: **lựa chọn** (tức là có thể có hoặc không).
- Chúng có thể để trống, **nhưng** toàn bộ dấu **chấm phẩy** phải giữ nguyên.
- Ví dụ: `for (; n<10;)`, là một vòng lặp không có khởi đầu và tăng biến (tương đương với vòng lặp **while**),
- và `for (;n<10;n++)` là: vòng lặp với tăng (increase), nhưng không có khởi đầu.
- Vòng lặp không có điều kiện (condition) là tương đương với một vòng lặp với điều kiện là true (nghĩa là vòng lặp vô hạn).
- Ví dụ: `for (; ;)`

4. Vòng lặp for

- Vì mỗi một trường được chạy ở một thời điểm nhất định trong chu kỳ của vòng lặp. Do đó, có thể chạy nhiều hơn biểu thức đơn ở các trường : (initialization, condition, increase).
- Trong trường hợp này, chúng không phải là câu lệnh, đơn giản là biểu thức, và không thể thay thế bằng một khối {}.
- Khi là biểu thức, chúng có thể sử dụng toán tử dấu phẩy (,). Đây là toán tử phân cách biểu thức, và có thể tách riêng nhiều biểu thức.
- Ví dụ: vòng lặp for kết nối với hai biến đếm, trong cả trường khởi đầu và trường tăng.

4. Vòng lặp for

```
for ( n = 0, i = 100 ; n != i ; ++n, --i )  
{  
    // whatever here...  
}
```



- Vòng lặp này sẽ chạy 50 lần nếu như cả `n` và `i` không thay đổi trong vòng lặp
- `n` bắt đầu với giá trị bằng 0, và `i = 100`, điều kiện là `n!=i` (nghĩa là `n` không bằng `i`). Bởi vì `n` tăng 1 và `i` giảm 1 cho mỗi bước lặp, điều kiện của vòng lặp sẽ trở thành false sau 50 bước lặp, khi đó cả `n` và `i` đều bằng 50.

Sử dụng vòng for

- Ví dụ in bảng mã ASCII – và sử dụng một số ký tự đặc biệt
- Biểu diễn:

$$\int F(x)dx, \cos(\alpha x)$$

```
for (int i = 0; i<255; i++) {  
    cout<<"i = "<<i<<" "<<char(i)<<"\n";  
}  
cout<<"\n\n";  
cout<<char(244)<<"\n"<<char(245)<<"F(x)dx\n";  
cout<<"cos("<<char(224)<<"x)\n";
```



```
E:\tincoso4\side  
i = 236 ω  
i = 237 ø  
i = 238 €  
i = 239 ñ  
i = 240 ï  
i = 241 ±  
i = 242 ∑  
i = 243 ≤  
i = 244 ∫  
i = 245 ÷  
i = 246 ≈  
i = 247 °  
i = 248 ·  
i = 249 ·  
i = 250 ·  
i = 251 √  
i = 252 ∞  
i = 253 ∞  
i = 254 ■  
  
F(x)dx  
cos(αx)
```

Ví dụ: tạo bảng cửu chương

```
#include<iostream> #include<stdlib.h> using namespace std;
```

```
int main() {
```

```
    int count;
```

```
    char ch;
```

```
    do {
```

```
        cout<<"Nhap bang cuu chuong may [1-10] : ";
```

```
        cin>>count;
```

```
        if(count > 0 && count <= 10) {
```

```
            for (int i = 0; i<10; i++) {
```

```
                cout<<count<<" x "<<(i + 1)<<" = "<<count*(i+1)<<endl;
```

```
            }
```

```
        }
```

```
        else
```

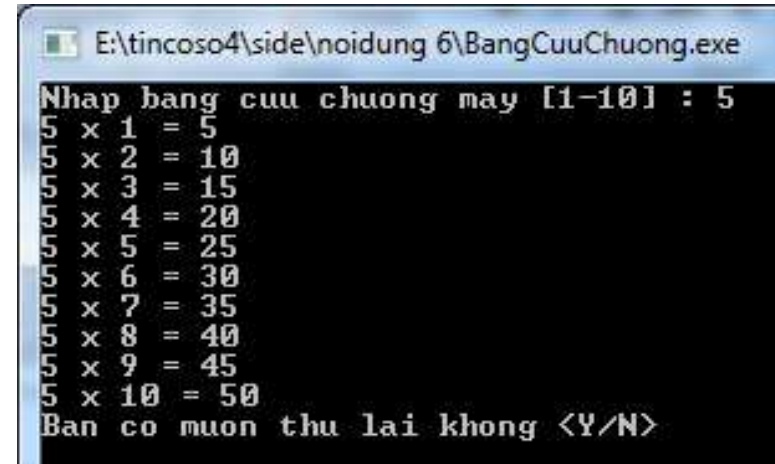
```
            cout<<"Ban nhap lai bang cuu chuong chi [1-10]"<<endl;
```

```
            cout<<"Ban co muon thu lai khong <Y/N> ";
```

```
            cin>>ch;
```

```
            system("cls");
```

```
    }while(toupper(ch) != 'Y'); BangCuuChuong
```



```
E:\tincoso4\side\noidung 6\BangCuuChuong.exe
Nhap bang cuu chuong may [1-10] : 5
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
Ban co muon thu lai khong <Y/N>
```

4. Vòng lặp dải cơ bản for (Range-based for loop)

- Cú pháp khác của vòng lặp for, được sử dụng chạy với một dải
for (declaration : range) statement;
- Đây là kiểu vòng lặp for qua toàn bộ phần tử trong range, trong đó declaration(khai báo) một biến có thể lấy giá trị của một phần tử trong dải này.
- Dải (range) là một chuỗi các phần tử, bao gồm mảng, container, và bất kỳ kiểu khác hỗ trợ chức năng (functions) bắt đầu (begin) và kết thúc (end).
- Hầu hết kiểu trên không được trình bày ở đây, nhưng chúng ta có một kiểu thuộc dải này: string (xâu), là chuỗi các ký tự.
- ```
float x[3] = {1.1, 2, 3.6};
 for (float i : x) { // for (auto i : x)
 cout << i << '\n' ; // x[0], x[1], x[2]
 } // In ra: 1.1, 2, 3.6
```

## 4. Vòng lặp dải cơ bản for

```
#include <iostream>
#include <string>
using namespace std;
int main ()
{
 string str {"Hello!"};
 for (char c : str)
 {
 cout << "[" << c << " ";
 }
 cout << '\n';
}
```

Output

-----  
[H][e][l][l][o][!]

- Chú ý. Đứng trước (:) trong vòng lặp for là khai báo biến kiểu char (toàn bộ phần tử trong string (xâu) là kiểu char).
- Khi đó sử dụng biến c, trong khối câu lệnh để biểu diễn **giá trị** của mỗi phần tử trong dải.
- Vòng lặp tự động và không yêu cầu khai báo biến của bất kỳ biến đếm nào
- Dải dựa trên vòng lặp được sử dụng làm kiểu suy diễn cho các kiểu của phần tử với *auto*, vậy dải trên được viết như sau  
for (auto c : str)  
 cout << "[" << c << " ";
- Kiểu của c được suy diễn tự động như kiểu của các phần tử trong str.



## 5. Câu lệnh nhảy : break, continue, goto

---

- Câu lệnh “nhảy” cho phép biến đổi mạch của chương trình thực hiện bởi bước nhảy tới vị trí xác định.

### a) Câu lệnh break - The break statement

- Cú pháp: break;
- Mục đích: thoát khỏi vòng lặp, ngay cả điều kiện kết thúc của nó không được thực thi.
- Được sử dụng để kết thúc một vòng lặp vô hạn, hoặc là ép vòng lặp này kết thúc trước khi nó kết thúc tự nhiên.
- Ví dụ, hãy dừng đếm ngược trước khi nó kết thúc tự nhiên.

## 5a.Câu lệnh break - The break statement

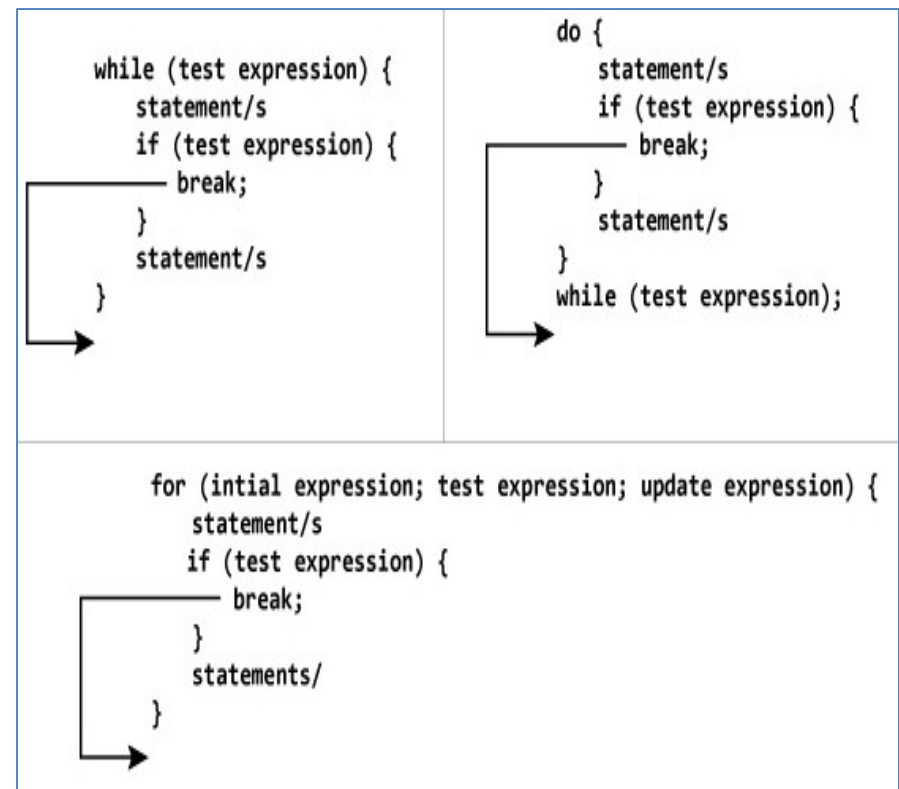
### // break loop example

```
int main ()
{
 for (int n = 10; n>0; n--)
 {
 cout << n << ", ";
 if (n == 3)
 {
 cout << "Dung dem nguoc!";
 break;
 }
 }
}
```

Output

-----  
10, 9, 8, 7, 6, 5, 4, 3,  
Dung dem nguoc!

### How break statement works?



## 5b. Câu lệnh continue -The continue statement

- **Cú pháp:** continue;
- Câu lệnh continue: làm cho chương trình bỏ qua phần còn lại của vòng lặp tại bước lặp hiện tại, làm cho nó nhảy tới vòng lặp tiếp theo.
- Ví dụ: bỏ số 5 trong chương trình đếm ngược

```
while (test expression) {
 statement/s
 if (test expression) {
 continue;
 }
 statement/s
}
```

```
do {
 statement/s
 if (test expression) {
 continue;
 }
 statement/s
}
while (test expression);
```

```
for (initial expression; test expression; update expression) {
 statement/s
 if (test expression) {
 continue;
 }
 statements/
}
```

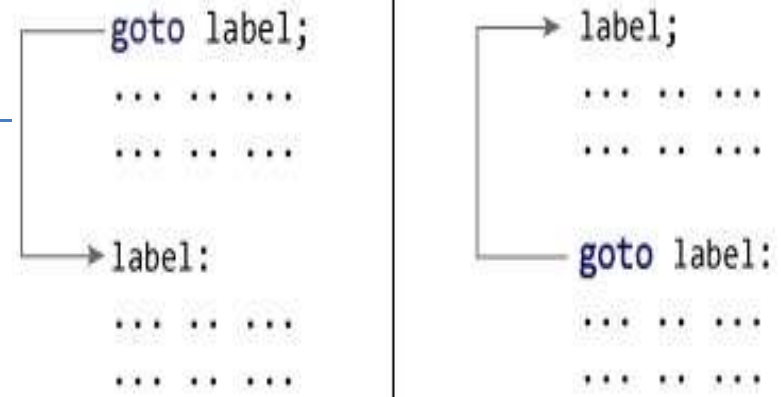
```
// continue loop example
int main () {
 for (int n = 10; n>0; n--) {
 if (n == 5) continue;
 cout << n << ", ";
 }
 cout << "liftoff!\n";
}
```

Output:

10, 9, 8, 7, 6, 4, 3, 2, 1,

## 5c.Câu lệnh goto

- **goto**: cho phép làm một bước nhảy thuần túy tới một điểm khác trong chương trình....
- **Điểm đến**: được định danh bởi một nhãn (label), được sử dụng như một đối cho câu lệnh goto.
- **Nhãn (label) là**: định danh hợp lệ theo sau bởi dấu hai chấm (:)
- goto là: mức thấp, không được sử dụng trong lập trình cấp cao, mô hình C++.
- Ví dụ: đếm ngược



```
// goto loop example
#include <iostream>
using namespace std;
int main ()
{
 int n = 10;
 mylabel:
 cout << n << ", ";
 n--;
 if (n > 0) goto mylabel;
 cout << "liftoff!\n";
}
10, 9, 8, 7, 6, 5, 4, 3, 2, 1, liftoff!
```

## 5c.Câu lệnh goto

---

```
int main()
{
float num, average, sum = 0.0; int i, n;
cout << "Maximum number of inputs: ";
cin >> n;
for(i = 1; i <= n; ++i) {
 cout << "Enter n" << i << ": ";
 cin >> num;
 if(num < 0.0)
 { // Control of the program move to jump:
 goto jump;
 }
 sum += num;
}
jump: average = sum / (i - 1);
cout << "\nAverage = " << average; return 0;
}
```

## 6. Câu lựa chọn switch

- Mục đích: kiểm tra một giá trị trong số biểu thức hằng. Nó tương tự câu lệnh if – else, nhưng giới hạn cho biểu thức hằng.
- Cú pháp:

```
switch (expression)
```

```
{
```

```
 case constant1:
```

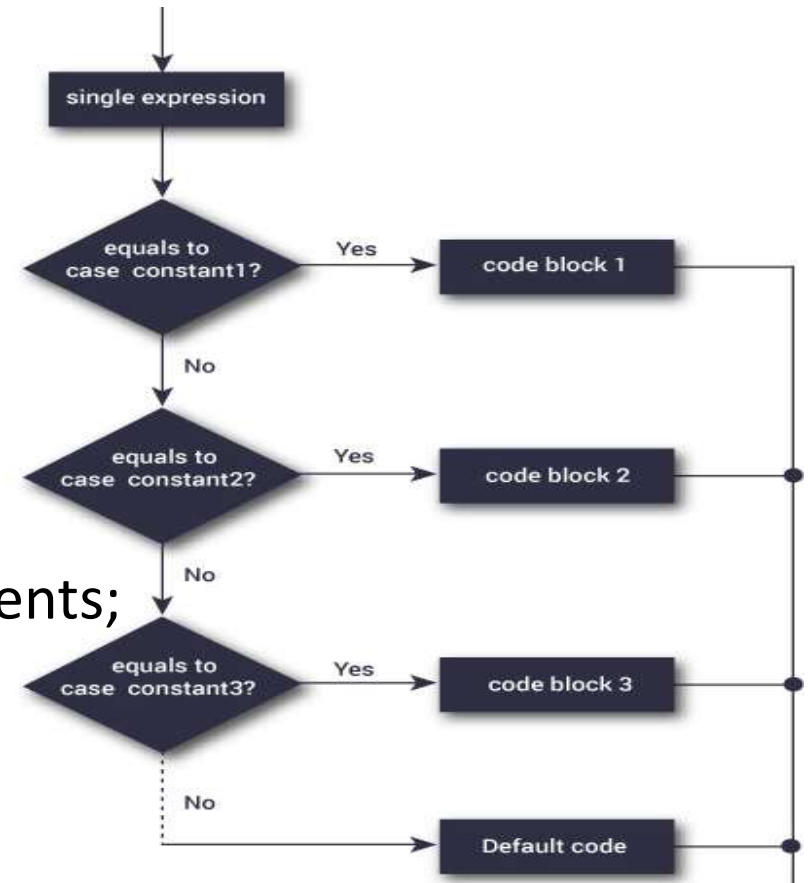
```
 group-of-statements-1; break;
```

```
 case constant2:
```

```
 group-of-statements-2; break;
```

```
 default: default-group-of-statements;
```

```
}
```



## 6. Câu lựa chọn switch

---

Minh họa câu lệnh: if – else tương đương với switch. Hai đoạn mã sau có ứng xử tương đương.

### switch example

```
switch (x) {
 case 1:
 cout << "x is 1";
 break;
 case 2:
 cout << "x is 2";
 break;
 default:
 cout << "value of x unknown";
}
```

### if-else equivalent

```
if (x == 1) {
 cout << "x is 1";
}
else if (x == 2) {
 cout << "x is 2";
}
else {
 cout << "value of x unknown";
}
```

## 6. Câu lựa chọn switch

---

- Nếu thiếu lệnh **break**: sau nhóm đầu tiên **case 1** thì chương trình không nhảy tự động đến cuối của câu lệnh switch sau khi in x bằng 1 và thay vào đó là **tiếp tục** chạy **case 2**, cũng in x bằng 2 và cứ làm như thế đến khi bắt gặp câu lệnh break hoặc kết thúc khối switch.
- Do đó, không cần phải kẹp kín các câu lệnh trong mỗi case bằng ngoặc {}.
- Switch: hữu ích để chạy một nhóm câu lệnh với giá trị có thể khác nhau.



## 6. Câu lựa chọn switch

- Ví dụ: (có **break** – không có **break**)

```
switch (x) {
```

```
 case 1:
```

```
 case 2:
```

```
 case 3:
```

```
 cout<<"Hello case 3"
```

```
 cout << "x is 3";
```

```
 break;
```

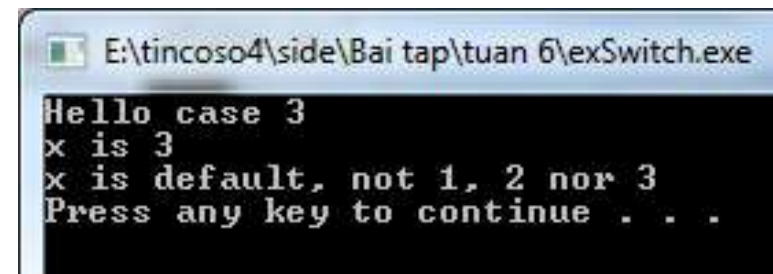
```
 default:
```

```
 cout << "x is default, not 1, 2 nor 3";
```

```
}
```



```
E:\tincoso4\side\Bai tap\tuan 6\exSwitch.exe
Hello case 3
x is 3
Press any key to continue . . . _
```

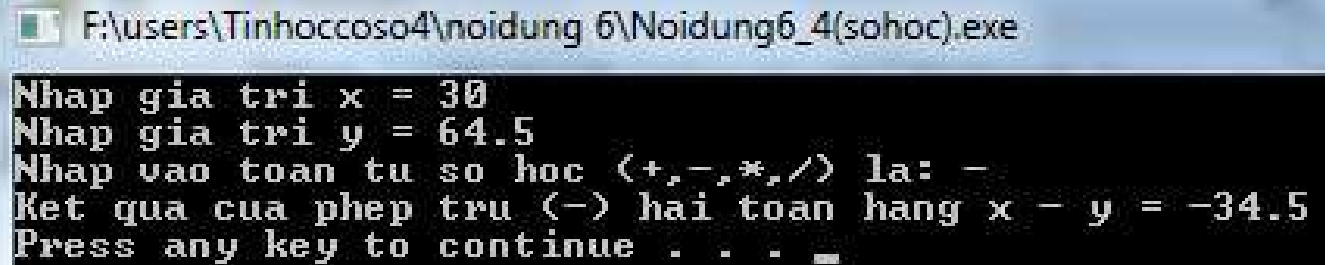


```
E:\tincoso4\side\Bai tap\tuan 6\exSwitch.exe
Hello case 3
x is 3
x is default, not 1, 2 nor 3
Press any key to continue . . .
```

## Ví dụ : sử dụng phép toán số học

```
int main(){
 float x,y, ketqua;
 char z;
 cout<<"Nhap gia tri x = ";
 cout<<"Nhap gia tri y = ";
 cout<<"Nhap vao toan tu so hoc (+,-,*,/) la: ";
 cin>>z;
 switch(z) {
 case '+':
 ketqua = x + y;
 cout<<"Ket qua cua phep cong (+) hai toan hang x + y = "<<ketqua<<endl;
 break;
 case '-':
 ketqua = x - y;
 cout<<"Ket qua cua phep tru (-) hai toan hang x - y = "<<ketqua<<endl;
 break;
 default: cout<<"Ban nhap vao khong dung"<<endl;
 }
}
```

[Noidung6\\_4\(sohoc\)](#)



```
F:\users\Tinhoccoso4\noidung 6\Noidung6_4(sohoc).exe
Nhap gia tri x = 30
Nhap gia tri y = 64.5
Nhap vao toan tu so hoc (+,-,*,/) la: -
Ket qua cua phep tru (-) hai toan hang x - y = -34.5
Press any key to continue . . . _
```

## Ví dụ : đoán nguyên âm

```
#include<iostream> , #include<stdlib.h> using namespace std;
```

```
int main() {
```

```
 char y;
```

```
 cout <<"Nhap mot ky tu bat ky: ";
```

```
 cin >>y;
```

```
 switch(y) {
```

```
 case 'u':
```

```
 cout<<"Day la nguyen am \n"; break;
```

```
 case 'e':
```

```
 cout<<"Day la nguyen am \n"; break;
```

```
 case 'o':
```

```
 cout<<"Day la nguyen am \n"; break;
```

```
 case 'a':
```

```
 cout<<"Day la nguyen am \n"; break;
```

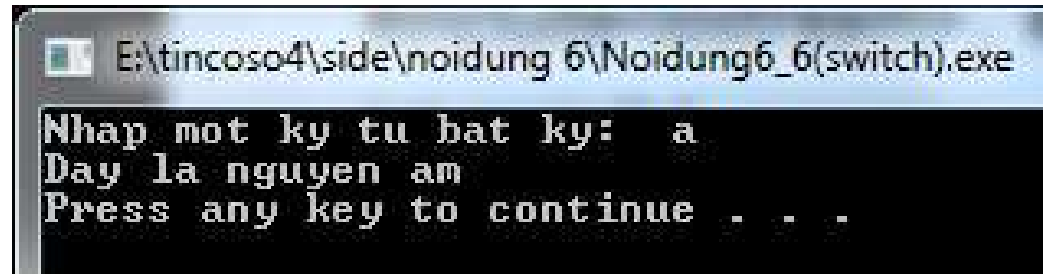
```
 case 'i':
```

```
 cout<<"Day la nguyen am \n"; break;
```

```
 default: cout<<"Khong phai la nguyen am \n";
```

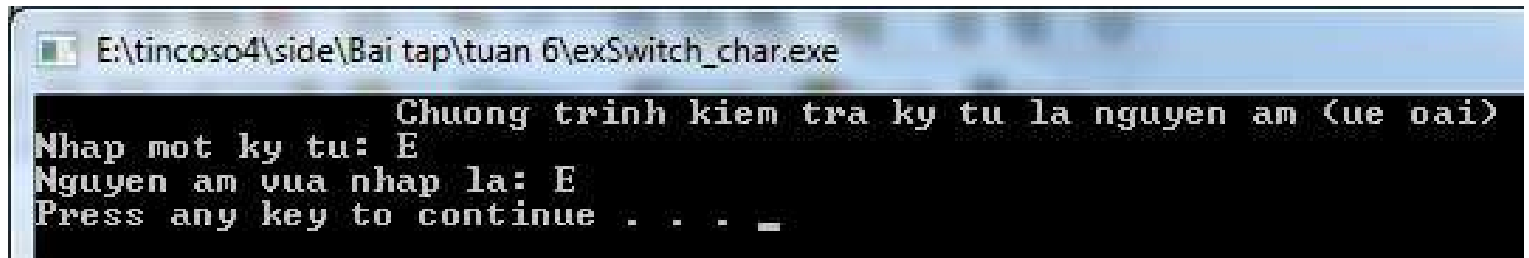
```
 }
```

```
} Noidung6_6\(switch\)
```



## Ví dụ : Đoán nguyên âm trong chuỗi ký tự

```
char ch;
cout<<"Nhập một ký tự: ";
cin>>ch;
switch(ch) {
 case 'a':
 case 'A':
 case 'e':
 case 'E':
 case 'i':
 case 'I':
 case 'o':
 case 'O':
 case 'u':
 case 'U':
 cout<<"Nguyên âm vừa nhập là: "<<ch<<endl;
 break;
 default:
 cout<<"Ký tự vừa nhập không phải là nguyên âm!"<<endl;
}
```



```
E:\tincoso4\side\Bai tap\tuan 6\exSwitch_char.exe
Chương trình kiểm tra ký tự là nguyên âm (ue oai)
Nhập một ký tự: E
Nguyên âm vừa nhập là: E
Press any key to continue . . . _
```

## Tổng kết

---

- Cấu trúc điều khiển gồm:
  - ✓ 1. Cấu trúc rẽ nhánh: có điều kiện if, if ..else, if...else if, switch.
  - ✓ 2. Cấu trúc vòng lặp: while, do ... while, for, goto
  - ✓ 3. Các từ khóa break, continue.

## Các hàm làm tròn số trong C++

- **Hàm round(x)**
- Làm tròn về số nguyên gần nhất so với số thực x.
- **Hàm trunc(x)**
- Trả về số thực có giá trị bằng phần nguyên của x.
- **Hàm ceil(x)**
- Làm tròn lên số thực x. Trả về số thực có giá trị bằng số nguyên nhỏ nhất lớn hơn hoặc bằng x.
- **Hàm floor(x)**
- Làm tròn xuống số thực x. Trả về số thực có giá trị bằng số nguyên lớn nhất nhỏ hơn hoặc bằng x.
- **Chú ý:** Tất cả các hàm trên đều thuộc thư viện math.

| Giá trị | round | floor | ceil  | trunc |
|---------|-------|-------|-------|-------|
| -----   | ----- | ----- | ----- | ----- |
| 2.3     | 2.0   | 2.0   | 3.0   | 2.0   |
| 3.8     | 4.0   | 3.0   | 4.0   | 3.0   |
| 5.5     | 6.0   | 5.0   | 6.0   | 5.0   |
| -2.3    | -2.0  | -3.0  | -2.0  | -2.0  |
| -3.8    | -4.0  | -4.0  | -3.0  | -3.0  |
| -5.5    | -6.0  | -6.0  | -5.0  | -5.0  |

## Enum – kiểu liệt kê

---

- Một số hằng số không liên quan đến nhau nhưng được khai báo gần nhau khiến chúng ta dễ rối.
- Định nghĩa: enum là kiểu người sử dụng định nghĩa, là tập các tên hằng.
- **Cú pháp khai báo kiểu liệt kê**
- **enum** <name\_of\_enumeration>

```
{ //Danh sách toàn bộ giá trị ở bên trong khối này
 // mỗi thành phần cách nhau bởi dấu phẩy, không là ;
};
```

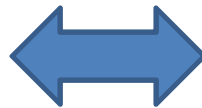
Ví dụ: enum Color {RED, GREEN, BLUE} myColor;  
enum Color {RED = 1, GREEN = 5, BLUE};  
Color myColor;

- **enum name : type { enumerator = constexpr , enumerator = constexpr , ... } C++11.**

## Enum – kiểu liệt kê

```
int main() {
 const float PI = 3.14;
 const float ACCELERATION_OF_GRAVITY = 9.8;
 float s, r;
 cout<<"Nhap ban kinh duong tron [m], r = ";
 cin>>r;
 s = PI*r*r;
 cout<<"Dien tich hinh tron la s = "<<s<<" [m^2]"<<endl;
}
```

```
const int SUNDAY = 0;
const int MONDAY = 1;
const int TUESDAY = 2;
const int WEDNESDAY = 3;
const int THURSDAY = 4;
const int FRIDAY = 5;
const int SATURDAY = 6;
```



```
enum DaysOfWeek {
 SUNDAY,
 MONDAY,
 TUESDAY,
 WEDNESDAY,
 THURSDAY,
 FRIDAY,
 SATURDAY
};
```



## Enum – kiểu liệt kê

- Chú ý:
- Không cần trực tiếp gán giá trị cho các tên hằng số, **compiler** đã tự động khởi tạo giá trị cho chúng, bắt đầu với giá trị **0** và tăng dần. (SUNDAY = 0) và SATURDAY = 6.
- Biến chỉ có thể lấy các tên của sự khai báo.
- Ví dụ: [enumExample.cpp](#)

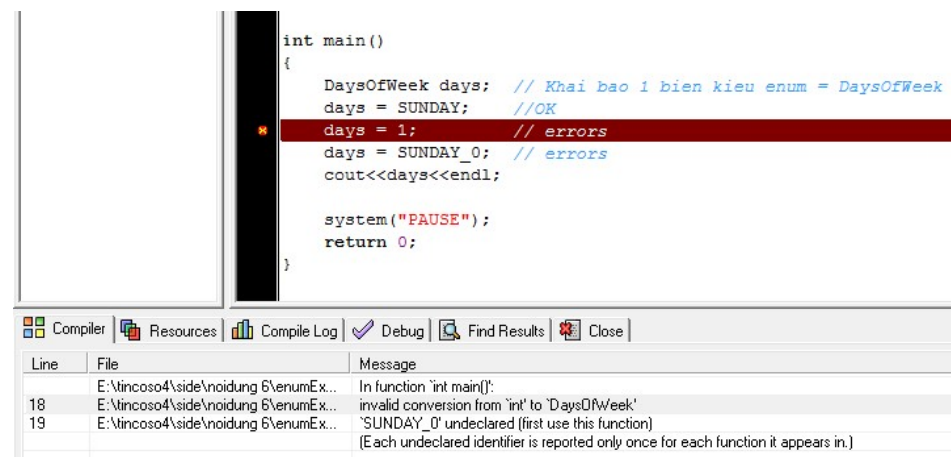
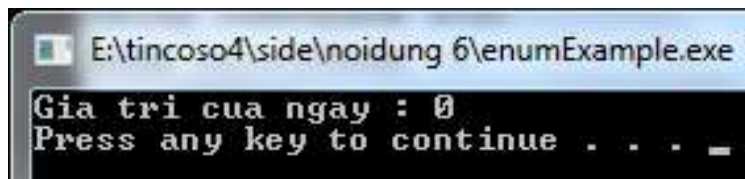
```
DaysOfWeek days; // Khai báo 1 biến kiểu enum = DaysOfWeek
```

```
days = SUNDAY; //OK
```

```
days = 1; // errors
```

```
days = SUNDAY_0; // errors
```

```
cout<<days<<endl;
```



- Một enum sau khi đã định nghĩa xong thì **không** thể thay đổi những giá trị của danh sách các phần tử nữa.
- Biến kiểu enum chỉ có thể được gán giá trị là một trong số các hằng đã khai báo bên trong kiểu dữ liệu của chính nó, không thể sử dụng hằng của kiểu enum khác.

- **Ví dụ: enum Direction {**

UP = 1, //assigned 1 by programmer

DOWN = 3, //assigned 3 by programmer

LEFT, //assigned 4 by compiler

RIGHT //assigned 5 by compiler

};

```
cout << UP << " " << DOWN << " " << LEFT << " " << RIGHT << endl;
```

Output: 1 3 4 5

---

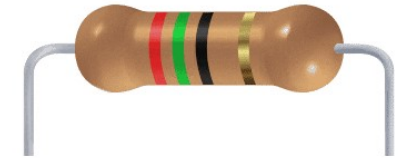
**Compiler** tự động gán giá trị cho các phần tử không được khởi tạo giá trị. Ngoại trừ phần tử đầu tiên trong **enum**, những hằng số khác sẽ được gán giá trị bằng phần tử trước nó cộng thêm 1.

## Emum – kiểu liệt kê

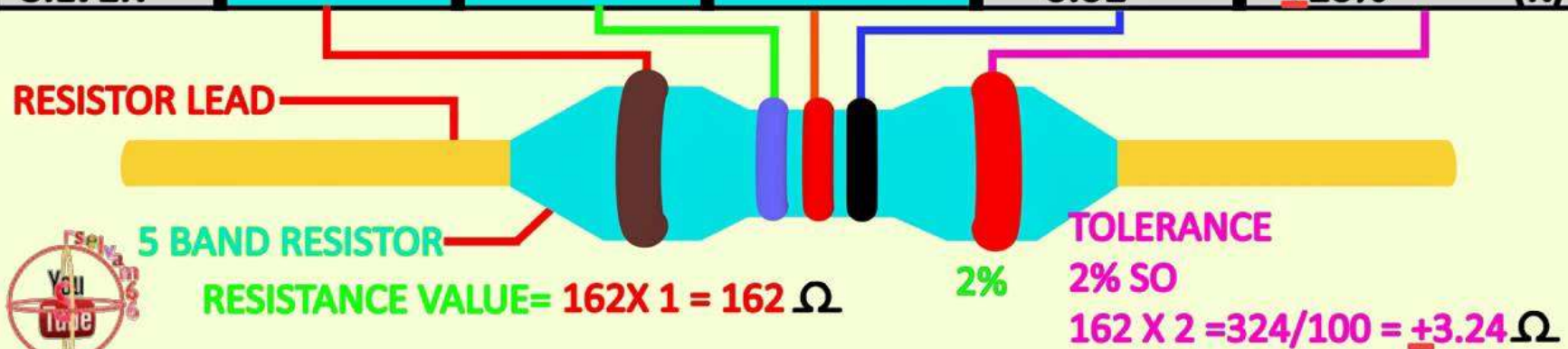
---

```
enum {red, orange, yellow, green, blue, violet, indigo}; // Gia tri 0 1 2 3 4 5 6
int main() {
 cout << "Enter color code (0-6): ";
 int code;
 cin >> code;
 while (code >= red && code <= indigo) { // (code >=0 && code<=6)
 switch (code) {
 case red : cout << "red.\n"; break;
 case orange : cout << "orange.\n"; break;
 case yellow : cout << "yellow.\n"; break;
 case green : cout << "green.\n"; break;
 case blue : cout << "blue.\n"; break;
 case violet : cout << "violet.\n"; break;
 case indigo : cout << "indigo.\n"; break;
 // default : cout<<"Ban nhap code sai chi trong khoang [0-6]"<<endl;
 }
 cout << "Enter color code (0-6): ";
 cin >> code;
 }
} Noidung6_1\(switch\)
```

# Tính điện trở - Bài tập



| COLOR  | 1 <sup>ST</sup> BAND | 2 <sup>ND</sup> BAND | 3 <sup>RD</sup> BAND | MULTIPLIER    | TOLERANCE  |
|--------|----------------------|----------------------|----------------------|---------------|------------|
| BLACK  | 0                    | 0                    | 0                    | 1 $\Omega$    |            |
| BROWN  | 1                    | 1                    | 1                    | 10 $\Omega$   | + 1% (F)   |
| RED    | 2                    | 2                    | 2                    | 100 $\Omega$  | + 2% (G)   |
| ORANGE | 3                    | 3                    | 3                    | 1K $\Omega$   |            |
| YELLOW | 4                    | 4                    | 4                    | 10K $\Omega$  |            |
| GREEN  | 5                    | 5                    | 5                    | 100K $\Omega$ | +0.5% (D)  |
| BLUE   | 6                    | 6                    | 6                    | 1M $\Omega$   | +0.25% (C) |
| VIOLET | 7                    | 7                    | 7                    | 10M $\Omega$  | +0.10% (B) |
| GREY   | 8                    | 8                    | 8                    |               | +0.05%     |
| WHITE  | 9                    | 9                    | 9                    |               |            |
| GOLD   |                      |                      |                      | 0.1           | +5% (J)    |
| SILVER |                      |                      |                      | 0.01          | +10% (K)   |



# Vẽ màu điện trở

- enum colors1 {BLACK = 0, BLUE, GREEN,CYAN,RED, MAGENTA, BROWN, LIGHTGRAY, DARKGRAY, LIGHTBLUE,LIGHTGREEN,LIGHTCYAN,LIGHTRED, LIGHTMAGENTA, YELLOW,WHITE};



- Điện trở

