BUỔI 1

Link tải DEV C++: https://sourceforge.net/projects/orwelldevcpp/

I. Mục đích của lập trình C

1. **Xây dựng các nền tảng tính toán.**
   a. Luyện tư duy logic.
   b. Luyện cách fix bug.
2. **Thiết kế phần mềm hệ thống**.
   a. Lập trình nhúng.
   b. Lập trình game.
   c. Lập trình hệ điều hành.
3. **Phát triển ngôn ngữ mới**.
   a. Dễ dàng hơn khi học các ngôn ngữ khác.

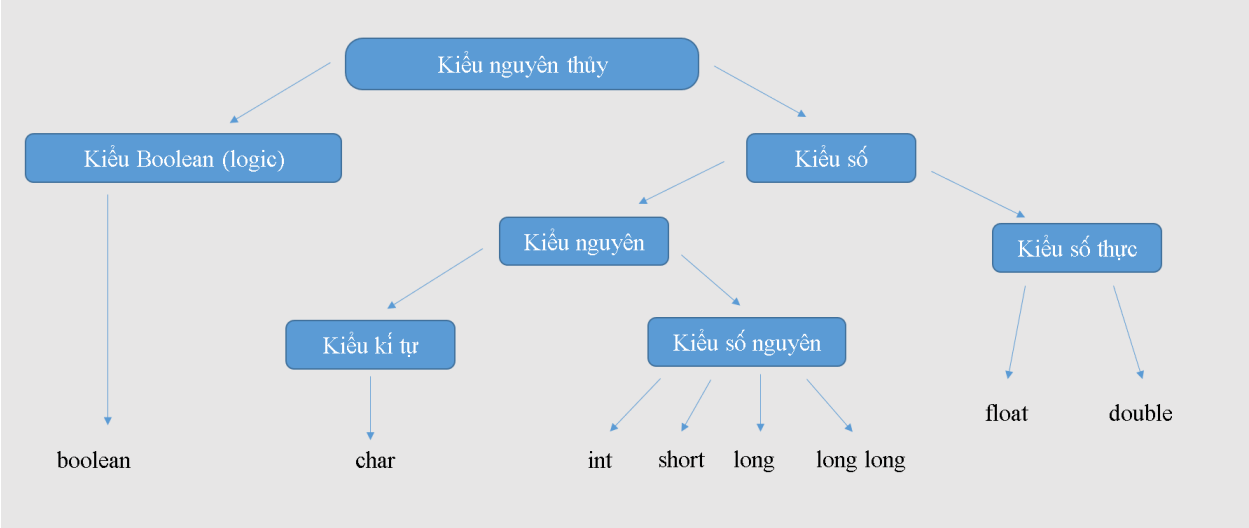
II. Chương trình C đầu tiên.

```c
#include <stdio.h>

int main(){
    printf("Hello world!");
    return 0;
}
```

- **#include <stdio.h>** là câu lệnh mặc định **bắt đầu** chương trình C, có tác dụng tải **thư viện chuẩn** vào chương trình.
- **main()** có tác dụng bắt đầu chương trình, chúng ta viết **tất cả** các xử lý chương trình bên trong hàm **main().**
- **int** chỉ định hàm main chỉ trả về giá trị kiểu **int**.
- **{} Bao bọc tất cả** các câu lệnh trong hàm main.
- Lệnh **printf** in ra dữ liệu .
- Lệnh **return** để **trả về** kết quả từ hàm.

*** Lưu ý:** kết thúc mỗi câu lệnh **đều** có dấu **;**

III. Kiểu dữ liệu trong C.



1. **Các kiểu dữ liệu**
   a. Kiểu số nguyên.

Kiểu	Kích thước	Vùng giá trị
char	1 byte	-128 tới 127 hoặc 0 tới 255
unsigned char	1 byte	0 tới 255
signed char	1 byte	-128 tới 127
int	2 hoặc 4 bytes	-32,768 tới 32,767 hoặc -2,147,483,648 tới 2,147,483,647
unsigned int	2 hoặc 4 bytes	0 tới 65,535 hoặc 0 tới 4,294,967,295
short	2 bytes	-32,768 tới 32,767
unsigned short	2 bytes	0 tới 65,535
long	4 bytes	-2,147,483,648 tới 2,147,483,647
unsigned long	4 bytes	0 tới 4,294,967,295
long long	8 bytes	-9223372036854775808 tới 9223372036854775807
unsigned long long	8 bytes	0 tới 18446744073709551615

b. **Kiểu số thực**

Type	Kích thước	Phạm vi giá trị	Độ chính xác
float	4 byte	1.2E-38 to 3.4E+38	6 chữ số thập phân
double	8 byte	2.3E-308 to 1.7E+308	15 chữ số thập phân
long double	10 byte	3.4E-4932 to 1.1E+4932	19 chữ số thập phân

c. **Kiểu kí tự**

Type	Kích thước	Phạm vi giá trị
char or signed char	1 byte	-128 to 127
unsigned char	1 byte	0 to 255

d. **Kiểu void:** Không trả về giá trị nào.

2. **Các kiểu dữ liệu hay dùng và đặc tả của nó.**
   a. **%d:** số nguyên hệ 10 có dấu
   b. **%u:** số nguyên hệ 10 không dấu
   c. **%x:** số nguyên hệ 16
   d. **%o:** số nguyên hệ bát phân
   e. **%s:** xâu kí tự
   f. **%c:** một kí tự đơn
   g. **%f:** số chấm động cố định
   h. **%e:** số chấm động (ký hiệu có số mũ)
   i. **l :** Tiền tố dùng kèm với %d, %x, %o để chỉ số nguyên dài (ví dụ %ld)

3. **Cách lấy kiểu dữ liệu và kích thước**

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

int main() {
    printf("Size cua kieu long long %d",sizeof(long long));
    printf("\nSize cua kieu bool %d",sizeof(bool));
    printf("\nGia tri min - max cua kieu int: %d %d", INT_MIN, INT_MAX);
    printf("\nGia tri min - max cua kieu char: %d %d", CHAR_MIN, CHAR_MAX);
    return 0;
}
```

IV. Biến trong C

1. **Khái niệm.**
   a. **Một biến** trong C là **tên** của **vị trí bộ nhớ**. Nó được sử dụng để **lưu trữ dữ liệu**. Giá trị của nó có thể được **thay đổi** và nó có thể được sử dụng lại **nhiều lần**. Mỗi biến trong C có một loại dữ liệu **cụ thể**, xác định **kích thước** của bộ nhớ của biến; phạm vi các giá trị có thể được **lưu trữ trong bộ nhớ đó**.
   b. **Biến** là một cách để thể hiện vị trí bộ nhớ thông **qua một cái tên** để nó có thể được xác định một cách **dễ dàng**. Biến trong C có **phân biệt chữ hoa và chữ thường**.

2. **Cú pháp khai báo biến.**
   a. Cú pháp

```
type variable_list;
```

   b. Ví dụ:

```
int a, b, c;
float _hb;
char xin6;
```

3. **Quy tắc khai báo biến trong C.**
   a. Một biến **có thể có** các **chữ cái, chữ số** và **dấu gạch dưới.**
   b. Tên biến **chỉ** có **thể bắt** đầu bằng **chữ cái hoặc dấu gạch dưới.**
   c. **Không có khoảng trắng** trong tên biến.
   d. Tên biến không phải là **từ khóa dành riêng** như int, float,…..
   e. Tên biến có **phân biệt hoa thường**.

4. **Các kiểu biến trong C.**
   a. **Biến local (địa phương).**
      - Khai báo trong hàm.

```
3  int main(){
4      int A;
5      char c;
6      return 0;
7  }
```

b. **Biến global (toàn cầu).**
   - Khai báo ngoài hàm

```
3  int heluu;
4  float k22;
5
6  int main(){
7
8      return 0;
9  }
```

c. Biến static (tĩnh).
   - Được khai báo với từ khóa **static.**

   ➔ Lưu lại giá trị khi gọi lại hàm.

d. Biến automatic.
   - Được khai báo với từ khóa **auto.**

   ➔ Cấp phát 1 giá trị bất kì khi tạo biến.

e. Biến external.
   - Được khai báo với từ khóa **externa.**

   ➔ Khai báo biến bên ngoài.

V. Một số kí tự điều khiển

a. **\n :** Xuống dòng
b. **\t :** Tab ngang (tạo khoảng trắng giống như khi bạn ấn phím Tab trên bàn phím trong soạn thảo văn bản)
c. **\r :** Nhảy về đầu hàng
d. **\a :** Kêu Bip
e. **\\ :** In ra dấu \
f. **\" :** In ra dấu "
g. **\' :** In ra dấu '
h. **%% :** In ra dấu %

VI. Một số toán tử cơ bản trong C

1. Một số phép toán cơ bản.

Toán tử	Miêu tả
+	Cộng
-	Trừ
*	Nhân
/	Chia
%	Lấy dư
<	Nhỏ hơn
<=	Nhỏ hơn hoặc bằng
>	Lớn hơn
>=	Lớn hơn hoặc bằng

2. Toán tử tăng giảm

Toán tử	Miêu tả	Ví dụ	
		Tiền tố	Hậu tố
++	Tăng lên 1 đơn vị	++a;	a++;
		Tương đương a = a+1;	
--	Trừ đi 1 đơn vị	--a;	a--;
		Tương đương a = a-1;	

3. Toán tử gán

Toán tử	Miêu tả	Ví dụ
=	Toán tử gán đơn giản. Gán giá trị toán hạng bên phải cho toán hạng trái.	a = 10; a = b + c;
+=	Thêm giá trị toán hạng phải tới toán hạng trái và gán giá trị đó cho toán hạng trái.	a += c; tương đương a = a+c;
-=	Trừ đi giá trị toán hạng phải từ toán hạng trái và gán giá trị này cho toán hạng trái.	a -= c; tương đương a = a-c;
*=	Nhân giá trị toán hạng phải với toán hạng trái và gán giá trị này cho toán hạng trái.	a *= c; tương đương a = a*c;

/=	Chia toán hạng trái cho toán hạng phải và gán giá trị này cho toán hạng trái.	a /= c; tương đương a = a/c;
%=	Lấy phần dư của phép chia toán hạng trái cho toán hạng phải và gán cho toán hạng trái.	a %= c; tương đương a = a%c;

4. **Các phép toán logic**

Toán tử	Miêu tả
&&	Và
||	Hoặc
!=	Khác

5. **Toán tử điều kiện.**

bieu_thuc_1 ? bieu_thuc_2 : bieu_thuc_3;

6. **Các ngôi của toán tử**

Một ngôi	Hai ngôi	Ba ngôi
++, --	+, -, *, /, % , !=, &&,…..	? :

VII. Một số chương trình cơ bản.

Code	Input	Output

#include <stdio.h>
 	
int main(){
  int a = 4;
  float b =16.5;
  printf("Gia tri cua a: %d",a);
		Gia tri cua a: 4
		Gia tri cua f: 16.500000
		a+5=9 -- b*2=33.00

printf("\nGia tri cua f: %f",b);
printf("\na+5=%d -- b*2=%.2f",a+5,b*2);
return 0;
}		

#include <stdio.h>
 	
int main(){
int a = 15, b=10;
printf("a/b = %d",a/b);
printf("\na/b = %f",a/b);
printf("\na/b = %.2f",(float)a/b);
return 0;
}
		a/b = 1
		a/b = 0.000000
		a/b = 1.50

#include <stdio.h>
 	
int main(){
int a;
scanf("%d",&a);
printf("a = %d",a);
return 0;
}
	30	a =30

#include <stdio.h>
 	
int main(){
int a;
float b;
char c;
scanf("%c%d%f",&c,&a,&b);
printf("%c %d %.2f",c,a,b);
return 0;
}
	c
	16
	2.99
		C 16 2.99

VIII. Ký tự và bảng mã ASCII

1. **Bảng mã ASCII là gì?**
   a. ASCII là từ viết tắt của American Standard Code for Information Interchange có nghĩa là chuẩn mã trao đổi thông tin Hoa Kỳ. Đây là bộ mã hóa ký tự cho bảng chữ cái La Tinh và được dùng để hiển thị văn bản trong máy tính.
   b. Về bản chất, bạn có thể hiểu ASCII là một bộ mã quy ước giúp máy tính có hiểu và hiển thị được các ký tự mà bạn muốn nhập vào máy tính hay

đơn giản hơn là các ký tự trên bàn phím máy tính chuẩn Anh. Tập hợp các mã ASCII tạo thành bảng mã ASCII.

ASCII Table

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	`
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(	72	48	110	H	104	68	150	h
9	9	11		41	29	51	)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[	123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	|
29	1D	35		61	3D	75	=	93	5D	135	]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~
31	1F	37		63	3F	77	?	95	5F	137	_	127	7F	177	

2. **Ví dụ:**

```
#include <stdio.h>

int main(){
    char c;
    scanf("%c",&c);
    printf("%c %d",c,c);
    return 0;
}
```

Input	Output
a	97
G	71
%	37

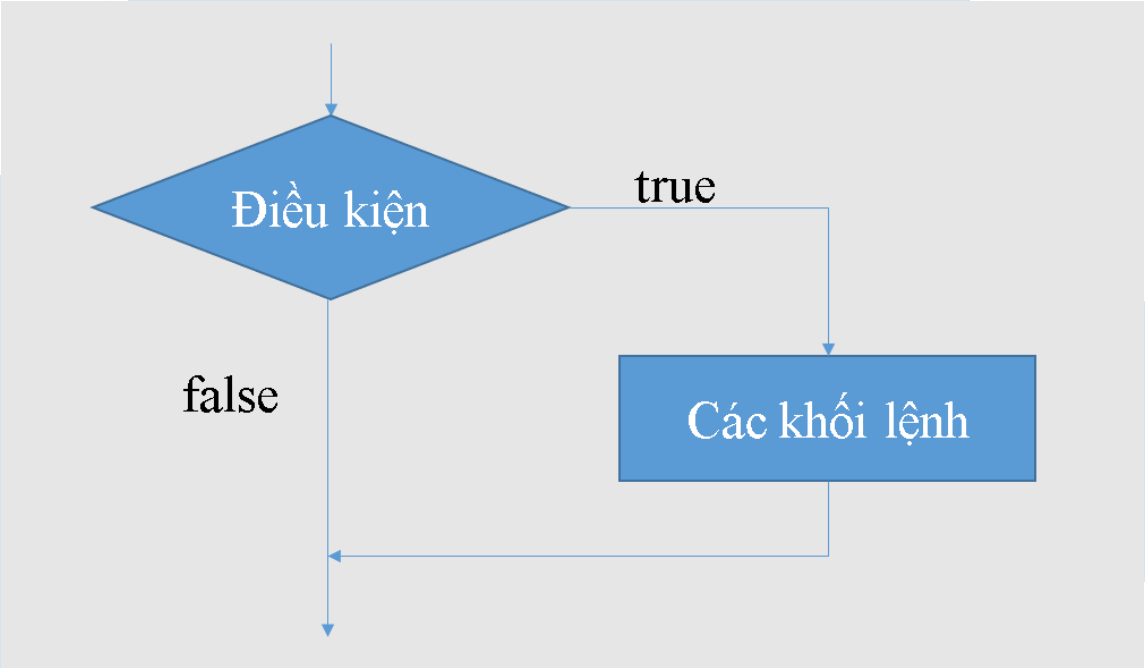
IX. Câu lệnh điều khiển trong C

1. Câu lệnh if

a. Cấu trúc

```
if (điều kiện){
    // Khối lệnh sẽ được thực hiện nếu <điều kiện> đúng.
}
```

b. Lưu đồ:



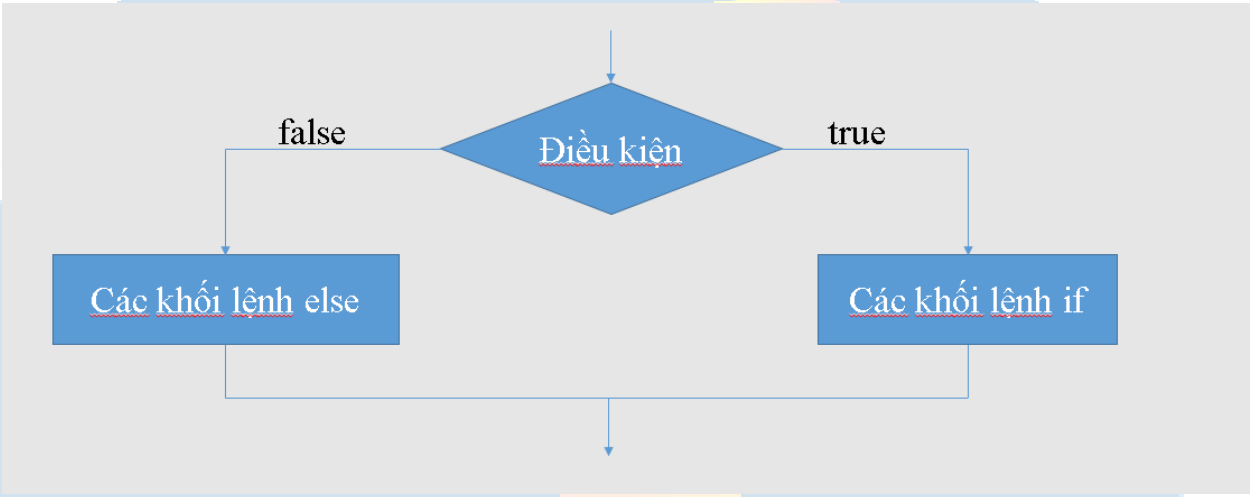
c. Ví dụ: Kiểm tra số nhập vào có phải số lẻ hay không?

```c
#include <stdio.h>

int main() {
    int a;
    printf("Nhap a = ");
    scanf("%d", &a);

    if (a % 2 != 0) {
        printf("%d la so le", a);
    }
    printf("\nHoan thanh!");
}
```

2. Câu lệnh if else
   a. Cấu trúc

```
if (điều kiện){
      // Khối lệnh sẽ được thực hiện nếu <điều kiện> đúng.
}else {
      // Khối lệnh sẽ được thực hiện nếu <điều kiện> sai.
}
```

   b. Lưu đồ:



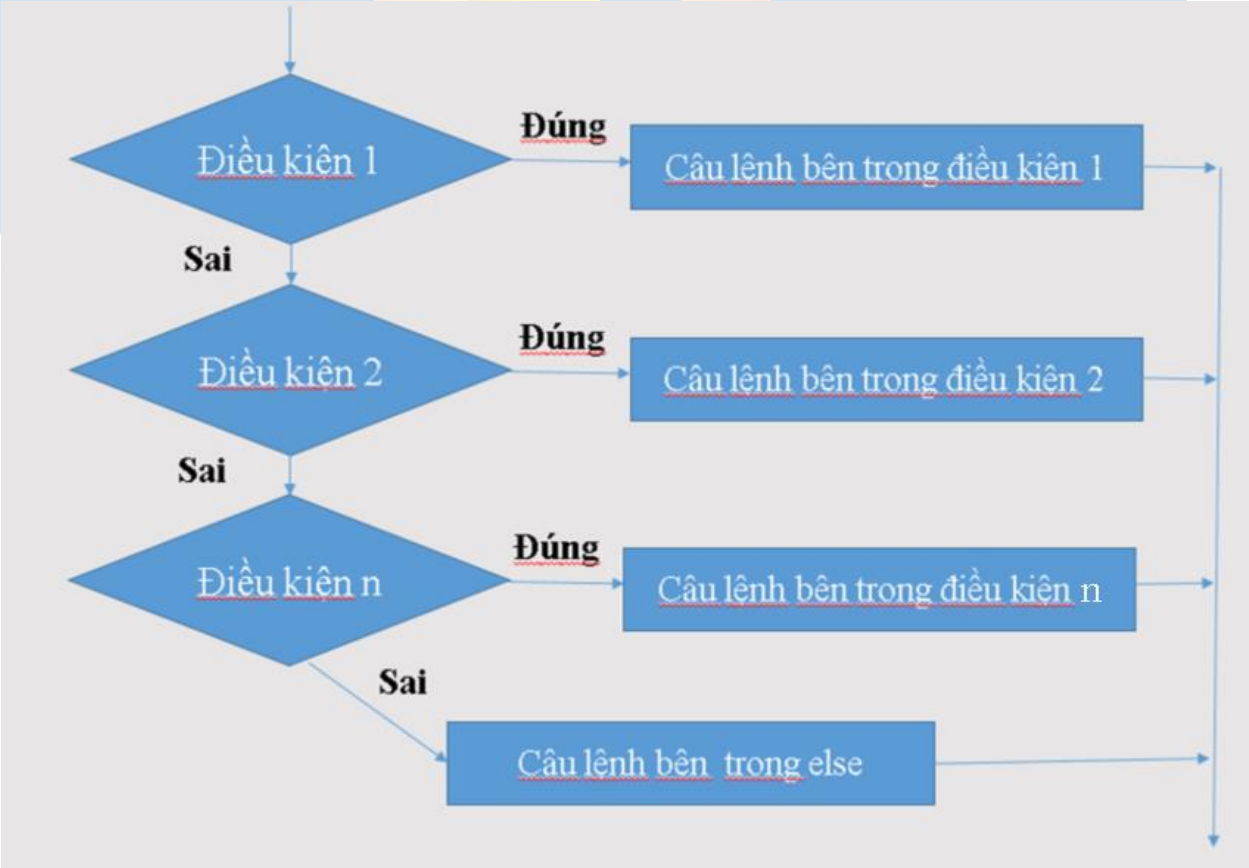
   c. Ví dụ: Kiểm tra số có chia hết cho 5 hay không?

```
3  #include <stdio.h>
4
5  int main(){
6      int n;
7      scanf("%d",&n);
8      if(n%5==0){
9          printf("%d chia het cho 5",n);
10     }else {
11         printf("%d khong chia het cho 5",n);
12     }
13     return 0;
14 }
```

3. **Cấu trúc if ….. else if …… else.**
   a. Cấu trúc

```
if (điều kiện 1){
      // Khối lệnh sẽ được thực hiện nếu <điều kiện 1> đúng.
}else if (điều kiện 2){
      // Khối lệnh sẽ được thực hiện nếu <điều kiện 2> đúng.
}else if (điều kiện 3){
      // Khối lệnh sẽ được thực hiện nếu <điều kiện 3> đúng.
}
.
.
.
else{
     // Khối lệnh sẽ được thực hiện nếu tất cả điều kiện sai.
}
```

   b. Lưu đồ:



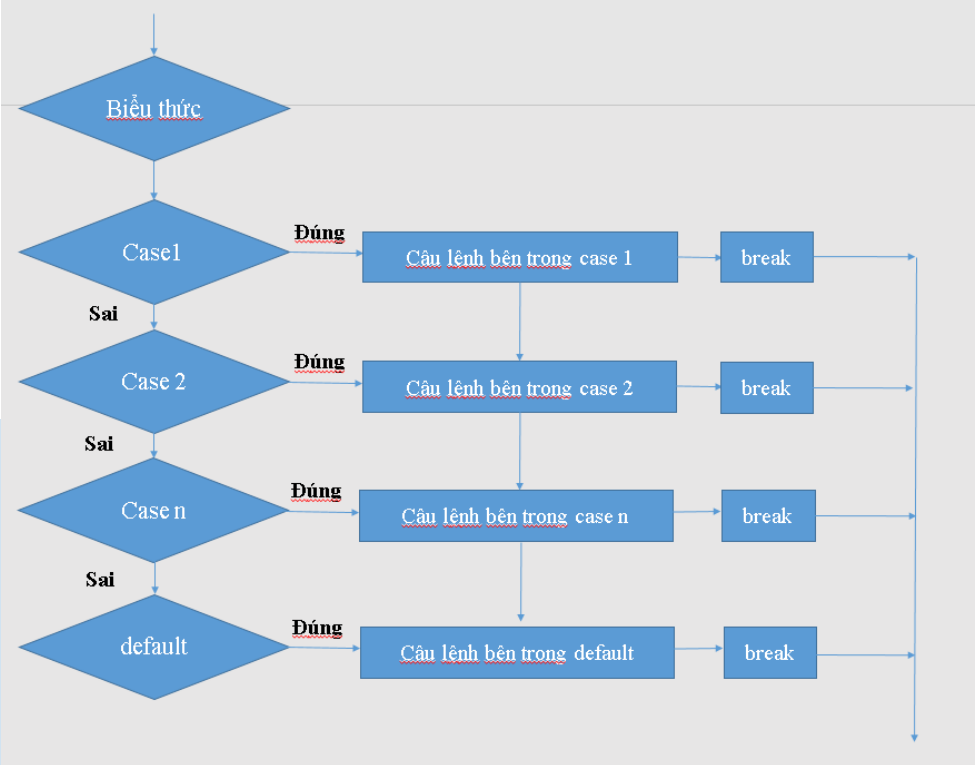
c. Ví dụ: So sánh 2 số nhập vào?

```c
#include <stdio.h>

int main() {
    int a,b;
    scanf("%d%d", &a, &b);

    if (a > b) {
        printf("%d > %d", a,b);
    }else if  (a < b){
        printf("%d < %d", a,b);
    }else {
        printf("%d = %d", a,b);
    }
}
```

4. **Câu lệnh switch case**
   a. Cấu trúc

```
switch (bieu_thuc) {
    case gia_tri_1:
      // Khối lệnh 1
      break;  //tùy chọn
    case gia_tri_2:
      // Khối lệnh 2
      break;  //tùy chọn
    ......
    case gia_tri_n:
      // Khối lệnh n
      break;  //tùy chọn
    default:
      // Khối lệnh
    }
```

b. Lưu đồ:

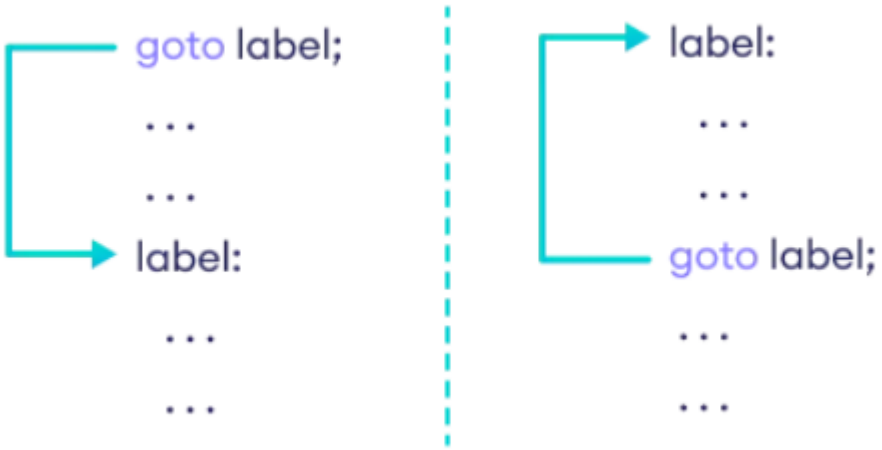


c. Ví dụ:

```c
#include <stdio.h>

int main() {
    int a;
    scanf("%d", &a);

    switch (a){
        case 10:
            printf("10");
        case 11:
            printf("\n11");
        case 12:
            printf("\n12");
            break;
        case 13:
            printf("\n13");
        default:
            printf("\nKhac!");
    }
}

```

5. **Câu lệnh goto**
   a. Cấu trúc

goto label; **…..** **…..** **label: Các câu lệnh;**	**label: Các câu lệnh;** **…..** **…..** **goto label;**



   b. Ví dụ:

```c
#include <stdio.h>

int main(){
    int n;
    nhap:
        printf("Xin chao moi nguoi\n");
    scanf("%d",&n);
    if(n>=10&&n<=100) {
        goto nhap;
    }
}
```