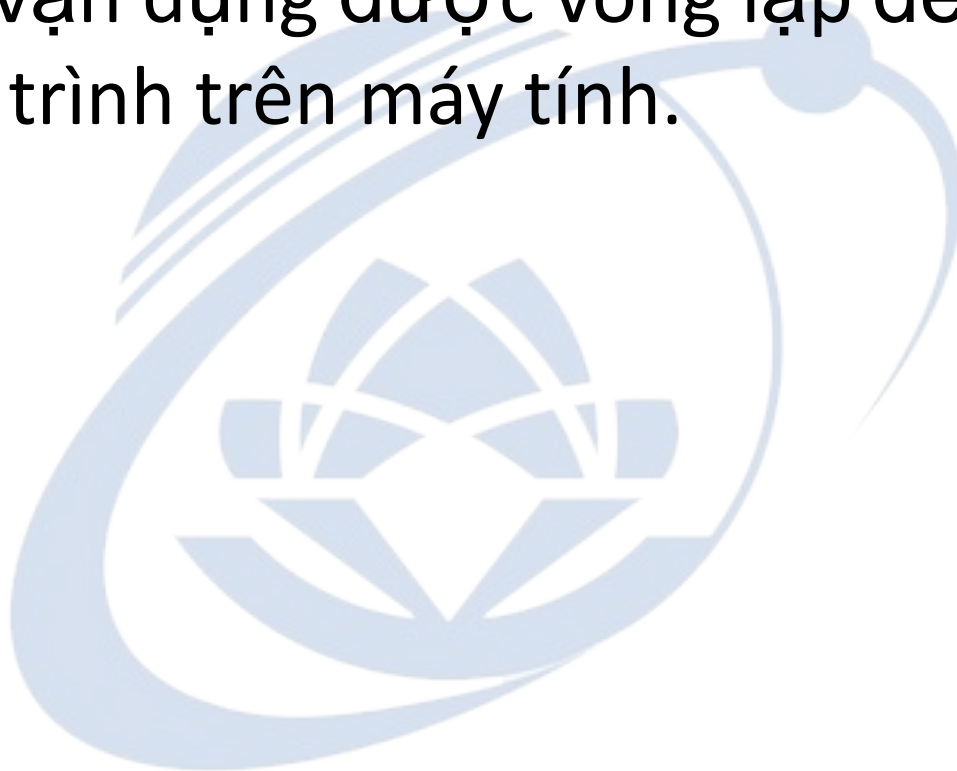


Nhập môn Lập trình – IT001

Buổi 05 – Các câu lệnh lặp

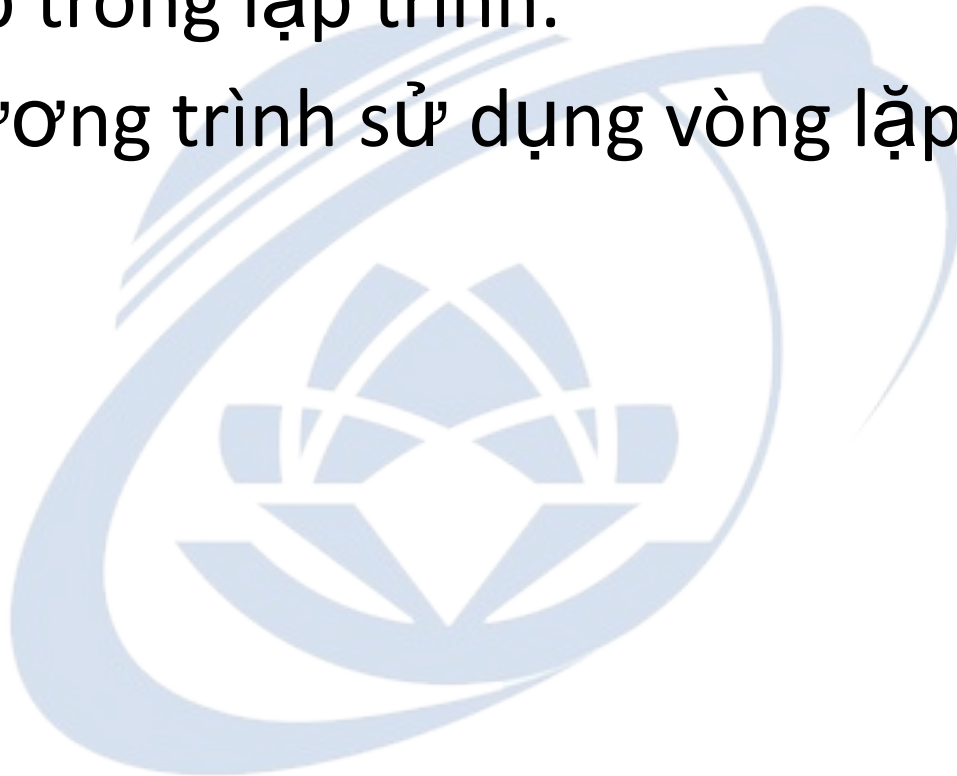
Mục tiêu buổi học – CĐR

- Hiểu và vận dụng được vòng lặp để viết chương trình trên máy tính.



Nội dung

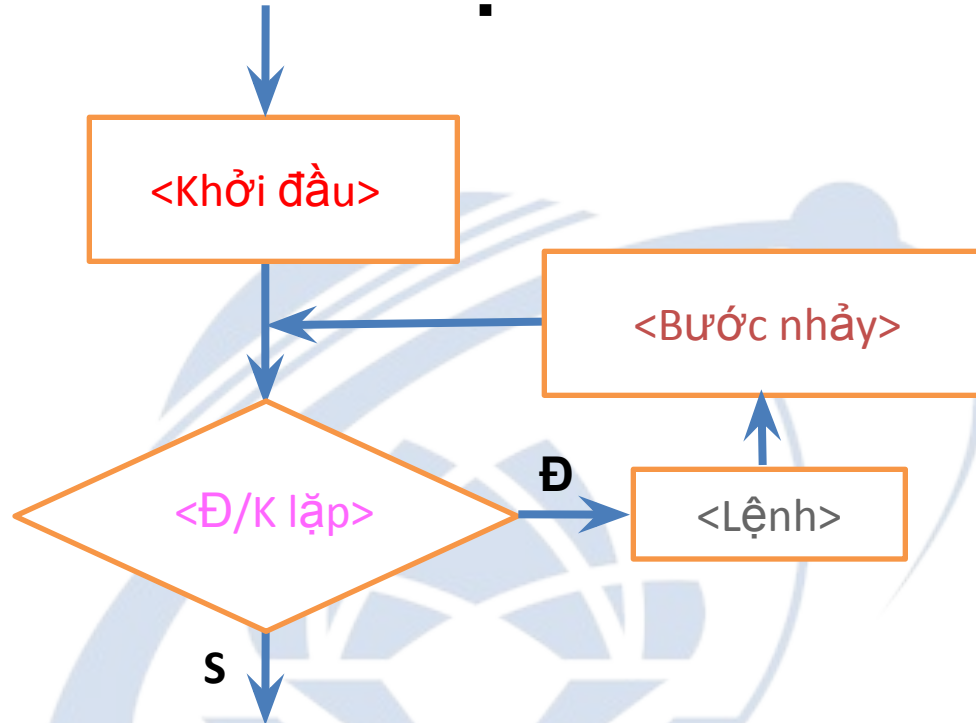
- Vòng lặp trong lập trình.
- Viết chương trình sử dụng vòng lặp



Đặt vấn đề

- Ví dụ
 - Viết chương trình xuất các số từ 1 đến 10
=> Sử dụng 10 câu lệnh printf
 - Viết chương trình xuất các số từ 1 đến 1000
=> Sử dụng 1000 câu lệnh printf !
- Giải pháp
 - Sử dụng cấu trúc lặp lại một hành động trong khi còn thỏa một điều kiện nào đó.
 - 3 lệnh lặp: for, while, do... while

Câu lệnh for



for (<Khởi đầu>; <Đ/K lặp>; <Bước nhảy>)
 <Lệnh>;

<Khởi đầu>, <Đ/K lặp>, <Bước nhảy>: là biểu thức C bất kỳ
 <Lệnh>: đơn hoặc khối lệnh.

Câu lệnh for

```
void main()
{
    int i;
    for (i = 0; i < 10; i++)
        printf("%d ", i);

    for (int j = 0; j < 10; j = j + 1)
        printf("%d ", j);

    for (int k = 0; k < 10; k += 2)
    {
        printf("%d ", k);
        printf("\n");
    }
}
```

Câu lệnh for - Một số lưu ý

- Câu lệnh for **có thể lồng nhau**.

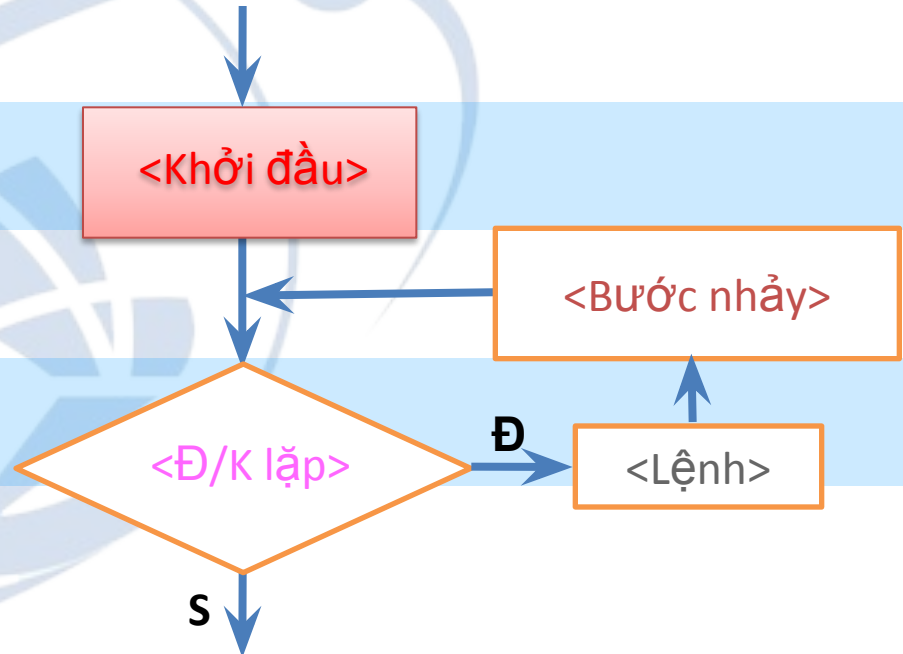
```
if (n < 10 && m < 20)
{
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < m; j++)
        {
            printf("%d", i + j);
            printf("\n");
        }
    }
}
```

Câu lệnh for - Một số lưu ý

- Trong câu lệnh for, có thể sẽ không có phần **<Khởi đầu>**

```
int i;
for (i = 0; i < 10; i++)
    printf("%d\n", i);
```

```
int i = 0;
for (; i < 10; i++)
    printf("%d\n", i);
```

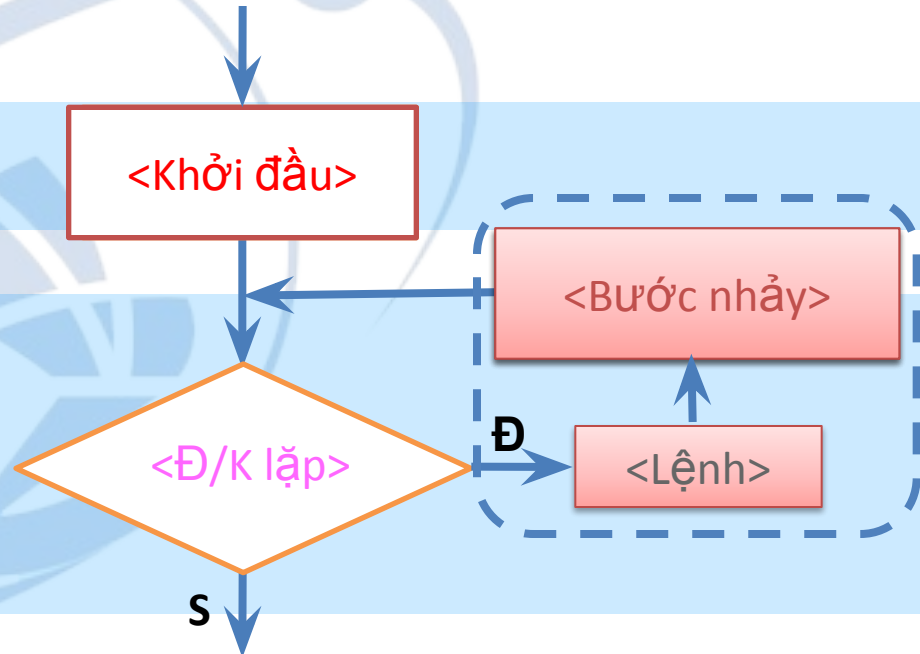


Câu lệnh for - Một số lưu ý

- Trong câu lệnh for, có thể sẽ không có phần **<Bước nhảy>**

```
int i;
for (i = 0; i < 10; i++)
    printf("%d\n", i);
```

```
for (i = 0; i < 10; )
{
    printf("%d\n", i);
    i++;
}
```



Câu lệnh for - Một số lưu ý

- Trong câu lệnh for, có thể sẽ không có phần **<Đ/K lặp>**

```
int i;  
for (i = 0; i < 10; i++)  
    printf("%d\n", i);
```

```
for (i = 0; ; i++)  
    printf("%d\n", i);
```

```
for (i = 0; ; i++)  
{  
    if (i >= 10)  
        break;  
    printf("%d\n", i);  
}
```

Câu lệnh for - Một số lưu ý

- Lệnh **break** làm kết thúc câu lệnh.
- Lệnh **continue** bỏ qua lần lặp hiện tại.

```
for (i = 0; i < 10; i++)  
{  
    if (i % 2 == 0)  
        break;  
    printf("%d\n", i);  
}
```

```
for (i = 0; i < 10; i++)  
{  
    if (i % 2 == 0)  
        continue;  
    printf("%d\n", i);  
}
```

Câu lệnh for - Một số lưu ý

- Không được thêm **;** ngay sau lệnh **for**.

=> Trường hợp câu lệnh rỗng.

```
for (i = 0; i < 10; i++);
{
    printf("%d", i);
    printf("\n");
}
```

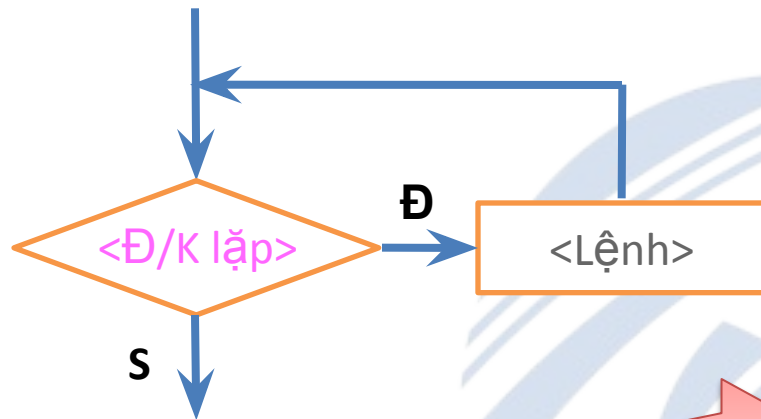
```
for (i = 0; i < 10; i++)
{
};
{
    printf("%d", i);
    printf("\n");
}
```

Câu lệnh for - Một số lưu ý

- Các thành phần <Khởi đầu>, <Đ/K lặp>, <Bước nhảy> cách nhau bằng dấu ;
- Nếu có nhiều thành phần trong mỗi phần thì được cách nhau bằng dấu ,

```
for (int i = 1, j = 2; i + j < 10; i++, j += 2)  
    printf("%d\n", i + j);
```

Câu lệnh while



Biểu thức C bất kỳ,
thường là biểu thức
quan hệ cho kết quả
0 (sai) và **!= 0** (đúng)

Câu lệnh đơn hoặc
Câu lệnh phức (kẹp
giữa **{** và **}**)

while (**<Đ/K lặp>**)
<Lệnh>;

Câu lệnh while

```
int i = 0;
while (i < 10)
{
    printf("%d\n", i);
    i++;
}
```

```
for (int i = 0; i < 10; i++)
    printf("%d\n", i);
```

```
int i = 0;
for (; i < 10; )
{
    printf("%d\n", i);
    i++;
}
```

Câu lệnh while - Một số lưu ý

- Câu lệnh **while** là một **câu lệnh đơn** và **có thể lồng nhau**.

```
if (n < 10 && m < 20)
{
    while (n >= 1)
    {
        while (m >= 1)
        {
            printf("%d", m);
            m--;
        }
        n--;
    }
}
```


Câu lệnh while - Một số lưu ý

- Câu lệnh **while** có thể không thực hiện lần nào do điều kiện lặp ngay từ lần đầu đã không thỏa.

```
void main()
{
    int n = 1;
    while (n > 10)
    {
        printf("%d\n", n);
        n--;
    }
    ...
}
```

Câu lệnh for - Một số lưu ý

- Không được thêm `;` ngay sau lệnh `while`.

```
int n = 0;  
while (n < 10) ;  
{  
    printf("%d\n", n);  
    n++;  
}
```

```
while (n < 10)  
{  
};  
{  
    printf("%d\n", n);  
    n++;  
}
```

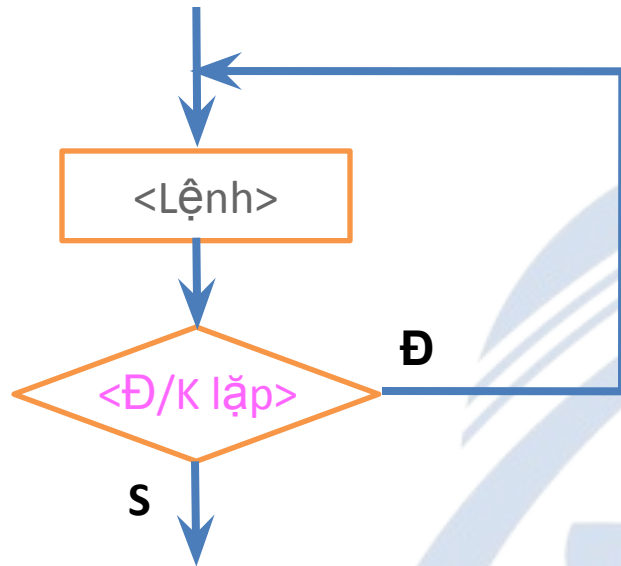
Câu lệnh while - Một số lưu ý

- Câu lệnh **while** có thể bị lặp vô tận (**loop**)

```
void main()
{
    int n = 1;
    while (n < 10)
    {
        printf("%d", n);
        n--;
    }

    n = 1;
    while (n < 10)
        printf("%d", n);
}
```

Câu lệnh do... while



do

<Lệnh>;

while (<Đ/K lặp>;

Câu lệnh đơn hoặc
Câu lệnh phức (kẹp
giữa { và })

Biểu thức C bất kỳ,
thường là biểu thức
quan hệ cho kết quả
0 (sai) và **!= 0** (đúng)

Câu lệnh do... while

```
int i = 0;  
do  
{  
    printf("%d\n", i);  
    i++;  
}  
while (i < 10);
```

```
int i = 0;  
printf("%d\n", i);  
i++;  
for (; i < 10; )  
{  
    printf("%d\n", i);  
    i++;  
}
```

Câu lệnh do... while - Một số lưu ý

- Câu lệnh **do... while** là một **câu lệnh đơn** và **có thể lồng nhau**.

```
int a = 1, b;
do
{
    b = 1;
    do
    {
        printf("%d\n", a + b);
        b = b + 2;
    }
    while (b < 20);
    a++;
}
while (a < 20);
```

Câu lệnh do... while - Một số lưu ý

- Câu lệnh **do... while** sẽ được thực hiện ít nhất 1 lần do điều kiện lặp được kiểm tra ở cuối.

```
void main()
{
    int n;
    do
    {
        printf("Nhap n: ");
        scanf("%d", &n);
    }
    while (n < 1 || n > 100);
}
```

Câu lệnh do... while - Một số lưu ý

- Câu lệnh **do... while** có thể bị lặp vô tận (**loop**)

...

```
int n = 1;
do
{
    printf("%d", n);
    n--;
}
while (n < 10);
```

```
n = 1;
do
    printf("%d", n);
while (n < 10);
```


for, while, do... while

- Điều có khả năng lặp lại nhiều hành động.

```
int n = 10;  
for (int i = 1; i <= n; i++)  
    printf("%d\n", i);
```

```
int i = 1;  
while (i <= n)  
{  
    printf("%d\n", i); i++;  
}
```

```
int i = 1;  
do {  
    printf("%d\n", i); i++;  
} while (i < n);
```

for, while, do... while

- Số lần lặp xác định ngay trong câu lệnh **for**

```
int n = 10;  
for (int i = 1; i <= n; i++)  
    ...;
```

```
int i = 1;  
while (i <= n)  
{  
    ...;  
}
```

```
int i = 1;  
do {  
    ...;  
} while (i > n);
```

while & do... while

- while có thể không thực hiện lần nào.
- do... while sẽ được thực hiện ít nhất 1 lần.

```
int n = 100;  
while (n < 10)  
{  
    ...;  
}  
...  
do  
{  
    printf("Nhap n: ");  
    scanf("%d", &n);  
}  
while (n > 10);
```

Một số lệnh liên quan đến lệnh lặp

Nhóm các lệnh ***goto, break, continue, return***

Lệnh ***goto***

Cú pháp khai báo

***goto** nhãn;*

- *Khi gặp lệnh **goto** máy sẽ nhảy tới thực hiện câu lệnh viết sau **nhãn**.*

Một số lệnh liên quan đến lệnh `lặp`

Ví dụ minh họa:

```
void main(){  
    ...           // Khai báo biến  
    tt:  
    clrscr();  
    ...           // Các câu lệnh  
    printf("\nNhan ESC de ket thuc chuong trinh...");  
    if (getch() != 27) // getch(): chờ nhấn một phím bất kỳ  
        goto tt;  
}
```

- Nếu **mã của phím nhấn vào** `!= ESC` thì thực hiện lại chương trình kể từ câu lệnh `clrscr()`;

Một số lệnh liên quan đến lệnh lặp

Lệnh *break*

- Lệnh *break* chỉ được khai báo bên trong các câu lệnh vòng lặp *for*, *while*, *do...while* hoặc *switch*.
- Khi gặp câu lệnh *break* máy sẽ thoát khỏi vòng lặp *trong cùng chứa nó*.

Một số lệnh liên quan đến lệnh lặp

Ví dụ minh họa:

Viết đoạn chương trình kiểm tra n có phải là số nguyên tố không?

```
int i, t;  
t = sqrt(n);  
i = 2;  
while ( i<=t )  
    if (n%i == 0) break; // kết thúc vòng lặp  
    else i++;  
if ( i>t ) printf(“%d là SNT”, n);  
else      printf(“%d không là SNT”, n);
```

Một số lệnh liên quan đến lệnh lặp

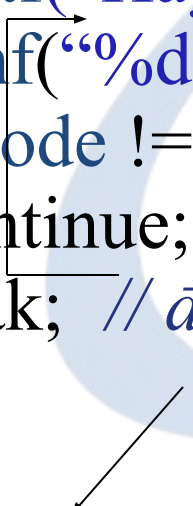
3. Lệnh *continue*

- Lệnh *continue* chỉ được khai báo bên trong các vòng lặp *for*, *while* hoặc *do...while*
- Khi gặp câu lệnh *continue* máy sẽ bỏ qua các câu lệnh còn lại trong thân vòng lặp để bắt đầu một lần lặp mới.

Một số lệnh liên quan đến lệnh lặp

Ví dụ minh họa:

```
...  
int code;  
while(1)    // vòng lặp vô hạn  
{ printf("Hay cho biet mat ma: ");  
  scanf("%d", &code);  
  if (code != 999)  
    continue;  
  break;    // đã nhập đúng, kết thúc vòng lặp  
}  
...
```



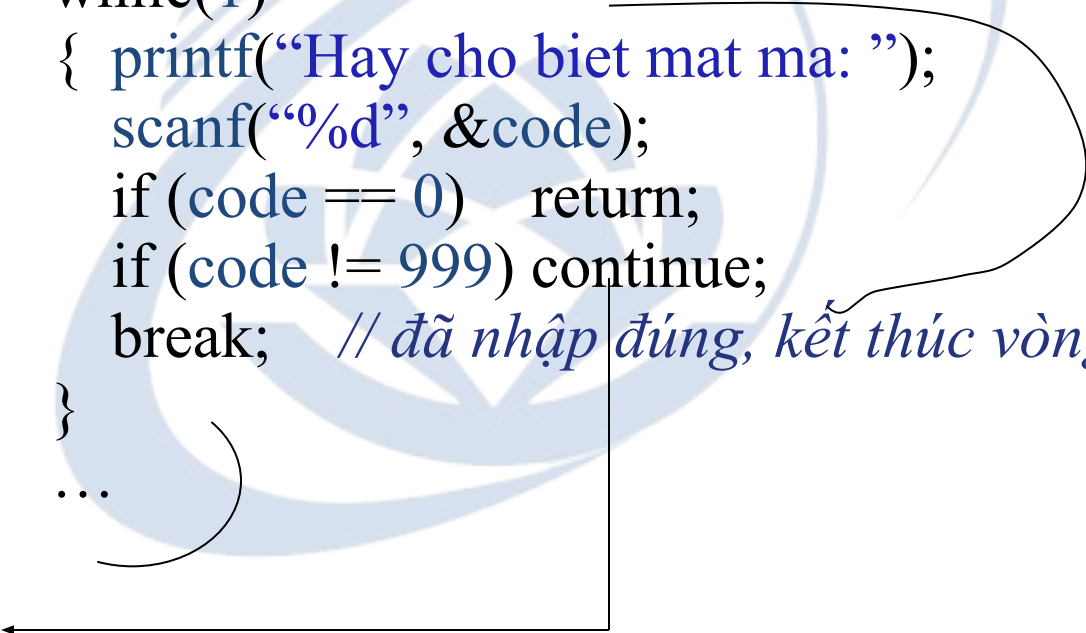
Một số lệnh liên quan đến lệnh lặp

4. Lệnh *return*

- Khi gặp lệnh *return* máy sẽ kết thúc hàm chứa nó.
- Ví dụ minh họa:

Một số lệnh liên quan đến lệnh lặp

```
void main()
{
    ...
    int code;
    while(1)
    {
        printf("Hay cho biet mat ma: ");
        scanf("%d", &code);
        if (code == 0) return;
        if (code != 999) continue;
        break; // đã nhập đúng, kết thúc vòng lặp
    }
    ...
}
```



Bài tập thực hành

3. Nhập một số nguyên dương n ($n > 0$).

Hãy cho biết:

- a. Có phải là số đối xứng? Ví dụ: 121, 12321, ...
- b. Có phải là số chính phương? Ví dụ: 4, 9, 16, ...
- c. Có phải là số nguyên tố? Ví dụ: 2, 3, 5, 7, ...
- d. Chữ số lớn nhất và nhỏ nhất?
- e. Các chữ số có tăng dần hay giảm dần không?



Bài tập thực hành

4. Nhập một số nguyên dương n . Tính:



a. $S = 1 + 2 + \dots + n$



b. $S = 1^2 + 2^2 + \dots + n^2$



c. $S = 1 + 1/2 + \dots + 1/n$



d. $S = 1 * 2 * \dots * n = n!$



e. $S = 1! + 2! + \dots + n!$







5. Nhập 3 số nguyên a , b và n với $a, b < n$. Tính tổng các số nguyên dương nhỏ hơn n chia hết cho a nhưng không chia hết cho b .



6. Tính tổng các số nguyên tố nhỏ hơn n ($0 < n \leq 50$)



Bài tập thực hành

-  7. Nhập một số nguyên dương n . Xuất ra số ngược lại. Ví dụ: Nhập 1706 □ Xuất 6071.
-  8. Tìm và in lên màn hình tất cả các số nguyên trong phạm vi từ 10 đến 99 sao cho tích của 2 chữ số bằng 2 lần tổng của 2 chữ số đó.
-  9. Tìm ước số chung lớn nhất của 2 số nguyên dương a và b nhập từ bàn phím.
10. Nhập n . In n số đầu tiên trong dãy Fibonacci.
 -  a. $a_0 = a_1 = 1$
 - b. $a_n = a_{n-1} + a_{n-2}$



Bài tập 3a

```
void main()
{
    int n, sogoc, sodao, donvi;
    printf("Nhap n: ");
    scanf("%d", &n);

    sogoc = n; sodao = 0;
    while (sogoc > 0)
    {
        donvi = sogoc % 10;
        sodao = sodao*10 + donvi;
        sogoc = sogoc / 10;
    }
    if (sodao == n) printf("DX");
    else printf("Khong doi xung");
}
```

Bài tập 3b

```
#include <math.h>

void main()
{
    int n, n_can_nguyen;

    printf("Nhap n: ");
    scanf("%d", &n);

    n_can_nguyen = int(sqrt(n));
    if (n_can_nguyen*n_can_nguyen == n)
        printf("%d la so CP.", n);
    else
        printf("%d khong la so CP.", n);
}
```


Bài tập 3c

```
void main()
{
    int n, i, souoc;

    printf("Nhap n: ");
    scanf("%d", &n);

    souoc = 0;
    for (i = 1; i <= n; i++)
        if (n % i == 0)
            souoc++;

    if (souoc == 2)
        printf("%d la so nguyen to");
    else
        printf("%d ko la so nguyen to", n);
}
```

Bài tập 3d

```
void main()  
{  
    int n, min, max, donvi;  
    ...  
    min = n % 10;  
    max = min;  
    n = n / 10;  
  
    while (n>0)  
    {  
        donvi = n % 10;  
        n = n / 10;  
        if (donvi < min) min = donvi;  
        if (donvi > max) max = donvi;  
    }  
    printf("So NN = %d, So LN = %d", min, max);  
}
```

Bài tập 3e

```
void main()
{
    int n, sotruoc, sosau;
    ... // Nhập n
    sotruoc = n % 10;
    do
    {
        sosau = sotruoc;
        n = n / 10;
        sotruoc = n % 10;
    } while (n != 0 && sotruoc < sosau);

    if (sotruoc < sosau)
        printf("Cac chu so tang dan");
    else
        printf("Cac chu so ko tang dan");
}
```

Bài tập 4a

```
void main()
{
    int n, i, s;

    printf("Nhap n: ");
    scanf("%d", &n);

    s = 0;
    for (i = 1; i <= n; i++)
        s = s + i;

    printf("1 + 2 + ... + %d = %d", n, s);
}
```

Bài tập 4b

```
void main()
{
    int n, i, s;

    printf("Nhap n: ");
    scanf("%d", &n);

    s = 0;
    for (i = 1; i <= n; i++)
        s = s + i*i;

    printf("1^2 + 2^2 + ... + %d^2 = %d", n, s);
}
```

Bài tập 4c

```
void main()
{
    int n, i;
    float s;

    printf("Nhap n: ");
    scanf("%d", &n);

    s = 0;
    for (i = 1; i <= n; i++)
        s = s + 1.0/i;

    printf("1 + 1/2 + ... + 1/%d = %f", n, s);
}
```

Bài tập 4d

```
void main()
{
    int n, i, s;

    printf("Nhap n: ");
    scanf("%d", &n);

    s = 1;
    for (i = 2; i <= n; i++)
        s = s * i;

    printf("%d! = %d", n, s);
}
```

Bài tập 4e

```
void main()
{
    int n, i, j, igt, s;
    printf("Nhap n: ");
    scanf("%d", &n);

    s = 0;
    for (i = 1; i <= n; i++)
    {
        igt = 1;
        for (j = 2; j <= i; j++)
            igt = igt * j;
        s = s + igt;
    }
    printf("1! + 2! + ... + %d! = %d", n, s);
}
```


Bài tập 5

```
void main()
{
    int a, b, n, i, s;
    do
    {
        printf("Nhap a, b, n: ");
        scanf("%d%d%d", &a, &b, &n);
    } while (a >= n || b >= n);

    s = 0;
    for (i = 1; i <= n - 1; i++)
        if (i % a == 0 && i % b != 0)
            s = s + i;

    printf("Tong cac thoa yeu cau la %d", s);
}
```

Bài tập 6

```
void main()
{
    int n, i, j, souoc, s;
    do
    {
        printf("Nhap n: ");
        scanf("%d", &n);
    } while (n <= 0 || n >= 50);
    s = 0;
    for (i = 2; i <= n - 1; i++)
    {
        ... // Đếm số ước của i
        if (souoc == 2) // Là số nguyên tố
            s = s + i;
    }
    printf("Tong cac so nt < %d la %d", n, s);
}
```

Bài tập 7

```
void main()
{
    int n, donvi;

    printf("Nhap n: ");
    scanf("%d", &n);

    printf("So dao cua %d la ", n);
    while (n > 0)
    {
        donvi = n % 10;
        n = n / 10;
        printf("%d", donvi);
    }
}
```

Bài tập 8

```
void main()
{
    int n, i, donvi, chuc;

    printf("Cac so thoa yeu cau la: ");
    for (i = 10; i <= 99; i++)
    {
        donvi = i % 10;
        chuc = i / 10;
        if (chuc*donvi == 2*(chuc + donvi))
            printf("%d", i);
    }
}
```

Bài tập 9

- Ví dụ: $a = 12, b = 8$
- Cách 1:
 - Cho 1 biến i chạy từ 8 trở về 1, nếu cả a và b đều chia hết cho i thì dừng và i chính là uscln.
 - 8, 7, 6, 5, 4 \Rightarrow USCLN của 12 và 8 là 4.
- Cách 2:
 - USCLN của a & b (a khác b), ký hiệu (a, b) là:
 - $(a - b, b)$ nếu $a > b$
 - $(a, b - a)$ nếu $b > a$
 - $(12, 8) = (4, 8) = (4, 4) = 4$

Bài tập 9

```
void main()
{
    int a, b, uscln;

    printf("Nhap a va b: ");
    scanf("%d%d", &a, &b);

    if (a < b) uscln = a;
    else uscln = b;

    while (a % uscln != 0 || b % uscln != 0)
        uscln--;

    printf("USCLN cua %d va %d la %d", a, b, uscln);
}
```

Bài tập 9

```
void main()
{
    int a, b;

    printf("Nhap a va b: ");
    scanf("%d%d", &a, &b);

    while (a <> b)
    {
        if (a > b)
            a = a - b;
        else
            b = b - a;
    }
    printf("USCLN cua a va b la %d', a);
}
```

Bài tập 10

- Dãy Fibonacy: $a_0 a_1 a_2 \dots a_{n-2} a_{n-1} a_n$
 - Với $a_0 = a_1 = 1$, $a_n = a_{n-1} + a_{n-2}$
- Ví dụ: 1 1 2 3 5 8 13 21 ...
- Xuất n phần tử đầu tiên của dãy Fibonacy
 - $n = 1 \Rightarrow 1$, $n = 2 \Rightarrow 1\ 1$
 - $n > 2$
 - Lưu lại 2 phần tử trước nó là a và b
 - Mỗi lần tính xong cập nhật lại a và b.
- Nên **thêm 2 phần tử ảo** đầu tiên là a_{-2} , a_{-1}
 - **1 0** 1 1 2 3 5 8 13 21 ...

Bài tập 10

```
void main()  
{  
    int n, an, an1, an2, i;  
  
    printf("Nhap n: ");  
    scanf("%d", &n);  
  
    an2 = 1; an1 = 0;  
    printf("%d phan tu dau tien cua day: ", n);  
    for (i = 1; i <= n; i++)  
    {  
        an = an2 + an1;  
        printf("%d ", an);  
        an2 = an1;  
        an1 = an;  
    }  
}
```

Bài tập

- $S = 1/2 + 1/4 + \dots + 1/2n$
- $S = 1 + 1/3 + 1/5 + \dots + 1/(2n+1)$
- $S = 1/(1x^2) + 1/(2x^3) + \dots + 1/(nx^{n+1})$
- $S = 1/2 + 2/3 + \dots + n/(n+1)$
- $S = 1 + 1/(1 + 2) + \dots + 1/(1 + 2 + \dots + n)$
- Liệt kê tất cả ước số của số nguyên dương n
- Tính tổng các ước số của số nguyên dương n
- Đếm số lượng ước số của số nguyên dương n
- Tính tổng các ước số chẵn của số nguyên dương n