Chương 3: Các cấu trúc điều khiển

Agenda

- Giới thiệu
- Cấu trúc lựa chọn if
- Cấu trúc lựa chọn switch
- Cấu trúc lặp while, do … while, for

1. Giới thiệu

- Một chương trình bao gồm nhiều câu lệnh. Thông thường, các câu lệnh được thực hiện 1 cách lần lượt theo thứ tự mà chúng được viết ra. Các câu lệnh điều khiển cho phép thay đổi trật tự nói trên.
- Có 3 loại cấu trúc điều khiển cơ bản:
  - Cấu trúc tuần tự (sequence),
  - Cấu trúc lựa chọn (selection).
  - Cấu trúc lặp (repetition or loop).

1. Giới thiệu

- Lệnh (statement): là một biểu thức kết thúc bởi dấu “;”
  - Ví dụ: a++;
- Khối lệnh (block): là một dãy bao gồm một hay nhiều lệnh được bao bọc bởi cặp dấu { }.
  - Về mặt cú pháp, khối lệnh tương đương với 1 câu lệnh đơn
  - Ví dụ:

```
if (a<b)
```

2. Cấu trúc điều khiển If (dạng 1)

Cấu trúc if có **2 dạng** tổng quát :

- Dạng 1:

```
if (expression)
   statement;
```

  - *expression*: biểu thức luận lý (có giá trị là true hay false)
  - *statement*: Câu lệnh

2. Cấu trúc điều khiển If (dạng 1)

- Ví dụ: Viết chương trình nhập vào một số thực a. In ra màn hình kết quả kiểm tra a có phải là 1 số âm hay dương?

```
#include <iostream.h>
#include <conio.h>
int main()
{
   int a;
   cout << "Input a = ";
   cin >> a;
   if(a>=0)
      cout << a << " is a positive.”;
   getch();
```

2. Cấu trúc điều khiển If (Dạng 2)

- Dạng 2:

```
if (expression)
    statement1;
else
    statement2;
```

2. Cấu trúc điều khiển If (Dạng 2)

- ***Ví dụ 1*** : Viết chương trình nhập vào một số thực a. In ra màn hình kết quả kiểm tra a có phải là 1 số âm hay dương?

```
#include <iostream.h>
#include <conio.h>
int main()
{
```

2. Cấu trúc điều khiển If (Dạng 2)

- Lưu ý:
  - Ta có thể sử dụng các câu lệnh if…else lồng nhau. Trong trường hợp if…else lồng nhau thì *else sẽ kết hợp với if gần nhất chưa có else*.
  - Trong trường hợp câu lệnh if "bên trong" không có else thì phải viết nó trong cặp dấu {} (coi nh… ệnh) … ư kết hợp else if s…
    - Ví dụ:

```
if  (n>0)
   if (a>b)
        z=a;
   else
        z=b;
```

```
if  (n>0)
{
   if (a>b)
        z=a;
   else
        z=b;
)
```

2. Cấu trúc điều khiển If (Dạng 2)

- Khi muốn thực hiện 1 trong n quyết định, ta có thể sử dụng toán tử if dưới dạng sau:

```
if(expression_1)
   statement_1;
else if (expression_2)
   statement_2;
..
else if (expression_n-1)
   statement_n-1;
else
   statement_n;
```

2. Cấu trúc điều khiển If (Dạng 2)

- Ví dụ:

```
if (ch >= '0' && ch <= '9')
  cout<<“Chu so”;
else
{
      if (ch >= 'A' && ch <= 'Z')
  cout<<“Chu in hoa”;

      else
      {
           if (ch >= 'a' && ch <=
'z')
        cout<<“Chu thuong”;
          else
        cout<<“ktu dac biet”;
       }
}
```

```
if (ch >= '0' && ch <= '9‘)
  cout<<“Chu so”;
else if (ch >= 'A' && ch <= 'Z')
cout<<“Chu in hoa”;
else if (ch >= 'a' && ch <= 'z')
  cout<<“Chu thuong”;
else
      cout<<“ktu dac biet”;
```

2. Cấu trúc điều khiển If (Dạng 2)

- ***Ví dụ 2***: Viết chương trình nhập vào một số nguyên dương là tháng trong năm và in ra số ngày của tháng đó. Biết rằng:
  - Tháng có 31 ngày: 1, 3, 5, 7, 8, 10, 12
  - Tháng có 30 ngày: 4, 6, 9, 11
  - Tháng có 28 hoặc 29 ngày : 2

2. Cấu trúc điều khiển If (Dạng 2)

- ***Ví dụ 3***: Giả sử để theo dõi trình độ cán bộ ta dùng bảng mã sau:

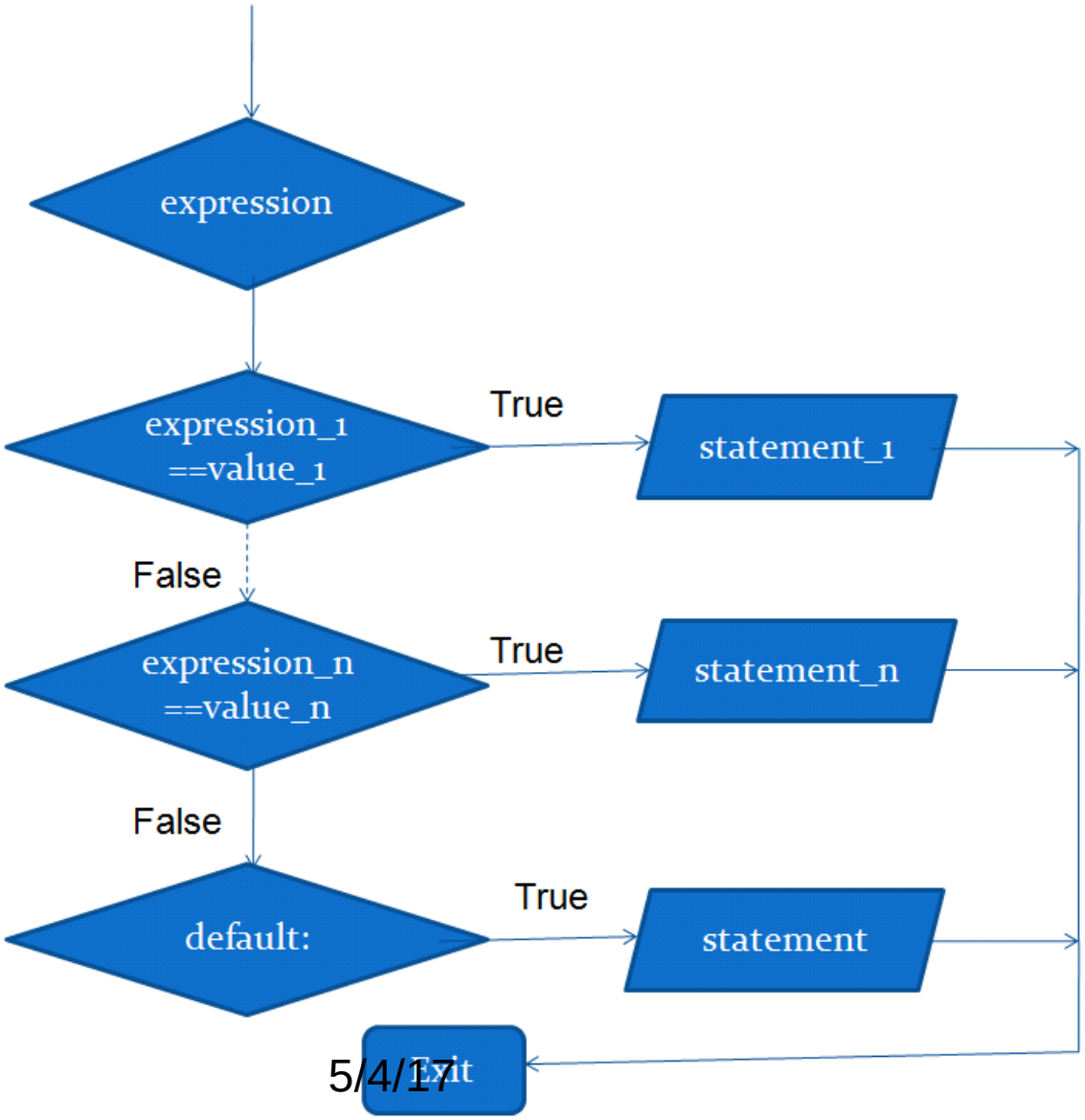
Mã	Trình độ
1	Sơ cấp
2	Trung cấp
3	Cao đẳng
4	Đại học
5	Cao học
6	Phó tiến sĩ
7	Tiến sĩ

3. Cấu trúc lựa chọn switch

- Cấu trúc switch là một cấu trúc lựa chọn có nhiều nhánh. Khi có nhiều sự lựa chọn thi đây là cấu trúc phù hợp thay vì phải dùng một chuỗi

```
switch (expression)
{
   case value_1:
      statement_1;
      [break;]
   …
   case value_n:
      statement_n;
      [break;]
   [default:
      statement;]
}
```

3.Cấu trúc lựa chọn switch



3. Cấu trúc lựa chọn switch

Giải thích:

- Trước hết chương trình se định giá trị của ***expression***.
  - Nếu giá trị của ***expression*** bằng ***value_1*** thì thực hiện ***statement_1*** rồi thoát.
  - Nếu giá trị của ***expression*** khác ***value_1*** thì so sánh với ***value_2***, nếu bằng ***value_2*** thì thực hiện ***statement_2*** rồi thoát..., so sánh tới ***value_n***.
  - Nếu tất cả các phép so sánh trên đều sai

3. Cấu trúc lựa chọn switch

Lưu ý:

- Biểu thức trong switch() phải có kết quả là giá trị kiểu ***số nguyên*** (int, char, long, short).
- Các giá trị sau case cũng phải là kiểu số nguyên.
- Không bắt buộc phải có default.
- Thông thường sau mỗi câu lệnh **statement** có 1 câu lệnh break; Khi thực hiện lệnh tương ứng của case có

Ví dụ 1: Nhập vào một số nguyên và kiểm tra xem đây là 1 số chẵn hay là số lẻ.

3. Cấu trúc lựa chọn switch

```
#include <iostream.h>
#include <conio.h>
void main ()
{
    int n, remainder;
    cout<<”Input an number: "; cin>>n;
    remainder = (n % 2);
    switch(remainder)
    {
        case 0: cout << n << ” is an even.";
        break;
        case 1: cout << n << ” is an odd."; break;
```

3. Cấu trúc lựa chọn switch

- *Ví dụ 2:* Nhập vào 2 số nguyên và 1 phép toán.
  - Nếu phép toán là '+', '-', '*' thì in ra kết qua là tổng, hiệu, tích của 2 số.
  - Nếu phép toán là '/' thì kiểm tra xem số thứ 2 có khác không hay không? Nếu khác không thì in ra thương của chúng, ngược lại thì in ra thông báo "Cannot divide by zero!".
- *Ví dụ 3:* Yêu cầu người thực hiện chương trình nhập vào một số nguyên dương là tháng trong năm và in ra số ngày của tháng đó.

4. Cấu trúc lặp (Loop structures)

- Cấu trúc lặp cho phép lặp đi lặp lại nhiều lần 1 câu lệnh hay 1 khối lệnh nào đó cho đến khi biểu thức điều kiện còn thỏa.
- Các loại cấu trúc lặp:
  - Cấu trúc while
  - Cấu trúc do .. while
  - Cấu trúc for

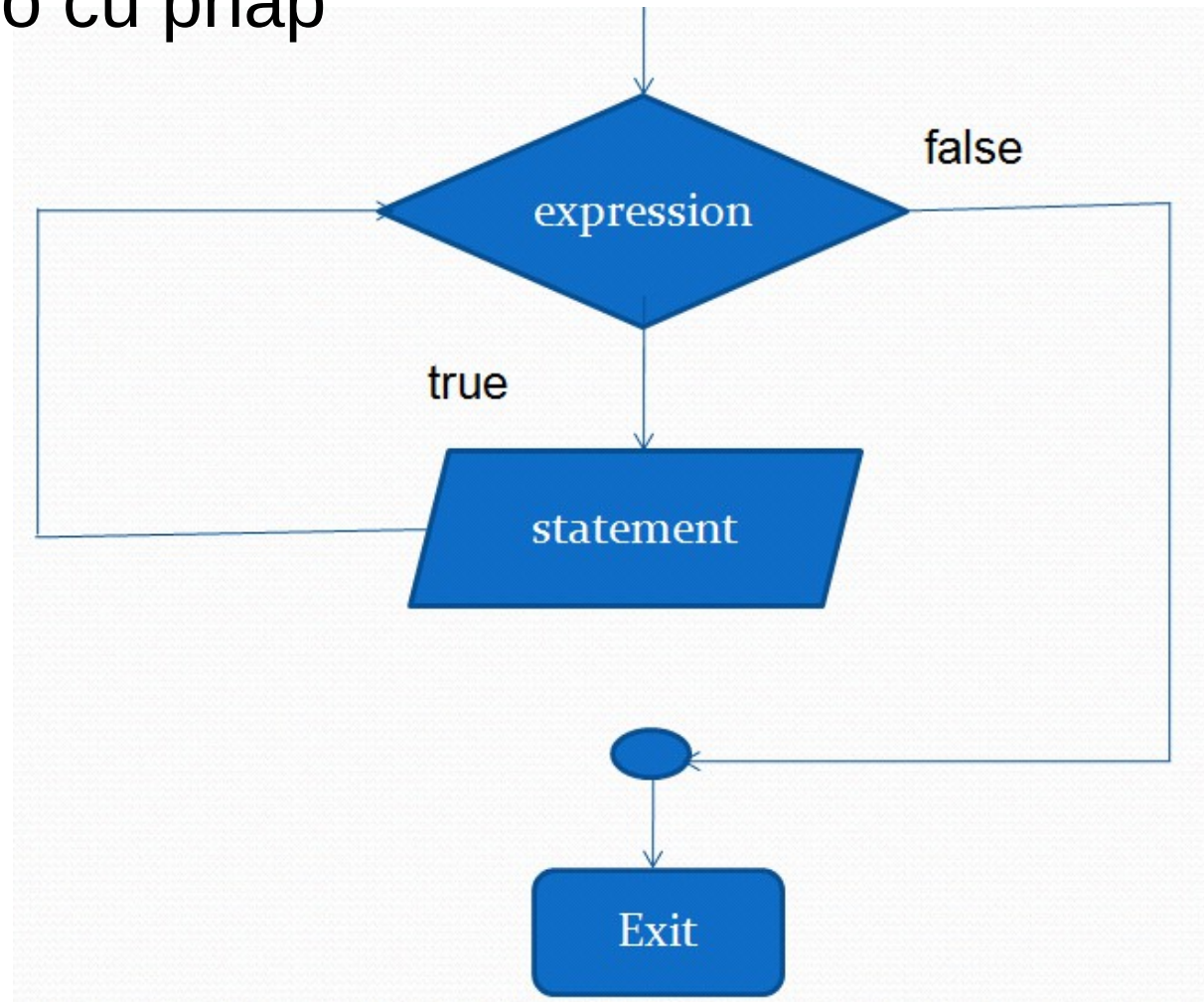
4.1 Cấu trúc while

- Cấu trúc while cho phép thực hiện statement trong khi expression vẫn còn đúng
- Cú pháp

```
while (expression)
    statement;
```

4.1 Cấu trúc while

- Lưu đồ cú pháp



4.1 Cấu trúc while

- *Ví dụ 1*: viết chương trình tính tổng của tất cả các số nguyên từ 1 tới n.
- *Ví dụ 2:* viết chương trình in ra tất cả các số nguyên từ 1 tới 10.
- **Lưu ý:** vòng lặp phải kết thúc ở một điểm nào đó, vì vậy bên trong vòng lặp phải cung cấp một phương thức nào đó để buộc ***<biểu thức điều kiện>*** trở thành sai nếu không thì chương trình sẽ lặp vô tận

4.2 Cấu trúc do … while

- Cấu trúc do… while dùng để lặp 1 statement trong khi expression là true. Statement luôn được thực hiện ít nhất lần

```
do
{
   statement;
} while (condition);
```

4.2 Cấu trúc do … while

- *Ví dụ 1*: Viết đoạn chương trình in dãy số nguyên từ 1 đến 10.

```
void main()
{
    int counter = 1; //khởi tạo giá trị của biến counter
    do
    {
        cout << counter << " "; // hiển thị
```

4.2 Cấu trúc do … while

Kết quả sau khi chạy chương trình:

1 2 3 4 5 6 7 8 9 10

4.2 Cấu trúc do … while

- *Ví dụ 1*: Viết đoạn chương trình in dãy số nguyên từ 1 đến 10.

```
#include <iostream.h>
#include <conio.h>
void main()
{
    int i;
    clrscr();
    cout<<”Display one to ten: ";
    i=1;
    while (i<=10)
    {
        cout<< “  ” << i;
        i++;
     }
     getch();
}
```

```
#include <iostream.h>
#include <conio.h>
void main ()
{
        int i;
        clrscr();
        cout<<"Display one to ten: ";
        i=1;
        do
        {
            cout << “  “ << i;
            i+=1;
        } while(i<=10);
        getch();
}
```

4.2 Cấu trúc do … while

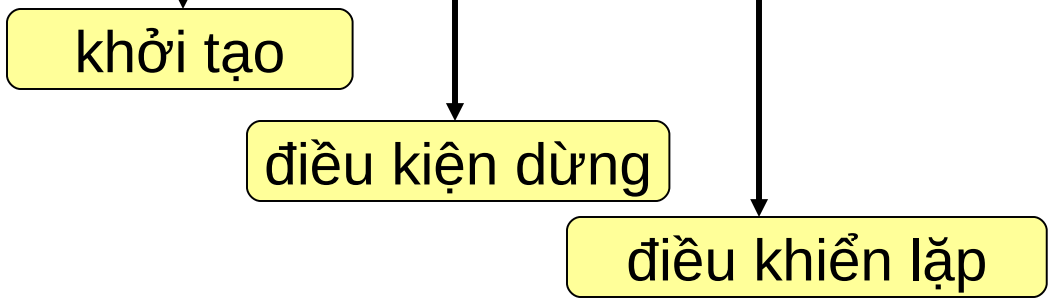
- Ví dụ 1: Viết chương trình in các số chẵn trong đoạn từ 1 đến n.
- Ví dụ 2: Viết chương trình tính tổng các số chẵn trong đoạn từ 1 đến n.

4.3. Cấu trúc lặp for:

- Chức năng chính của vòng lăp for là lặp lại một đọan lệnh nào đó đến khi nào ***exp2*** còn mang giá trị true, vòng lặp for thường sử dụng trong những chương trình mà số lần lặp lại một đoạn lệnh nào đó được biết trước.

Cú pháp

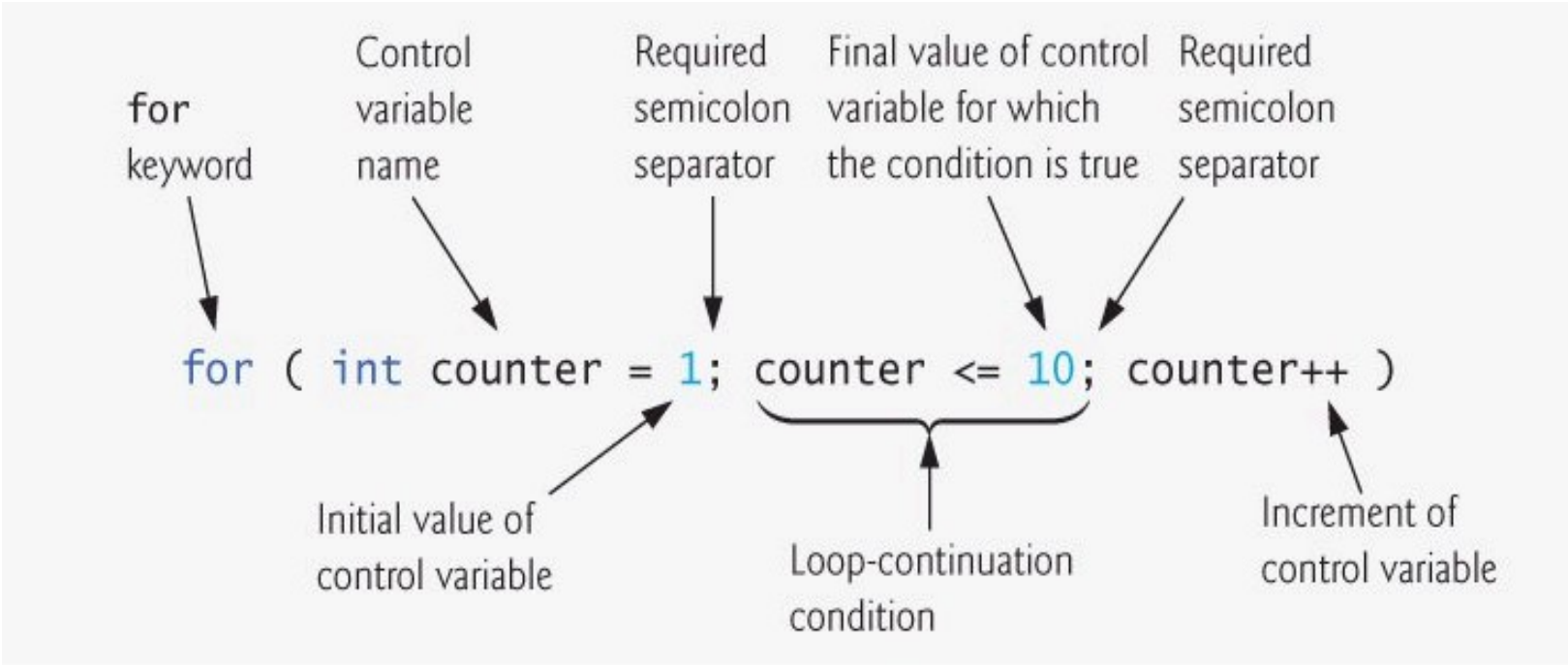
```
for (exp1 ; exp2 ; exp3)
    statements ;
```



4.3. Cấu trúc lặp for:

```
1  // Fig. 5.2: fig05_02.cpp
2  // Counter-controlled repetition with the for statement.
3  #include <iostream>
4  using std::cout;
5  using std::endl;
6
7  int main()
8  {
9     // for statement header includes initialization,
10    // loop-continuation condition and increment.
11    for ( int counter = 1; counter <= 10; counter++ )
12       cout << counter << " ";
13
14    cout << endl; // output a newline
15    return 0; // indicate successful termination
16 } // end main
```

4.3. Cấu trúc lặp for:



4.3. Cấu trúc lặp for:

- Ví dụ: Viết chương trình xuất các số nguyên từ n → 1

```
for (int i = 10; i>= 1; --i)
    cout << setw(4) << i;
```

- C++ cho phép biểu thức đầu tiên trong vòng lặp for là một định nghĩa biến.

4.3 Cấu trúc lặp for

- Bất kỳ biểu thức nào trong 3 biểu thức của vòng lặp for đều có thể rỗng. Xóa tất cả các biểu thức cho chúng ta một vòng lặp vô hạn.

```
for (;;) // vòng lặp vô hạn
{
```

4.3 Cấu trúc lặp for

- Ví dụ:
- for (int i = 1; i <= 100; i++)
- for (int i = 100; i >= 1; i--)
- for (int i = 7; i <= 77; i += 7)
- for (int i = 20; i >= 2; i -= 2)
- for (int i = 2; i <= 20; i += 3)
- for (int i = 99; i >= 0; i -= 11)

5.CÁC LỆNH RẼ NHÁNH VÀ LỆNH NHẢY

1. Lệnh break thường dùng trong phần case của cấu trúc switch để thoát khỏi cấu trúc switch sau khi các lệnh tương ứng của case đã được thực hiện.

- Trong các cấu trúc lặp, nếu muốn thoát khỏi 1 vòng lặp tức thì mà không chờ cho đến khi biểu thức điều kiện (conditional expression) của cấu trúc được định trị là false, ta dùng lệnh break.
- Lệnh break thường liên đới với một câu lệnh if trong những trường hợp này. Không sử dụng lệnh break bên ngoài các cấu trúc lặp

5.CÁC LỆNH RẼ NHÁNH VÀ LỆNH NHẢY

Ví dụ 1:

```
1  // Fig. 5.13: fig05_13.cpp
2  // break statement exiting a for statement.
3  #include <iostream>
4  using std::cout;
5  using std::endl;
6
7  int main()
8  {
9     int count; // control variable also used after loop terminates
10
11    for ( count = 1; count <= 10; count++ ) // loop 10 times
12    {
13       if ( count == 5 )
14          break; // break loop only if x is 5
15
16       cout << count << " ";
17    } // end for
18
19    cout << "\nBroke out of loop at count = " << count << endl;
20    return 0; // indicate successful termination
21 } // end main
```

Output: **1 2 3 4**

5.1 Break

- **Ví dụ 2:** Viết chương trình tính tổng các số nguyên được nhập từ bàn phím, chương trình được kết thúc khi nhập số âm.

```
void main()
{
    int num,sum=0;
    clrscr();
    while(1)
    {
        cout <<"Input a number (negative for exit): ";
        cin >> num;
        if (num < 0) break;
        sum += num;
    }
cout<<"\nSum all inputs is " << sum;
```

5.2 Lệnh “continue”

- Trái ngược với lệnh “break”, lệnh “continue” dùng để bắt đầu một vòng mới của chu trình bên trong nhất chứa nó.

5.2 Lệnh “continue”

```
1  // Fig. 5.14: fig05_14.cpp
2  // continue statement terminating an iteration of a for statement.
3  #include <iostream>
4  using std::cout;
5  using std::endl;
6
7  int main()
8  {
9     for ( int count = 1; count <= 10; count++ ) // loop 10 times
10    {
11       if ( count == 5 ) // if count is 5,
12          continue;      // skip remaining code in loop
13
14       cout << count << " ";
15    } // end for
16
17    cout << "\nUsed continue to skip printing 5" << endl;
18    return 0; // indicate successful termination
19 } // end main
```

Output: 1 2 3 4 6 7 8 9 10

5.2 Lệnh “continue”

Ví dụ: một vòng lặp thực hiện đọc một số, xử lý nó nhưng bỏ qua những số âm, và dừng khi số là 0, có thể diễn giải như sau:

Cách 1:

```
do
{
    cin >> num;
    if (num < 0) continue;
    {
       // process num here
    }
} while (num != 0);
```

Cách 2:

```
do
{
    cin >> num;
    if (num >= 0)
    {
     // process num here
    }
} while (num != 0);
```