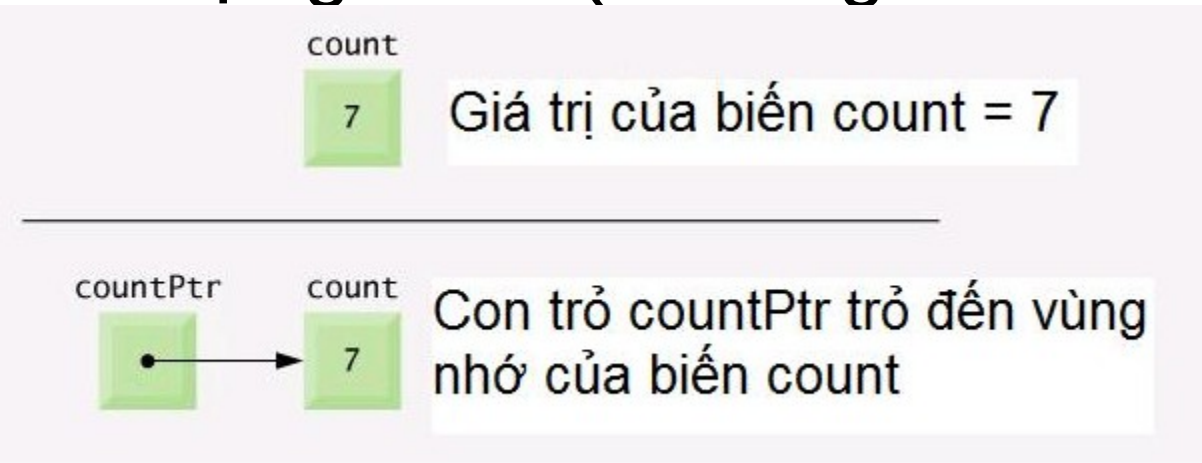
Chương 5: con trỏ (pointer)

Agenda

- Khái niệm về con trỏ
- Toán tử con trỏ
- Thao tác trên con trỏ
- Con trỏ void
- Con trỏ null
- Mảng con trỏ
- Cấp phát bộ nhớ động

1. Khái niệm về con trỏ

- Con trỏ là 1 biến chứa 1 địa chỉ bộ nhớ. Địa chỉ này là vị trí của 1 đối tượng khác (thường là 1 biến) trong



1. Khái niệm về con trỏ

- Giống như tất cả các biến, con trỏ phải được khai báo trước khi sử dụng.
- Dạng tổng quát để khai báo 1 biến

type *pointerName;

  - type: Kiểu dữ liệu của biến mà con trỏ trỏ đến

2. Toán tử con trỏ (pointer operation)

- Có 2 toán tử con trỏ là * và &
- Toán tử &: là toán tử 1 ngôi trả về địa chỉ bộ nhớ của toán hạng của nó.
- Cú pháp:

&variableName

- Ví dụ:

int y = 50; // Khai báo biến y

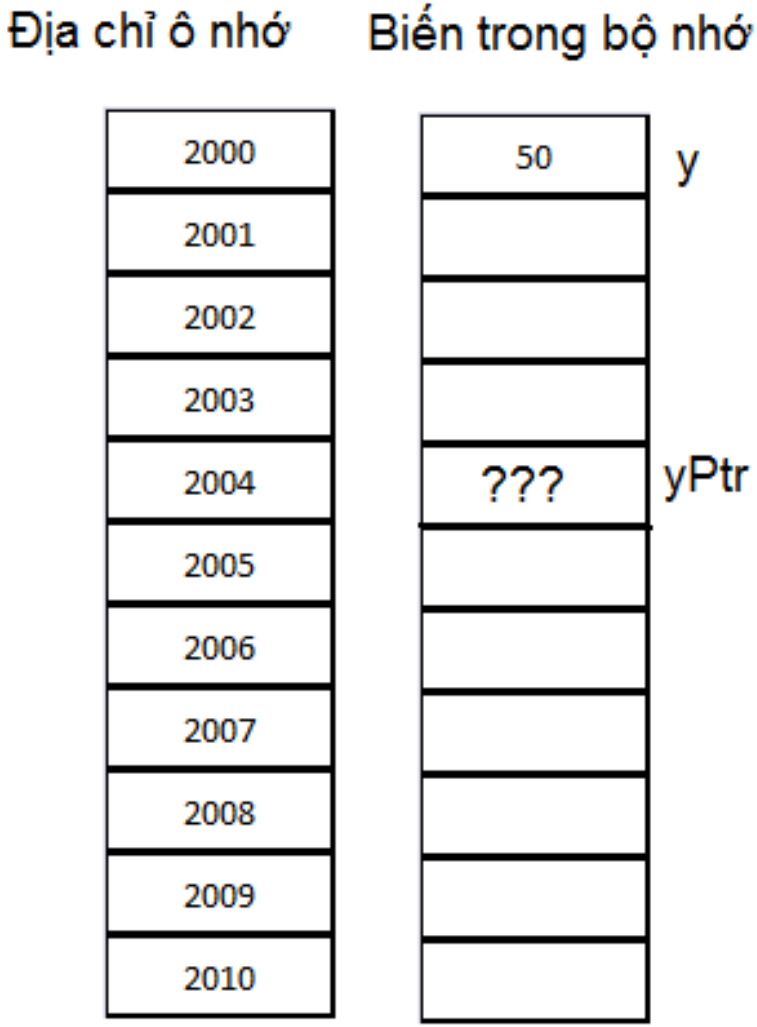
2. Toán tử con trỏ (pointer operation)

int y = 50;

int *yPtr;

yPtr = &y;

➔ **yPtr = ???**



2. Toán tử con trỏ (pointer operation)

- Toán tử * : là toán tử một ngôi trả về giá trị tại địa chỉ con trỏ trỏ đến
- Cú pháp:

```
*pointerName
```

- Ví dụ:

```
int y = 50; // Khai báo biến y
int *yPtr; // Khai báo con trỏ yPtr
cout << *yPtr; // in ra giá tri của
```

2. Toán tử con trỏ (pointer operation)

- Ví dụ 1:

```
#include <iostream>
int main()
{
   int a;
   int *aPtr; // aPtr is an int * -- pointer to an
integer
   a = 7;
   aPtr = &a;
   cout << endl << &a
        << endl << aPtr;
   cout << endl <<  a
        << endl << *aPtr;
   cout << "\n&*aPtr = " << &*aPtr
        << "\n*&aPtr = " << *&aPtr << endl;
   return 0;
} // end main
```

2. Toán tử con trỏ (pointer operation)

- Ví dụ 2:

```
#include <iostream>
int main ()
{
    int firstvalue, secondvalue;
    int * mypointer;
    mypointer = &firstvalue;
    *mypointer = 20;
    mypointer = &secondvalue;     *mypointer = 10;
    cout << “1st value is " << firstvalue;
    cout << “2nd value is " << secondvalue ;
    return 0;
}
```

2. Toán tử con trỏ (pointer operation)

- Ví dụ 3:

```
#include <iostream>
void main ()
{
  int firstvalue = 5, secondvalue = 15;
  int * p1, * p2;
  p1 = &firstvalue;
  p2 = &secondvalue;

  *p1 = 10;// value pointed by p1 = 10
  *p2 = *p1;

  p1 = p2; //value of pointer is copied
  *p1 = 20; // value pointed by p1 = 20
  cout << "firstvalue is " << firstvalue ;
  cout << "secondvalue is " << secondvalue ;
}
```

3. Các thao tác trên con trỏ

- **Lệnh gán con trỏ**: dùng phép gán để gán giá trị của một con trỏ cho một con trỏ khác có cùng kiểu
- **Ví dụ:**

```
int x;
int *p1, *p2;
p1 = &x;
p2 = p1;
```

- Sau khi đoạn lệnh trên được thực

3. Các thao tác trên con trỏ

- **Phép toán số học trên con trỏ**:
  Chỉ có 2 phép toán sử dụng trên con trỏ là phép cộng và phép trừ.

3. Các thao tác trên con trỏ

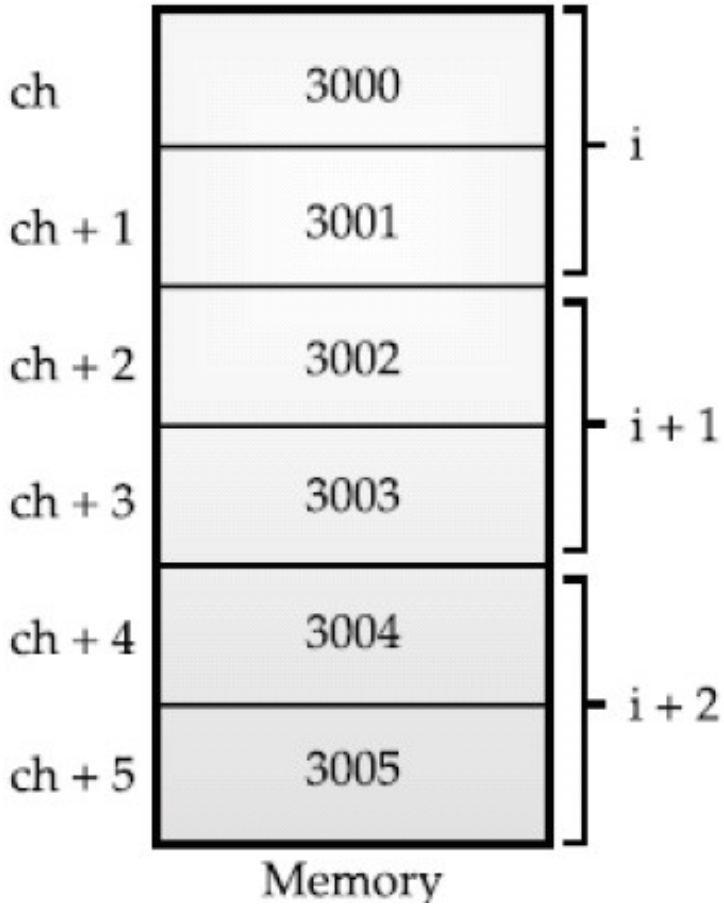
- **Tất cả con trỏ sẽ tăng hay giảm với đơn vị là kích thước của kiểu dữ liệu của nó.**

```
char *ch;

int *i;

ch = ch + 3;
```

➔ ch = ???

```
i = i + 2;
```

➔ i = ???



3. Các thao tác trên con trỏ

- **Ví dụ :**
  - char *a;
  - short *b;
  - long *c;
- Các con trỏ a, b, c lần lượt trỏ tới ô nhớ

1000, 2000 và 3000.

- Cộng các con trỏ với một số nguyên:
  - a = a + 1; //con trỏ a dời đi ? byte

4. con trỏ Void (void pointer)

- Ví dụ:
- int a, *pa;
- float f, *pf;
- pa = &a; // hợp lệ
- pf = &f; // hợp lệ
- **pa = &f; // không hợp lệ** vì pa là con trỏ kiểu int nên chỉ chứa địa chỉ của biến kiểu int
- pf = &a; //không hợp lệ vì pf là con trỏ kiểu float nên chỉ

4. con trỏ Void (void pointer)

- Con trỏ void là một loại con trỏ đặc biệt mà có thể trỏ đến bất kỳ kiểu dữ liệu nào.
- C

```
void *pointerVariable;
```

- Ví dụ:

```
void *p;
p = &a; // con trỏ p trỏ đến biến nguyên
a
```

4. con trỏ Void (void pointer)

- Tùy thuộc con trỏ void đang trỏ đến kiểu dữ liệu nào, ta phải ép về đúng kiểu tương ứng khi dùng trong các biểu thức

- Ví dụ:

- Nếu p đang trỏ đến biến nguyên a, để tăng giá trị của biến a lên 10 ta phải dùng lệnh sau:

***(int*)p += 10;**

5. Mảng & con trỏ

- Giữa mảng và con trỏ có 1 mối quan hệ tương quan. Tên của 1 mảng có thể được xem như là 1 con trỏ.
  - int b[5]; // khai báo 1 mảng kiểu int 5 phần tử
  - int *bPtr; // khai báo 1 con trỏ trỏ đến kiểu int

- Tên của một mảng tương ứng 1 con trỏ trỏ đến phần tử đầu tiên trong mảng.
  - bPtr = b; // gán địa chỉ của mảng b cho con trỏ bPtr

5. Mảng con trỏ

```
#include <iostream.h>
#include <conio.h>
void main ()
{
int numbers[5], * p;
p = numbers; *p = 10;
p++; *p = 20;
p = &numbers[2]; *p = 30;
p = numbers + 3; *p = 40;
```

5. con trỏ null (null pointer)

- Một con trỏ hiện hành không trỏ đến một địa chỉ bộ nhớ hợp lệ thì được gán giá trị NULL.
- Nói các khác, con trỏ NULL là con trỏ không trỏ đến đâu cả.
- NULL được định nghĩa trong <stdlib>
- **Ví dụ:**

```
#include <iostream.h>

void main()
{
```

6. Cấp phát vùng nhớ động

- Cấp phát động là phương tiện nhờ đó chương trình có thể dành thêm bộ nhớ trong lúc đang thực thi.
- Cấp phát động trong C++
  - C++ cung cấp 2 toán tử dùng trong việc cấp phát và thu hồi bộ nhớ là **new** và **delete**

6. Cấp phát vùng nhớ động

- **Toán tử new**: cấp phát bộ nhớ và trả về 1 con trỏ đến byte đầu tiên trong vùng nhớ được cấp phát
  - **Cú pháp**

```
pointerName = new datatype;
```

  - datatype : Kiểu dữ liệu của con trỏ pointer

```
pointerName = delete;
```

- **Toán tử delete**: Thu hồi vùng nhớ

6. Cấp phát vùng nhớ động

- Ví dụ:

```
#include <iostream.h>
#include <new>
int main()
{
    int *p;
    p = new int; // allocate space for an
int
    *p = 100;
```

7. Mảng con trỏ

- Mỗi biến con trỏ là một biến đơn. Ta có thể tạo mảng của các con trỏ với mỗi phần tử của mảng là một con trỏ.
- **Cú pháp:**

```
type *pointerArray[elements];
```

  - **type: kiểu dữ liệu mà các con trỏ phần tử** trỏ đến.
  - **pointerArray: tên mảng con trỏ.**
  - **elements: số phần tử của mảng con trỏ**

7. Mảng con trỏ

- int *p[5];
- int a=6;
- p[0] = &a;
- p[2] = p[0];
- int b;
- b = *p[0];