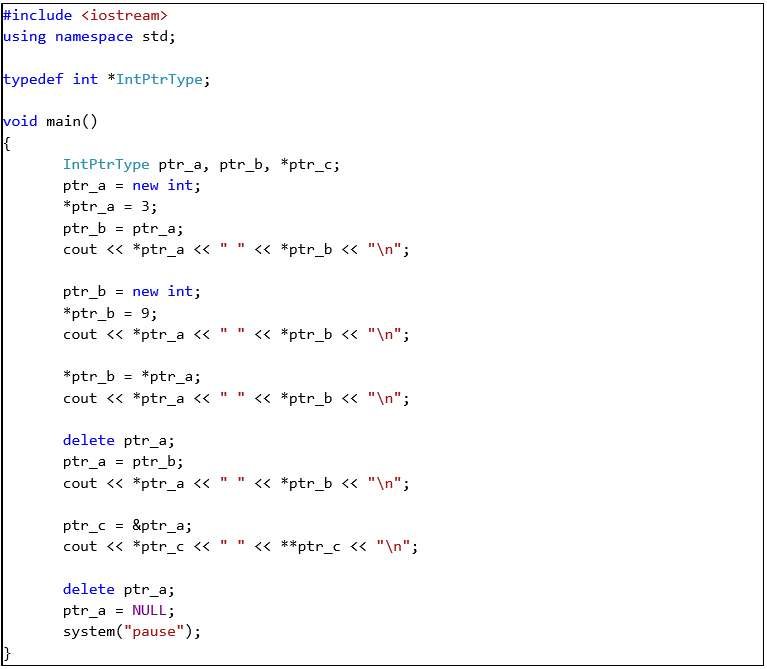
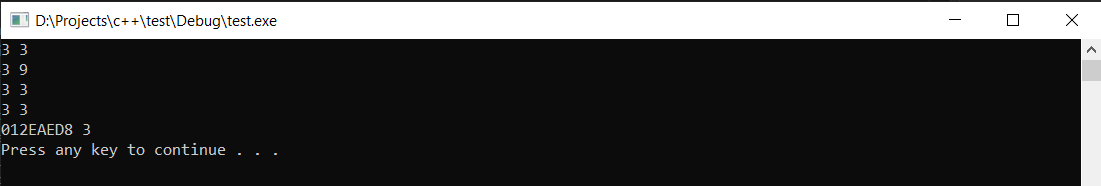
**Practice2\_Pointer**

1. **POINTER**
2. **Explain the result of following program**



Sau khi chạy chương trình trên:

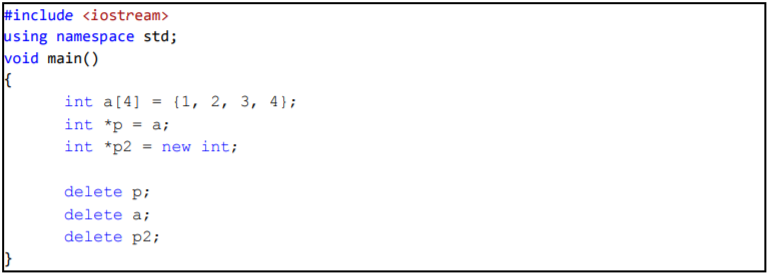


Giải thích:

**IntPtrType** là một biến được đinh nghĩa là biến con trỏ trỏ đến biến kiểu **int**.

* Dòng 1, **3 3**: **ptr\_a** trỏ đến 1 vùng nhớ được cấp phát động kiểu **int** (4 bytes) sau đó được gán giá trị bằng **3**, **\*ptr\_a = 3**. Địa chỉ vùng nhớ **ptr\_a** trỏ tới được gán cho **ptr\_b** nên b cũng sẽ trỏ tới vùng nhớ 4 bytes có giá trị bằng **3**, **\*ptr\_b = 3**.
* Dòng 2, **3 9**: Giá trị vùng nhớ **ptr\_a** trỏ tới không thay đổi, **\*ptr\_a = 3**. **ptr\_b** trỏ đến 1 vùng nhớ mới được cấp phát động kiểu **int** và được gán giá trị bằng **9**, **\*ptr\_b = 9**.
* Dòng 3, **3 3**: Giá trị vùng nhớ **ptr\_a** trỏ tới không thay đổi, **\*ptr\_a = 3**. Giá trị vùng nhớ **ptr\_b** trỏ tới được gán bằng giá trị vùng nhớ mà **ptr\_a** trỏ tới bằng **3**, **\*ptr\_b = 3**.
* Dòng 4, **3 3**: **ptr\_a** trả lại quyền sử dụng bộ nhớ được cấp phát trước đó cho hệ điều hành sau đó được trỏ đến cùng vùng nhớ với **ptr\_b**, **\*ptr\_a = \* ptr\_b = 3**.
* Dòng 5, **012EAED8 3**: **ptr\_c** là con trỏ trỏ đến con trỏ **ptr\_a**, **\*ptr\_c = &ptr\_a =** **012EAED8** (địa chỉ của con trỏ ptr\_a). **\*\*ptr\_c = \*ptr\_a = 3** (giá trị vùng nhớ con trỏ ptr\_a trỏ tới).

1. **Detect and solve problems of following program**



* Lỗi nằm ở 2 dòng lệnh: **delete p;** và **delete a;**
* Giải thích: cả 2 cấu lệnh **delete p;** và **delete a;** đều được sử dụng để trả quyền sử dụng vùng nhớ cho hệ điều hành trong trường hợp vùng nhớ nó trỏ tới được cấp phát động, việc sử dung **delete** cũng chỉ trả lại quyền sử dụng vùng nhớ cho phần từ **a[0]** chứ không phải toàn bộ mảng. Vì cả **p** và **a** đều trỏ tới cùng 1 nơi nên không thể sử dụng cả 2 lệnh này cùng lúc (chỉ cần sử dụng 1 trong 2). Ở đây thì mảng được cấp phát tĩnh (dữ liệu lưu trong vùng nhớ **Stack**).
* Giải quyết:
* **Cách 1:** xóa 2 dòng lệnh trên và khi chương trình kết thúc sẽ tự giải phóng vùng nhớ chứa mảng a.
* **Cách 2:** cấp phát động cho **mảng a** (a được khai báo kiểu con trỏ) và chỉ sử dụng 1 trong 2 câu lệnh **delete [] p;** hoặc **delete [] a;** để trả quyền sử dụng vùng nhớ chứa dữ liệu mảng a cho hệ điều hành cũng như gọi hàm **destructor** cho object được tạo bởi operator **new[]**.

1. **Using pointer:**

* Why should we use **delete**?

Chúng ta nên sử dụng **delete** để trả lại quyền sử dụng vùng nhớ đã xin cấp phát trước đó cho hệ điều hành khi không cần sử dụng đến nữa, tránh lãng phí bộ nhớ trong trường hợp vùng nhớ được cấp phát động.

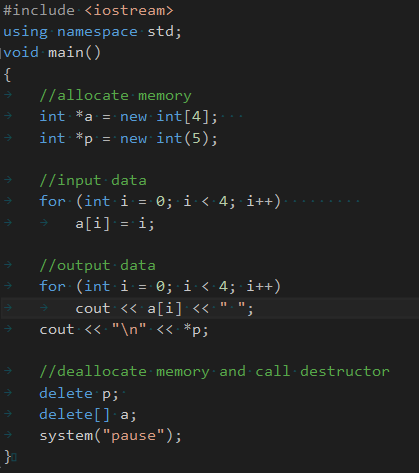
* When we use **delete**?

Chúng ta sử dụng **delete** khi vùng nhớ được cấp phát động thông qua việc gọi toán từ **new** và không cần sử dụng vùng nhớ đó nữa.

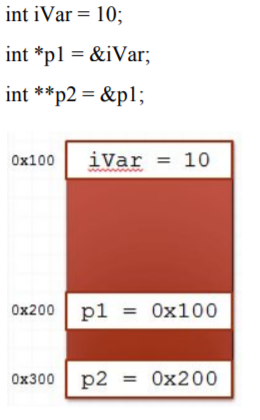
* Difference between **delete** and **delete[]**. Write a demo

|  |  |  |
| --- | --- | --- |
|  | **delete** | **delete []** |
| Gọi hàm hủy cho đối tượng được tạo bởi | **new** | **new[]** |

Demo:



1. **Given below code**



What are the result?

1. \*p1 == 10
2. \*p2 == 0x100
3. \*(\*p2) == 10