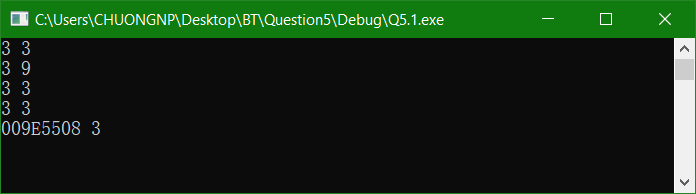
# Question 5: Pointer

**Q5.1: Explain the result**

|  |
| --- |
| #include <stdio.h>  #include <iostream>  using namespace std;  typedef int \*IntPtrType;  int main()  {  IntPtrType ptr\_a, ptr\_b, \*ptr\_c;  ptr\_a = new int;  \*ptr\_a = 3;  ptr\_b = ptr\_a;  cout << \*ptr\_a << " " << \*ptr\_b << "\n";  ptr\_b = new int;  \*ptr\_b = 9;  cout << \*ptr\_a << " " << \*ptr\_b << "\n";  \*ptr\_b = \*ptr\_a;  cout << \*ptr\_a << " " << \*ptr\_b << "\n";  delete ptr\_a;  ptr\_a = ptr\_b;  cout << \*ptr\_a << " " << \*&\*&\*&\*&\*ptr\_b << "\n";  ptr\_c = &ptr\_a;  cout << \*ptr\_c << " " << \*\*ptr\_c << "\n";  delete ptr\_a;  ptr\_a = NULL;  return 0;  } |

**Kq:**



typedef int \*IntPtrType; -> IntPtrType là kiểu con trỏ int

IntPtrType ptr\_a, ptr\_b, \*ptr\_c; -> ptr\_a, ptr\_b là con trỏ cấp 1, ptr\_c là con trỏ cấp 2 kiểu int.

ptr\_a = new int;

Cấp phát vùng nhớ cho con trỏ ptr\_a.

\*ptr\_a = 3;

Gán giá trị cho vùng nhớ mà ptr\_a trỏ tới là 3.

ptr\_b = ptr\_a;

Gán địa chỉ vùng nhớ mà a đang nắm giữ cho b.

cout << \*ptr\_a << " " << \*ptr\_b << "\n";

-> kq in ra là 3 3 như trên (do ptr\_a, ptr\_b cùng trỏ tới một địa chỉ có nội dung là 3).

ptr\_b = new int;

\*ptr\_b = 9;

cout << \*ptr\_a << " " << \*ptr\_b << "\n";

b được cấp phát một vùng nhớ mới và được gán nội dung cho vùng nhớ này = 9 -> kq in ra là 3 9.

\*ptr\_b = \*ptr\_a;

cout << \*ptr\_a << " " << \*ptr\_b << "\n";

Nội dung của ptr\_b = nội dung của ptr\_a => kq 3 3 (\*ptr\_a = 3).

delete ptr\_a;

ptr\_a = ptr\_b;

cout << \*ptr\_a << " " << \*&\*&\*&\*&\*ptr\_b << "\n";

Thu hồi vùng nhớ mà ptr\_a trỏ đến, sau đó gán địa chỉ vùng nhớ mà ptr\_b nắm giữ cho ptr\_a, mà nội dung ptr\_b = 3 nên kq in ra la 3 3. Phần bôi đỏ của \*&\*&\*&\*&\*ptr\_b lấy nội dung của địa chỉ của nội dung .... nội dung của ptr\_b tức là lấy nội dung của ptr\_b.

ptr\_c = &ptr\_a;

cout << \*ptr\_c << " " << \*\*ptr\_c << "\n";

ptr\_c (con trỏ cấp 2) được gán địa chỉ của con trỏ ptr\_a. do đó, \*ptr\_c là địa chỉ của ptr\_a (vì ptr\_c là con trỏ cấp 2), \*\*ptr\_c là nội dung cùng nhớ mà địa chỉ ptr\_c trỏ tới (chính xác là nội dung mà ptr\_a trỏ tới vì ptr\_c trỏ tới ptr\_a). => kq: địa chỉ 1 vùng nhớ 3 (009E5508 3).

**Q5.2: Introduce int variables x and y and int\* pointer variables p and q. Set x to 2, y to 8, p to the address of x, and q to the address of y. Then print the following information:**

1. **The address of x and the value of x.**
2. **The value of p and the value of \*p.**
3. **The address of y and the value of y.**
4. **The value of q and the value of \*q.**
5. **The address of p (not its contents!).**
6. **The address of q (not its contents!).**

**Use the Hex function to print all pointer/address values and format the output so it is easy to make comparisons.**

**Code:**

#include<iostream>

using namespace std;

int main() {

int x = 2, y = 8;

int \*p = &x, \*q = &y;

//1. the address of x and the value of x

cout << "&x = 0x" << &x <<" x = " << x << endl;

//2. the calue of p and the value of \*p

cout << "p = 0x" << p << " \*p = " << \*p << endl;

//3. the address of y and the value of y

cout << "&y = 0x" << &y << " y = " << y << endl;

//4. the calue of q and the value of \*q

cout << "q = 0x" << q << " \*q = " << \*q << endl;

//5. the address of p (not its contents!)

cout << "&p = 0x" << &p << endl;

//6. thw address of q (not its contents!)

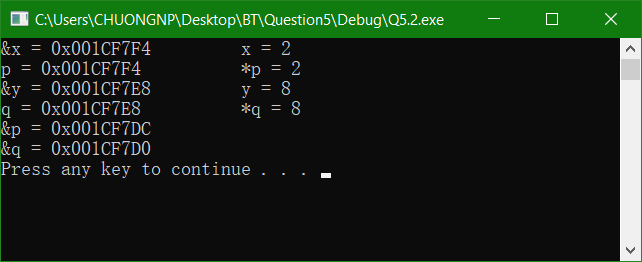
cout << "&q = 0x" << &q << endl;

system("pause");

return 0;

}

**Kq:**



**Q5.3: Introduce int variables x, y, z and int\* pointer variables p, q, r. Set x, y, z to three distinct values. Set p, q, r to the addresses of x, y, z respectively.**

1. **Print with labels the values of x, y, z, p, q, r, \*p, \*q, \*r.**
2. **Print the message: Swapping values.**
3. **Execute the swap code: z = x; x = y; y = z;**
4. **Print with labels the values of x, y, z, p, q, r, \*p, \*q, \*r.**

**Draw diagrams to explain the results.**

**Code:**

#include<iostream>

#include<iomanip>

using namespace std;

int main() {

int x = 3, y = 8, z = 6;

int \*p = &x, \*q = &y, \*r = &z;

cout << "x = " << setw(15)<<left<<x

<< "y = " << setw(15) << left << y

<< "z = " << setw(15) << left << z << endl;

cout << "p = " << setw(15) << left

<< p << "q = " << setw(15) << left

<< q << "r = " << setw(15) << left << r << endl;

cout << "\*p = " << setw(14) << left

<< \*p << "\*q = " << setw(14) << left

<< \*q << "\*r = " << setw(14) << left << \*r << endl;

cout << "\nSwapping values: z = x ; x = y ; y = z\n" << endl;

z = x; x = y; y = z;

cout << "x = " << setw(15) << left

<< x << "y = " << setw(15) << left

<< y << "z = " << setw(15) << left << z << endl;

cout << "p = " << setw(15) << left

<< p << "q = " << setw(15) << left

<< q << "r = " << setw(15) << left << r << endl;

cout << "\*p = " << setw(14) << left

<< \*p << "\*q = " << setw(14) << left

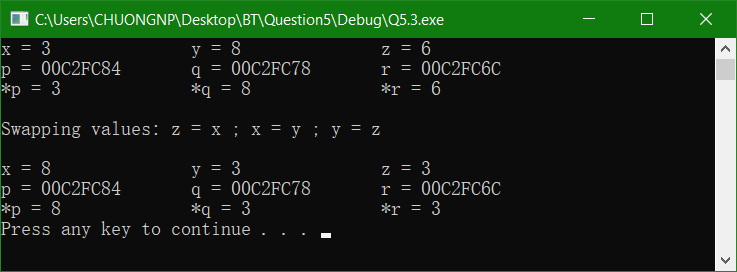
<< \*q << "\*r = " << setw(14) << left << \*r << endl;

system("pause");

return 0;

}

**KQ:**



Lúc khởi tạo:

6

8

3

00C2FC6

00C2FC78

00C2FC84

z

y

x

r

q

p

Thực hiện lệnh : z = x;

3

8

3

00C2FC6

00C2FC78

00C2FC84

z

y

x

r

q

p

Thực hiện lệnh : x = y;

3

8

8

00C2FC6

00C2FC78

00C2FC84

z

y

x

r

q

p

Thực hiện lệnh : y = z;

3

3

8

00C2FC6

00C2FC78

00C2FC84

z

y

x

r

q

p

**Q5.4: Introduce int variables x, y, z and int\* pointer variables p, q, r. Set x, y, z to three distinct values. Set p, q, r to the addresses of x, y, z respectively.**

1. **Print with labels the values of x, y, z, p, q, r, \*p, \*q, \*r.**
2. **Print the message: Swapping pointers.**
3. **Execute the swap code: r = p; p = q; q = r;**
4. **Print with labels the values of x, y, z, p, q, r, \*p, \*q, \*r.**

**Draw diagrams to explain the results.**

**Code:**

#include<iostream>

#include<iomanip>

using namespace std;

int main() {

int x = 3, y = 8, z = 6;

int \*p = &x, \*q = &y, \*r = &z;

cout << "x = " << setw(15) << left << x

<< "y = " << setw(15) << left << y

<< "z = " << setw(15) << left << z << endl;

cout << "p = " << setw(15) << left

<< p << "q = " << setw(15) << left

<< q << "r = " << setw(15) << left << r << endl;

cout << "\*p = " << setw(14) << left

<< \*p << "\*q = " << setw(14) << left

<< \*q << "\*r = " << setw(14) << left << \*r << endl;

cout << "\nSwapping values: r = p; p = q; q = r;\n" << endl;

r = p; p = q; q = r;

cout << "x = " << setw(15) << left

<< x << "y = " << setw(15) << left

<< y << "z = " << setw(15) << left << z << endl;

cout << "p = " << setw(15) << left

<< p << "q = " << setw(15) << left

<< q << "r = " << setw(15) << left << r << endl;

cout << "\*p = " << setw(14) << left

<< \*p << "\*q = " << setw(14) << left

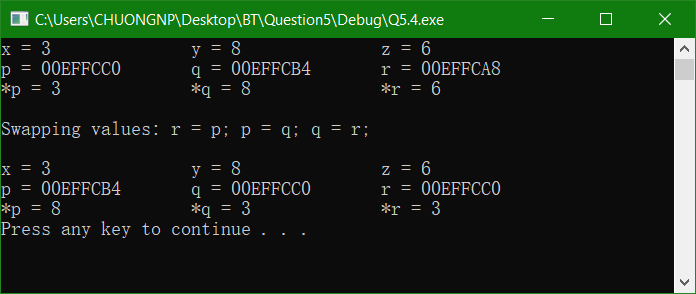
<< \*q << "\*r = " << setw(14) << left << \*r << endl;

system("pause");

return 0;

}

**KQ:**



Lúc khởi tạo:

z

y

x

r

q

p

3

00EFFCC0

00EFFCC0

00EFFCB4

8

00EFFCB4

00EFFCA8

6

00EFFCA8

r = p;

3

z

y

x

r

q

p

00EFFCC0

00EFFCC0

00EFFCB4

8

00EFFCB4

00EFFCA8

6

00EFFCC0

p = q;

3

y

x

r

q

p

00EFFCC0

00EFFCB4

00EFFCB4

8

00EFFCB4

00EFFCA8

z

6

00EFFCC0

3

y

x

r

q

p

q = r;

00EFFCC0

00EFFCB4

00EFFCB4

8

00EFFCC0

00EFFCA8

z

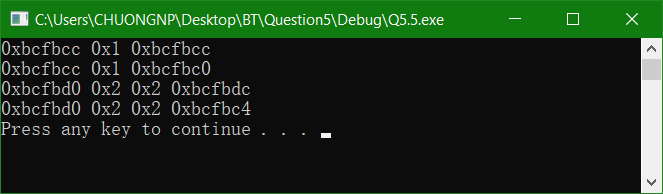
6

00EFFCC0

**Q5.5: Draw diagrams to explain the results:**

|  |
| --- |
| #include <iostream>  using namespace std;  void main()  {  int a[4] = {1, 2, 3, 4};  int \*p = a;  printf("0x%x 0x%x 0x%x\n", a, \*a, &a);  printf("0x%x 0x%x 0x%x\n", p, \*p, &p);  printf("0x%x 0x%x 0x%x 0x%x\n", (a + 1), \*(a + 1), \*a + 1, &a + 1);  printf("0x%x 0x%x 0x%x 0x%x\n", (p + 1), \*(p + 1), \*p + 1, &p + 1);  } |

**KQ:**



|  |  |
| --- | --- |
| 0xbcfbdc |  |
| 0xbcfbd8 | 4 |
| 0xbcfbd4 | 3 |
| 0xbcfbd0 | 2 |
| 0xbcfbcc | 1 |
|  |  |
| 0xbcfbc4 |  |
| 0xbcfbc0 | p |

&p

&aa

&a+1a

a

&p+1a

a+1

p+1

**Q5.6: Detect and solve problems of following program:**

|  |
| --- |
| void main()  {  int a[4] = {1, 2, 3, 4};  int \*p = a;  int \*p2 = new int;    delete p;  delete a;  delete p2;  } |

Problem: a và p không được cấp phát bộ nhớ động nhưng lại dùng toán tử delete để thu hồi vùng nhớ -> lỗi.

Solve: không cần thu hồi vùng nhớ của a và p.

void main()

{

int a[4] = {1, 2, 3, 4};

int \*p = a;

int \*p2 = new int;

delete p2;

}

**Q5.7:**

* Why should we use delete?
* When we use delete?
* Difference between delete and delete[]. Write a demo.
* Sử dụng toán tử delete để thư hồi bộ nhớ được cấp phát động bởi toán tử new.
* Sử dụng delete khi cần thu hồi lại cùng nhớ đã cấp phát trước đó.
* Sự khác nhau giữa delete và delete[] là:

+) delete dùng để thu hồi vùng nhớ của biến được cấp phát động.

+) delete[] thu hồi một mảng được cấp phát động.

+) Demo:

#include<iostream>

using namespace std;

int main() {

int \*a, \*b;

a = new int;

b = new int[10];

\*a = 1;

b[0] = 1;

delete a;

delete[]b;

return 0;

}

**Q5.8: Detect and solve problems of following program:**

|  |
| --- |
| #include <iostream>  using namespace std;  #define COUNT 10  #define MAX(a, b) ((a) < (b) ? (a):(b))  void main()  {  int \*p = new int[COUNT];  int a[COUNT];  for (int i = 0; i < sizeof(a); i++)  {  a[i] = i;  }  for (int i = 0; i <; i++)  {  p[i] = i;  }    for (int i = 0; i < MAX(sizeof(a), sizeof(p)); i++)  {  printf("%d %d\n", a[i], p[i]);  }  } |

Detect:

#include <iostream>

using namespace std;

#define COUNT 10

#define MAX(a, b) ((a) < (b) ? (a):(b))

// max(a,b) ?

void main()

{

int \*p = new int[COUNT];

int a[COUNT];

// truy cập ngoài vùng nhớ được cấp phát => ct dịch không báo lỗi // nhưng có thể gây lỗi khi chạy.

for (int i = 0; i < sizeof(a); i++)

{

a[i] = i;

}

// thiếu điều kiện kết thúc.

for (int i = 0; i <; i++)

{

p[i] = i;

}

for (int i = 0; i < MAX(sizeof(a), sizeof(p)); i++)

{

printf("%d %d\n", a[i], p[i]);

}

// chưa thu hồi vùng nhớ

}

Solve:

#include <iostream>

using namespace std;

#define COUNT 10

#define MAX(a, b) ((a) > (b) ? (a):(b))

void main()

{

int \*p = new int[COUNT];

int a[COUNT];

for (int i = 0; i < sizeof(a)/sizeof(int); i++)

{

a[i] = i;

}

for (int i = 0; i <COUNT; i++)

{

p[i] = i;

}

for (int i = 0; i < MAX(sizeof(a), sizeof(p))/sizeof(int); i++)

{

printf("%d %d\n", a[i], p[i]);

}

delete[]p;

}

**Q5.9: Explain the result**

|  |
| --- |
| void main()  {  int \*a = new int;  \*a = 10;  delete a;  printf("%d", \*a);  } |

Lỗi tại câu lệnh printf. Vì vùng nhớ của a đã bị thu hồi trước đó.

Solve: đưa câu lệnh delete a ra sau câu lệnh printf.

void main()

{

int \*a = new int;

\*a = 10;

printf("%d", \*a);

delete a;

}