**MSVV: 21120566**

**Họ tên: Nguyễn Hữu Thuận**

**Bài 01: Chương trình không có lỗi**

|  |  |
| --- | --- |
| int i = 2; | tạo biến i có giá trị là 2 |
| int \*j; | tạo biến con trỏ j |
| j = &i; | cho con trỏ j trỏ đến địa chỉ của biến i |
| **printf("%d\n", i);** | **in ra giá trị của biến i** |
| **printf("%d\n", j);** | **in ra giá trị của biến con trỏ j** |
| **printf("%d\n", \*j);** | **in ra giá trị ở địa chỉ mà con trỏ j đang trỏ tới (cũng là giá trị của biến i)** |
| **printf("%d\n", &j);** | **in ra địa chỉ của biến con trỏ j** |
| **printf("%d\n", &i);** | **in ra địa chỉ của biến i (cũng là giá trị của biến con trỏ j)** |

**Màn hình sẽ xuất ra:**

**2**

**6422220**

**2**

**6422216**

**6422220**

**Bài 02: Chương trình không có lỗi**

|  |  |
| --- | --- |
| int \*a = new int; | tạo biến con trỏ a và cấp phát vùng nhớ động cho nó |
| int \*b = new int; | tạo biến con trỏ b và cấp phát vùng nhớ động cho nó |
| \*a = 2; | cho giá trị mà con trỏ a đang trỏ đến bằng 2 |
| b = a; | cho địa chỉ mà con trỏ b đang trỏ tới bằng với địa chỉ mà con trỏ a đang trỏ tới |
| **printf("%d\n", \*a);** | **in ra giá trị mà con trỏ a đang trỏ đến** |
| **printf("%d\n", \*b);** | **in ra giá trị mà con trỏ b đang trỏ đến** |
| delete a; | giải phóng vùng nhớ của biến con trỏ a |
| delete b; | giải phóng vùng nhớ của biến con trỏ b |

**Màn hình sẽ xuất ra:**

**2**

**2**

**Bài 03: Chương trình không có lỗi**

|  |  |
| --- | --- |
| int x, \*p1, \*p2; | tạo biến x, biến con trỏ p1 và biến con trỏ p2 |
| x = 1; | cho giá trị của x bằng 2 |
| p1 = new int; | cấp phát vùng nhớ động cho p1 |
| \*p1 = 5; | cho giá trị mà con trỏ p1 đang trỏ đến bằng 5 |
| p2 = new int; | cấp phát vùng nhớ động cho p2 |
| \*p2 = 3; | cho giá trị mà con trỏ p2 đang trỏ đến bằng 3 |
| **printf("x=%d\n", x);** | **in ra giá trị của biến x** |
| **printf("p1=%d\n", \*p1);** | **in ra giá trị mà con trỏ p1 đang trỏ đến** |
| **printf("p2=%d\n", \*p2);** | **in ra giá trị mà con trỏ p2 đang trỏ đến** |
| x = \*p2; | cho giá trị của biến x bằng giá trị mà con trỏ p2 đang trỏ đến |
| p1 = &x; | cho p1 trỏ đến địa chỉ của biến x |
| **printf("x=%d\n", x);** | **in ra giá trị của biến x** |
| **printf("p1=%d\n", \*p1);** | **in ra giá trị mà con trỏ p1 đang trỏ đến** |
| **printf("p2=%d\n", \*p2);** | **in ra giá trị mà con trỏ p2 đang trỏ đến** |
| p1 = NULL; | cho con trỏ p1 trỏ vào hư không (void) |
| **printf("p1=%d\n", x);** | **in ra giá trị mà con trỏ p1 đang trỏ đến (không in ra gì cả vì đang trỏ vào void)** |
| p2 = 0; | cho con trỏ p2 trỏ vào hư không (void) |
| **printf("p2=%d\n", x);** | **in ra giá trị mà con trỏ p2 đang trỏ đến (không in ra gì cả vì đang trỏ vào void)** |

**Màn hình sẽ xuất ra:**

**x=1**

**p1=5**

**p2=3**

**x=3**

**p1=3**

**p2=3**

**Bài 04: Chương trình không có lỗi**

|  |  |
| --- | --- |
| int \*p1; | tạo biến con trỏ p1 |
| int \*p2 = new int; | tạo biến con trỏ p2 và cấp phát vùng nhớ động cho nó |
| if (p1 == NULL) | nếu biến con trỏ p1 trỏ vào hư không (void) thì in ra **"P1 bang NULL"**  ngược lại, in ra **"P1 khac NULL"**  vì con trỏ p1 sau khi tạo sẽ mặc định trỏ vào rác (không phải NULL) nên câu lệnh else sẽ được thực thi |
| **printf(“P1 bang NULL”);** |
| else |
| **printf(“P1 khac NULL”);** |
| if (p2 == NULL) | nếu biến con trỏ p2 trỏ vào hư không (void) thì in ra **"P2 bang NULL"**  ngược lại, in ra **"P2 khac NULL"**  vì con trỏ p2 sau khi tạo sẽ mặc định trỏ vào rác (không phải NULL) nên câu lệnh else sẽ được thực thi |
| **printf(“P2 bang NULL”);** |
| else |
| **printf(“P2 khac NULL”);** |

**Màn hình sẽ xuất ra:**

**P1 khac NULLP2 khac NULL**

**Bài 06: Chương trình không có lỗi**

|  |  |
| --- | --- |
| int \*a; | tạo biến con trỏ a |
| int x; | tạo biến x |
| printf("x="); | cho người dùng nhập giá trị của biến x |
| scanf("%d", &x); |
| for (int i = 0; i < x; i++)  {  a[i] = i\*10;  } | cho vòng lặp với i chạy từ 0 đến trước x  với mỗi bước nhảy, nó sẽ lưu giá trị i\*10 vào vị trí i của mảng con trỏ a |
| **printf("a=%d\n", a);** | **in ra địa chỉ ban đầu của biến a (a[0])** |
| **printf("a=%d\n", \*a);** | **in ra giá trị ban đầu của biến a (a[0])** |
| **printf("a+1=%d\n", a);** | **in ra địa chỉ ban đầu của biến a (a[0])** |
| **printf("a+5=%d\n", \*(a + 5));** | **di chuyển con trỏ từ địa chỉ ban đầu của a (a[0]) sang phải 5 phần tử (a[5]) rồi in ra giá trị của vùng nhớ đó** |

**Màn hình sẽ xuất ra:**

**a=18028808**

**a=0**

**a+1=18028808**

**a+5=50**

**Bài 07: Chương trình có lỗi**

|  |  |
| --- | --- |
| int \*p; | tạo biến con trỏ p |
| **for (int i = 0; i < 10000; i++)**  **{**  **p = new int[100000];**  **}** | **cho vòng lặp với i chạy từ 0 đến trước 10000**  **với mỗi bước nhảy, nó sẽ cấp phát một vùng nhớ động gồm 100000 phần tử cho p** |

**Lỗi:**

**Vì trong vòng lặp, ta cấp phát vùng nhớ cho p mà không giải phóng nó. Do đó, sau mỗi bước nhảy, nó sẽ tạo ra những vùng nhớ mồ côi (vùng nhớ bị rò rỉ).**

**Bài 10: Chương trình có lỗi**

|  |  |
| --- | --- |
| int x = 10; | tạo biến x có giá trị là 10 |
| const int \*p1; | tạo con trỏ hằng p1 |
| p1 = &x; | cho con trỏ hằng p1 trỏ đến địa chỉ của biến x |
| **printf("%d\n", \*p1);** | **in ra giá trị mà con trỏ hằng p1 đang trỏ đến** |
| x = 100; | cho x bằng 100 |
| **printf("%d\n", \*p1);** | **in ra giá trị mà con trỏ hằng p1 đang trỏ đến** |
| **\*p1 = 1000;** | **cho giá trị mà con trỏ hằng p1 đang trỏ đến bằng 1000** |
| **printf("%d\n", \*p1);** | **in ra giá trị mà con trỏ hằng p1 đang trỏ đến** |

**Lỗi:**

**Vì p1 là con trỏ hằng nên vùng nhớ mà p1 đang trỏ đến là vùng nhớ hằng. Do đó sẽ có lỗi ở dòng 7 vì ta không được phép truy xuất đến vùng nhớ của con trỏ hằng thông qua chính con trỏ đó.**

**Bài 11: Chương trình có lỗi**

|  |  |
| --- | --- |
| int x = 10; | tạo biến x có giá trị là 10 |
| int y = 5; | tạo biến y có giá trị là 5 |
| int \*const p = &x; | tạo hằng con trỏ p trỏ vào địa chỉ của biến x |
| **printf("%d\n", \*p);** | **in ra giá trị mà hằng con trỏ p đang trỏ đến** |
| x = 1000; | cho x bằng 1000 |
| **printf("%d\n", \*p);** | **in ra giá trị mà hằng con trỏ p đang trỏ đến** |
| \*p = 10000; | cho giá trị mà hằng con trỏ p đang trỏ đến bằng 10000 |
| **printf("%d\n", \*p);** | **in ra giá trị mà hằng con trỏ p đang trỏ đến** |
| **p = 7;** | **cho hằng con trỏ p trỏ vào vùng nhớ có địa chỉ là 7** |
| **printf("%d\n", \*p);** | **in ra giá trị mà hằng con trỏ p đang trỏ đến** |
| **p = &y;** | **cho hằng con trỏ p trỏ vào địa chỉ của biến y** |
| **printf("%d\n", \*p);** | **in ra giá trị mà hằng con trỏ p đang trỏ đến** |

**Lỗi:**

**Vì p là hằng con trỏ nên nó sẽ trỏ vào một địa chỉ cố định. Do đó, ta có thể thay đổi giá trị mà con trỏ đang trỏ đến nhưng không thể thay đổi giá trị của chính con trỏ.**

**Bài 12: Chương trình không có lỗi**

|  |  |
| --- | --- |
| int \*p; | tạo biến con trỏ p |
| int n; | tạo biến n |
| scanf("%d", &n); | cho người dùng nhập giá trị của biến n |
| p = new int [n]; | cấp phát cho biến con trỏ p một vùng nhớ động gồm n phần tử |
| \*(p + 5) = 5; | di chuyển con trỏ từ địa chỉ ban đầu của p (p[0]) sang phải 5 phần tử (p[5]) rồi gán cho giá trị ở đó bằng 5 |
| **printf("%d\n", \*(p + 5));** | **di chuyển con trỏ từ địa chỉ ban đầu của p (p[0]) sang phải 5 phần tử (p[5]) rồi in ra giá trị của vùng nhớ đó** |
| f1(p, n); | gọi hàm f1, truyền biến con trỏ p và biến n vào hàm |
| void f1(int \*p, int n)  {  \*(p + 5) = 7;  } | di chuyển con trỏ từ địa chỉ ban đầu của p (p[0]) sang phải 5 phần tử (p[5]) rồi gán cho giá trị ở đó bằng 7 |
| **printf("%d\n", \*(p + 5));** | **di chuyển con trỏ từ địa chỉ ban đầu của p (p[0]) sang phải 5 phần tử (p[5]) rồi in ra giá trị của vùng nhớ đó** |

**Màn hình sẽ xuất ra:**

**5**

**7**

**Bài 13: Chương trình không có lỗi**

|  |  |
| --- | --- |
| int \*p; | tạo biến con trỏ p |
| int n; | tạo biến n |
| scanf("%d", &n); | cho người dùng nhập giá trị của biến n |
| p = new int [n]; | cấp phát cho biến con trỏ p một vùng nhớ động gồm n phần tử |
| \*(p + 5) = 5; | di chuyển con trỏ từ địa chỉ ban đầu của p (p[0]) sang phải 5 phần tử (p[5]) rồi gán cho giá trị ở đó bằng 5 |
| **printf("%d\n", \*(p + 5));** | **di chuyển con trỏ từ địa chỉ ban đầu của p (p[0]) sang phải 5 phần tử (p[5]) rồi in ra giá trị của vùng nhớ đó** |
| f2(p, n); | gọi hàm f1, truyền biến con trỏ p và biến n vào hàm |
| void f2(int \*p, int n)  {  p = new int [12];  \*(p + 5) = 7;  } | cấp phát cho biến con trỏ p một vùng nhớ động gồm 12 phần tử |
| di chuyển con trỏ từ địa chỉ ban đầu của p (p[0]) sang phải 5 phần tử (p[5]) rồi gán cho giá trị ở đó bằng 7  (vì ta đang thao tác với con trỏ vừa mới được cấp phát động, nên khi kết thúc hàm, con trỏ này sẽ biến mất mà không ảnh hưởng gì đến con trỏ p ban đầu) |
| **printf("%d\n", \*(p + 5));** | **di chuyển con trỏ từ địa chỉ ban đầu của p (p[0]) sang phải 5 phần tử (p[5]) rồi in ra giá trị của vùng nhớ đó** |

**Màn hình sẽ xuất ra:**

**5**

**5**