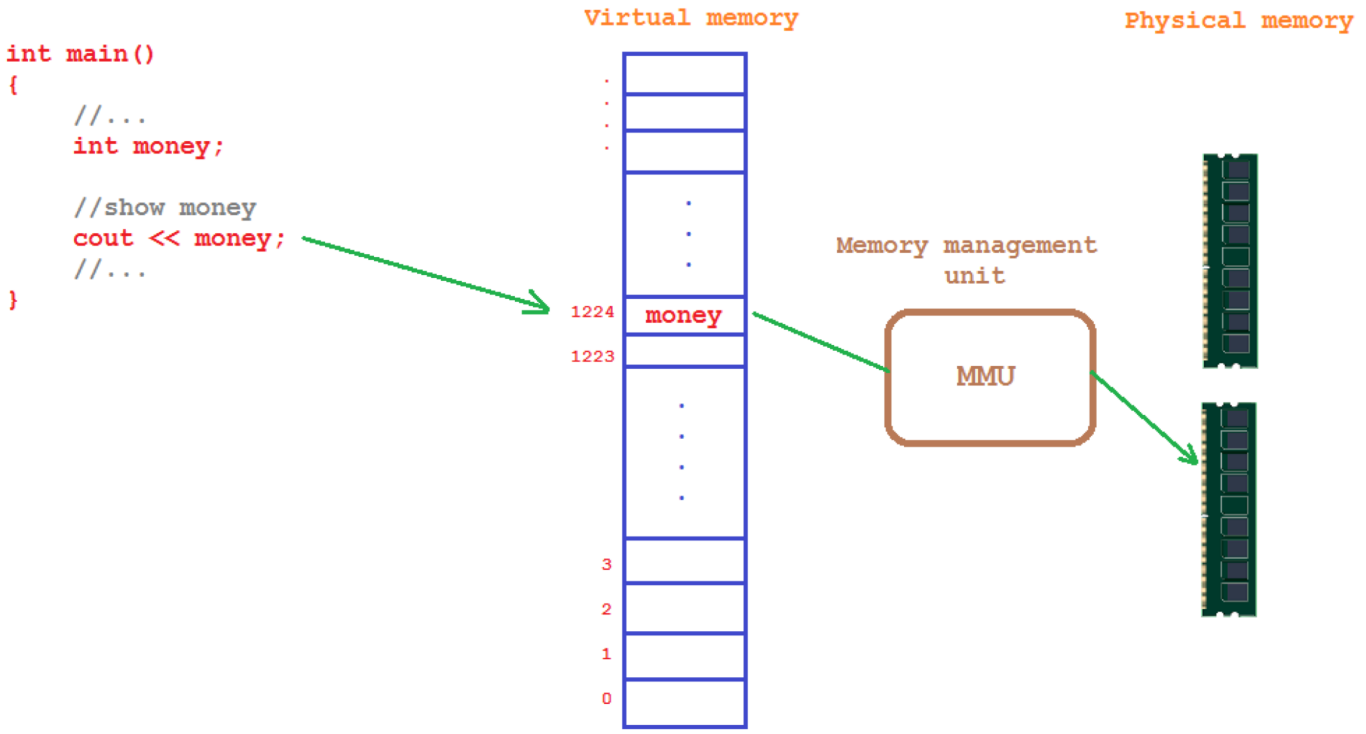
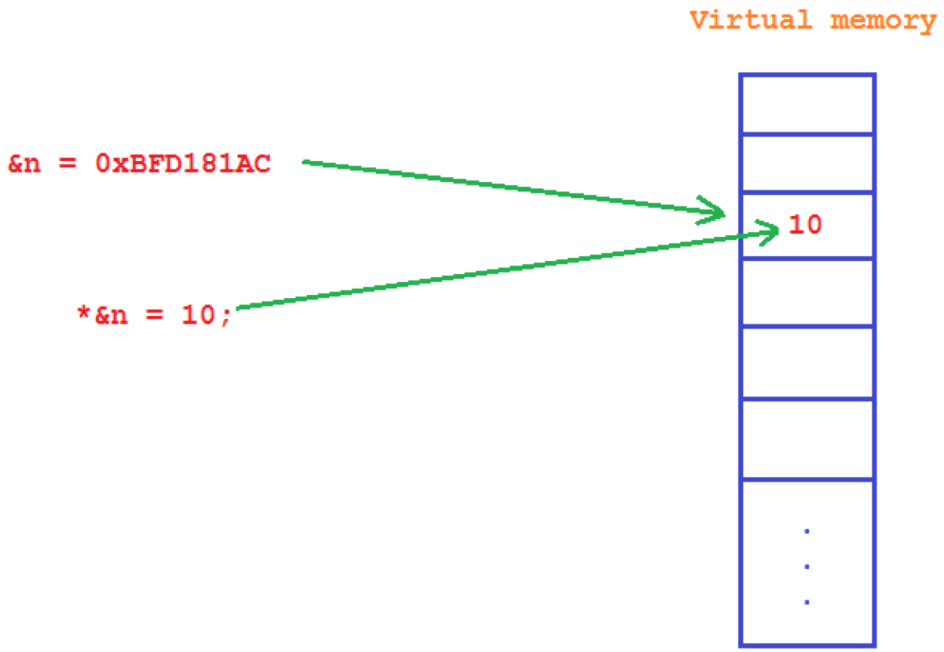
Địa chỉ của biến



Địa chỉ của biến



Địa chỉ của biến

```
int main() {
    int x = 6, y = 9;
    cout << &x << endl;
    cout << &y << endl;
    return 0;
}
//&x = 0x7ffeefbff598
//&y = 0x7ffeefbff594
```

Tham trị

```
int main() {
    int x = 6, y = 9;
    x = y;
    y = 10;
    cout<<x<<endl<<y<<endl;
    return 0;
}
//x = 9;
//y = 10;
```

Tham chiếu

Tham chiếu là ta tạo ra 1 biến mới nhưng sử dụng chung vùng nhớ với 1 biến cho trước. Khi khai báo tham chiếu ta phải cho biết cụ thể biến đó sẽ tham chiếu đến vùng nhớ của biến nào.

```
[KIỂU DỮ LIỆU VÙNG NHỚ THAM CHIẾU]& [TÊN BIẾN] = [BIẾN THAM CHIẾU];
```

```
int a;
int& b = a;

float c;
float&d = c;
```

Tham chiếu

```
int main() {
    int x = 6;
    int& y = x;
    cout<<x<<endl<<y<<endl;
    return 0;
}
//x = 6;
//y = 6;
```

Tham chiếu

```
int main() {
    float x = 6.0;
    int& y = x;
    cout<<x<<endl<<y<<endl;
    return 0;
}
//Lỗi
```

Tham chiếu

```
int main() {
    float x = 6.0;
    float& y;
    cout<<x<<endl<<y<<endl;
    return 0;
}
//Lỗi
```

Tham chiếu

```
int main() {
    int x = 6;
    int& y = x;
    y += x;
    cout<<x<<endl<<y<<endl;
    return 0;
}
//x = 12;
//y = 12;
```

Con trỏ

Con trỏ sinh ra để chứa địa chỉ của 1 vùng nhớ bất kỳ trong bộ nhớ. Tuy theo hệ điều hành mà ta có kích thước của con trỏ 4 bytes hoặc 8 bytes.

```
[KIỂU DỮ LIỆU VÙNG NHỚ]* [TÊN BIẾN CON TRỎ] = &[TÊN BIẾN];
```

```
int a;
int* b = &a;

float c;
float* d = &c;
```

Con trỏ

```
int main() {
    int x = 6;
    int* y = &x;
    cout<<x<<endl<<y<<endl;
    return 0;
}
//x = 6;
//y = 0x7ffeefbff598;
```

Con trỏ

Con trỏ cũng có vùng nhớ và địa chỉ.

```
int main() {
    int x = 6;
    int* y = &x;
    cout<<y<<endl<<&y<<endl;
    return 0;
}
//y = 0x7ffeefbff598;
//&y = 0x7ffeefbff590;
```

Con trỏ

```
int x = 6;
int* y = &x;
int** z = &y;
cout<<x<<endl;
cout<<y<<<<endl;
cout<<z<<<<endl;
//x = 6;
//y = 0x7ffeefbff598;
//z = 0x7ffeefbff608;
```

Con trỏ

Ta để dấu * phía trước con trỏ để lấy tham chiếu của vùng nhớ đang chứa địa chỉ.

```
int x = 6;
int* y = &x;
cout<<x<<endl;
cout<<y<<<<endl;
cout<<*y<<<<endl;
//x = 6;
//y = 0x7ffeefbff598;
//*y = 6;
```

Con trỏ

```
int x = 6;
int* y = &x;
int z = x * *y;
cout<<x<<endl;
cout<<y<<<<endl;
cout<<z<<<<endl;
//x = 6;
//y = 0x7ffeefbff598;
//z = 36;
```

Con trỏ

```
int x = 6;
int* y = &x;
int*& z = y;
cout<<x<<endl;
cout<<y<<<<endl;
cout<<z<<<<endl;
cout<<*z<<<<endl;
//x = 6;
//y = 0x7ffeefbff598;
//z = 0x7ffeefbff598;
//*z = 6;
```

Cấp phát tĩnh

Vùng nhớ được lưu trong stack. Và sẽ được tự động giải phóng sau khi kết thúc khối block hoặc chương trình.

Có kích thước bộ nhớ nhỏ và không quản lý được vòng đời của biến.

```
int x = 6;
int* y = &x;
int& z = x;
float a;
bool t = true;
```

Cấp phát động

- Vùng nhớ được lưu trong heap. Và không được giải phóng sau khi kết thúc chương trình. Ta phải sử dụng lệnh để giải phóng vùng nhớ.

- Kích thước bộ nhớ lớn và ta có thể quản lý được vòng đời của biến nên tiện cho việc sử dụng.

- Khi tạo vùng nhớ thì giá trị nhận được là 1 địa chỉ. Ta chỉ có thể gán giá trị thông qua tham chiếu.

- Sử dụng từ khoá new để tạo vùng nhớ và delete để giải phòng vùng nhớ.

Cấp phát động

```
int* y = new int;
*y = 10;

bool* z = new bool;
*z = true;
z = new bool;
*z = false;
```

Mảng và con trỏ

Mảng chính là con trỏ. Khi ta tạo ra 1 mảng thì biến mảng sẽ lưu địa chỉ của phần tử đầu tiên.

```
int a[10];
cout<<a;// &a[0];
cout<<a + 1;// &a[1];
cout<<a + 9;// &a[9];

*(a + 5) = 20;
*a = 10;
```

Mảng và con trỏ

```
int *a;
a =  new int[5];
…
cout<<a[0];
cout<<*a;
cout<<a[1];
cout<<*(a + 1);
```