Mảng động

Ở bài học trước chúng ta đã làm quan với mảng tĩnh. Trong thực tế mảng tĩnh có rất nhiều ưu điểm như truy cập và thao tác nhanh, do các phần tử của mảng tĩnh được lưu trong một khối bộ nhớ bao gồm các ô nhớ liên tiếp. Tuy nhiên mảng tĩnh cũng có một số nhược điểm như sau:

- Phải chỉ định kích thước khi khai báo mảng, nhưng trong thực tế ta có thể không biết được kích thước trước khi chạy chương trình.
- Phải ước lượng kích thước tối đa có thể cần đến, đôi khi do ta ước lượng sai gây ra lỗi chương trình hoặc gây lãng phí bộ nhớ do không dùng hết.

Do những nhược điểm trên của mảng tĩnh, mảng động được ra đời để khắc phục các nhược điểm đó. Do vậy ưu điểm của mảng động là:

- Kích thước của mảng động có thể thay đổi được, có thể co hay dãn khi cần.

Cách tạo mảng động

Để tạo mảng động ta dùng toán tử new (trong C++).

- Cấp phát động cho biến con trỏ.
- Sau đó có thể dùng nó như một mảng tĩnh thông thường.

Ví dụ:

```
typedef int * IntPtr;

IntPtr pi;

pi = new int[8]; //Tạo biến mảng cấp phát động pi có 8 phần tử, kiểu cơ sở là int
```

Xóa mảng động

Mảng động được cấp phát trong khi chạy chương trình, do vậy nó nên được hủy sau khi chương trình đã chạy xong.

Để xóa mảng động ta dùng cú pháp như sau:

```
typedef int * IntPtr;

IntPtr pi;

pi = new int[8];

… // Một vài xử lý ở đây

delete [] pi; // Xóa mảng động pi
```

Với thao tác này ta đã:

- Giải phóng tất cả vùng nhớ mà mảng động pi đang chiếm giữ.
- Sau khi xóa mảng động pi vẫn trỏ tới vùng nhớ đó. Để cho an toàn, sau khi xóa mảng động ta cần gán pi = NULL; (pi không trỏ tới đâu cả).

Hàm trả về một mảng

Không được phép trả về kiểu mảng trong hàm.

Ví dụ:

```
int [] function(); // Khai báo này không hợp lệ
```

Có thể thay băng việc trả về con trỏ có cùng kiểu dữ liệu cơ sở.

Ví dụ:

```
int * function(); // Khai báo này là hợp lệ
```

Tông kết:

Tùy vào từng trường hợp mà ta quyết định nên dùng mảng tĩnh hay mảng động.

Ưu điểm của mảng động:

Có thể dùng biến để làm tham số cho kích thước của mảng, tức là có thể dùng biến bên trong cặp ngoặc vuông []. Như vậy ta có thể tính toán được kích thước của mảng động trước khi cấp phát. tránh tình trạng lãng phí bị nhớ.

Mảng động được lưu trong vùng nhớ Heap là vùng nhớ thay đổi được, nên chúng ta không phải lo về tình trạng bị thiếu bộ nhớ.

Nhược điểm của mảng động:

Khai báo phức tạp hơn cấp phát tĩnh.

Phải kiểm soát được vùng nhớ động mà bạn đang sử dụng, vì trình biên dịch không tự động thu hồi vùng nhớ bạn đã cấp phát. Nên bạn phải dùng toán tử delete đối với biến đơn, delete [] đối với mảng để giải phóng bộ nhớ.

Các bài ví dụ liên quan tới mảng dộng:

- Thao tác với mảng động seqList
-