

****ListMembers**

Con trỏ cấp 2 trỏ tới vùng
nhớ chứa các con trỏ trỏ tới
giá trị



***Member**

Các con trỏ tới giá trị



Kiến trúc máy tính gồm 2
thành phần:
-Địa chỉ
-Dữ liệu



Các giá trị khởi tạo
InfoMember



Con trỏ trỏ tới NULL để thể
hiện vùng nhớ không xác
định (không tồn tại giá trị)

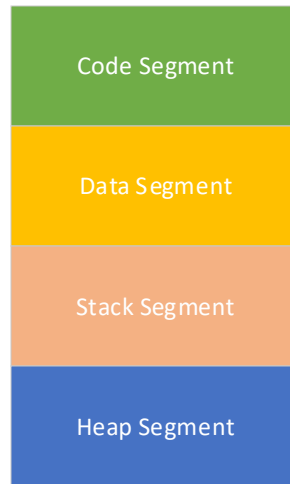
Thực chất nó là 1 mảng phần
tử tuy nhiên nó không liên
tục mà phân tán: Các con trỏ
tập trung thành 1 mảng, các
giá trị ở các vị trí khác nhau.

Khác với mảng A[] là các giá
trị xếp liền nhau, địa trị phần
tử đầu tiên là địa chỉ của
mảng

Các bước làm:

1. Khởi tạo con trỏ cấp 2 trỏ
đến vùng n con trỏ.
2. gán giá trị con trỏ trong
mảng là NULL.
3. Thêm phần tử ta chỉ việc
đến con trỏ trỏ tới vùng
NULL, khởi tạo giá trị rồi cho
con trỏ trỏ tới giá trị đó

Tổ chức bộ nhớ trong chương trình. Có thể coi đây là bộ nhớ chương trình khi chạy file exe



Mã nguồn của chúng ta sau khi hoàn tất quá trình Build sẽ chuyển thành các đoạn mã máy 0 và 1. Khi chương trình khởi chạy, các đoạn mã máy này sẽ được nạp vào Code Segment. Cách duy nhất để truy xuất vào vùng nhớ này là sử dụng Con trỏ hàm.

Code Segment có kích thước cố định.

Khi chương trình được khởi chạy, toàn bộ các biến toàn cục và static đều được lưu trữ ở đây, các đoạn chuỗi cố định cũng được lưu trữ trong Data Segment.

Data Segment có kích thước cố định.

Đây là vùng nhớ mà chúng ta cần quan tâm. Khi một hàm được gọi, hàm đó sẽ được đưa vào vùng nhớ Stack, các biến được khai báo trong hàm đó cũng được đưa vào vùng nhớ Stack. Khi hàm kết thúc, toàn bộ các biến trong hàm cùng với bản thân hàm sẽ được tự động giải phóng để các hàm sau sử dụng.

Stack Segment có kích thước cố định. Khi chúng ta khai báo quá nhiều biến hoặc một mảng có số lượng phần tử quá lớn, một hàm đệ quy vô hạn... bộ nhớ Stack sẽ bị đầy, dẫn đến chương trình bị dừng. Chúng ta cần lưu ý chuyện này.

Đây là vùng nhớ khác mà chúng ta cần quan tâm, vùng nhớ này chúng ta phải hoàn toàn kiểm soát nó. Khi chúng ta sử dụng con trỏ và cấp phát động một vùng nhớ cho con trỏ quản lý, vùng nhớ này sẽ nằm trong Heap Segment. Những vùng nhớ được cấp phát động sẽ không tự động thu hồi khi khối lệnh kết thúc, lập trình viên phải chủ động thu hồi chúng khi không còn nhu cầu sử dụng.

Heap Segment có kích thước không cố định, nên nó còn được gọi là vùng nhớ động, kích thước của nó có thể tăng hoặc giảm tùy vào sự cấp phát động. Lưu ý là Heap có thể mở rộng cho đến khi RAM đầy, nên chúng ta cần kiểm soát thật tốt, thu hồi các vùng nhớ được cấp phát động ngay khi không còn nhu cầu sử dụng để tránh việc lãng phí bộ nhớ.

1. Các biến mình khởi tạo (`int a`, `float B[10]` 10 phần tử) sẽ nằm ở bộ nhớ data. Các biến này sẽ khởi tạo ngay khi chạy chương trình. Điều này giúp truy cập nhanh hơn không tốn nhiều thời gian khởi tạo. Tuy nhiên vì vùng nhớ này cố định sẽ dẫn tới đầy bộ nhớ.
2. Con trỏ sẽ được lưu ở Data tuy nhiên dữ liệu con trỏ trỏ tới lại ở Heap thế nên ta có thể mở rộng chương trình nhiều hơn bằng cách cấp phát bộ nhớ. Tuy nhiên ta cần quản lý tốt: VD 2 con trỏ trỏ tới vùng địa chỉ thì thay đổi 1 cái sẽ thay đổi cả 2, máy tính cũng có giới hạn thế nên ta không thể cứ cấp phát mà không giải phóng -> tràn bộ nhớ, nếu con trỏ trỏ tới vùng nhớ khác thì vùng nhớ cũ vẫn còn tồn tại nếu không giải phóng dẫn tới tràn bộ nhớ,.... Bla bla
3. Yêu cầu khi sử dụng con trỏ cần có chiến lược hợp lý, quản lý chính xác, hạn chế thay đổi vùng nhớ, free bộ nhớ khi cần, không gán con trỏ này bằng con trỏ khác nhiều,...

Thêm chức năng tìm kiếm

1. Khi người dùng tìm kiếm một từ khóa. Chương trình trả lại danh sách member có liên quan.
2. Yêu cầu: Từ khóa người dùng nhập vào là 1 chuỗi ký tự, hàm sẽ trả về 1 danh sách các member có liên quan (phải kiểm tra tất cả name %s, id %d)
3. Tên hàm: int Search(InfoMember **members, InfoMember **searchMembers, char *key)
Trả về n số phần tử tìm được
Trả về -x với mã lỗi x
4. Hướng dẫn:
 - Khởi tạo con trỏ cấp hai lưu trữ kết quả tìm kiếm
 - Duyệt từng phần tử:
 - Chuyển id, tuổi,... %d thành dạng chuỗi ký tự, sau đó dùng hàm tìm kiếm (strstr) xem tồn tại key trong đó hay không. Nếu có gán con trỏ members[index] vào mảng kết quả

Sửa một số hàm đã làm

1. các hàm con trỏ nếu lỗi xảy ra trả về giá trị NULL
2. các hàm hành động void sửa thành kiểu trả về int với:
Trả về 1 nếu thành công
Trả về -x với mã lỗi x

Chức năng thêm member từ file

1. có 2 chế độ hoặc 2 hàm thêm từ file, làm mới từ file (xóa tất cả dữ liệu rồi thêm từ file)
2. Yêu cầu: người dùng nhập tên file chương trình đọc file đó kiểm tra xem file tồn tại hay không, dữ liệu hợp lệ hay không, ... nếu không lỗi thêm từ file vào mảng
3. Tên hàm: int AddFromFile(InfoMember **members, char *file)
Trả về 1 nếu thành công
Trả về -x với mã lỗi x

Chức năng In ra file tương tự tuy nhiên cần thống nhất cách in ra và cách thêm vào file (file in ra cũng có thể thêm vào) hạn chế xử lý chuỗi.

Thống nhất file có dạng:

```
Id1
Name1
Age1
Group_id1
Id2
Name2
Age2
Group_id2
....
```

Đọc đến khi ký tự kết thúc feof(file)