Nội dung

* Cấp phát bộ nhớ động
* File nhị phân

Cấp phát bộ nhớ động

* Bộ nhớ cấp phát cho biến lấy từ HEAP
* Địa chỉ ô nhớ cấp phát được xác định tại thời điểm chạy
* Có thể cấp phát cho các biến kích thước lớn và rất lớn ( cấp phát tĩnh chỉ dùng cấp phát cho các biến đơn giản)
* HDH KHÔNG tự giải phóng vùng nhớ cấp phát động riêng lẻ --> ai sẽ giải phóng
  + Người lập trình C/C++
  + Framework Java/Python - Garbage Collector
* Nếu không giải phóng bộ nhớ cấp phát động thì sao?
  + Nếu chương trình chạy liên tục trong thời gian dài, nếu cứ cấp phát bộ nhớ động mà không giải phóng --> hết bộ nhớ để cấp phát
  + Chương trình tắt (thoát) --> HDH sẽ tự giải phóng toàn bộ bộ nhớ cấp phát (cả động và tĩnh).
* Thói quen lập tình tốt? Luôn giải phóng bộ nhớ cấp phát động mỗi khi không cần dùng nữa

Hàm với C: malloc/calloc/realloc, free

Với C++: new, delete

Java : new (việc giải phóng do framework đảm nhiệm)

Các kiểu dữ liệu trong C

* Kiểu sơ cấp: int, char, float, double
* Kiểu dữ liệu thứ cấp: array, string (kiểu array of chars), struct, pointer, enum,...

Bài tập 1. Viết chương trình quản lý việc tiêm chủng của sinh viên trong lớp

Thông tin sinh viên gồm

* SHSV: char 15
* Họ tên: char 100
* SDT: char 15
* Char: số lần tiêm vaccine (0- chưa tiêm chủng, 1- đã tiêm 1 mũi,...)
* Ngày tiêm gần nhất (tiêm mũi cuối cùng): char 8 (YYYYMMDD), nếu chưa tiêm --> 00000000

Danh sách sinh viên này được lưu bằng file nhị phân với định dạng

Đầu tiên là số lượng phần tử trong file

Tiếp đến là thông tin của từng sinh viên trong file

Viết chương trình

* Tạo ra file nhị phân với thông tin sinh viên tiêm chủng được nhập từ bàn phím
* Bổ sung thêm sinh viên mới hoặc bổ sung thông tin (check trùng theo SHSV)
  + Nếu bị trùng --> hỏi người dùng có cập nhật lại số mũi tiêm và ngày tiêm gần nhất hay ko
* Ghi ra file
* Tra cứu thông tin tiêm chủng của sinh viên theo SDT hoặc mã sinh viên

Cần làm

// tim va in ra thong tin sinh vien theo SHSV neu co

void **traCuuThongTinTheoSHSV**(const SV \*list, int size, const char \*shsv)

{

}

// ham bo sung them thong tin moi cho danh sach

// doc vao SHSV, neu SHSV chua ton tai --> them phan tu moi (them vao cuoi)

// neu da ton tai --> update so mui tiem va ngay tiem gan nhat

// mang da du, muon luwu them --> phai thay doi kich thuoc!

// realloc them 10 phan tu nua (thuc te nen la nhan gap 2 so luong phan tu hien co)

// realloc(void \*ptr, size\_t size)

// voi kich thuoc moi size =2\*size

void **capNhatDanhSach**(SV \*\*list, int \*size)

{

}

=====================================

**Bài tập về nhà**

Bài 1. Bổ sung thêm các hàm cho phần quản lý thông tin tiêm chủng của ví dụ tuần 3

// tim va in ra thong tin sinh vien theo SHSV neu co

void **traCuuThongTinTheoSHSV**(const SV \*list, int size, const char \*shsv)

{

}

// tim va in ra thong tin sinh vien theo SDT neu co

void **traCuuThongTinTheoSDT**(const SV \*list, int size, const char \*sdt)

{

}

// ham bo sung them thong tin moi cho danh sach

// doc vao SHSV, neu SHSV chua ton tai --> them phan tu moi (them vao cuoi)

// neu da ton tai --> update so mui tiem va ngay tiem gan nhat

// mang da du, muon luu them --> phai thay doi kich thuoc!

// realloc them 10 phan tu nua (thuc te nen la nhan gap 2 so luong phan tu hien co)

// realloc(void \*ptr, size\_t size)

// voi kich thuoc moi size =2\*size

void **capNhatDanhSach**(SV \*\*list, int \*size)

{

}

// sap xep danh sach sinh vien hien co theo thu tu tang dan SHSV

// dieu chinh noi dung mang list, chi can truyen vao \*list

void **sapSepDanhSachTheoSHSV**(SV \*list, int size)

{

}

==============================

Bài 2. Viết chương trình quản lý danh bạ điện thoại

Contact gồm có các thông tin

* SDT char 15
* Contactname char 100
* Int type: 0 -cá nhân, 1- công ty,…

Hãy viết chương trình

* Đọc vào danh bạ điện thoại từ bàn phím (\*)
* Ghi danh bạ ra file nhị phân (\*)
* Đọc danh bạ từ file nhị phân (\*)
* Tra cứu thông tin của SDT (trả về tên người và loại contact) (\*)
* Bổ sung thêm SDT mới (check xem SDT đó đã có hay chưa, chỉ bổ sung khi SDT chưa có)
* Xóa 1 SDT trong danh bạ
* Tách file danh bạ điện thoại thành 2 file nhỏ hơn
* Gộp danh bạ điện thoại (từ 2 file nhỏ, trộn lại), bỏ qua SDT trùng