Nhắc lại một số kiến thức cơ bản

- Kiểu cấu trúc
- Kiểu con trỏ
- Con trỏ cấu trúc
- Cấp phát bộ nhớ động
- Cấu trúc tự trỏ

Kiểu cấu trúc-struct

- Cấu trúc là kiểu dữ liệu gồm một nhóm các thành phần có *kiểu có thể không giống nhau*, mỗi thành phần được xác định bằng một *tên riêng* biệt.

Định nghĩa và khai báo kiểu struct

- **Định nghĩa biến cấu trúc:**

[typedef] **struct** tên_kiểu_cấu_trúc
{
    <tên_kiểu 1> <tên_trường_1>;
    <tên_kiểu 2> <tên_trường_2>;
    ….
    <tên_kiểu n> <tên_trường_n>;
} [các_biến_kiểu_cấu_trúc];

- **Khai báo biến cấu trúc:**

    struct tên_kiểu_cấu_trúc tên_biến_1, tên_biến_2;

Ví dụ

```
typedef  struct  SV
{
        int MSV;
        char hoten[25];
        float DTB;
};
```

Truy nhập đến các thành phần kiểu cấu trúc

- Cú pháp:

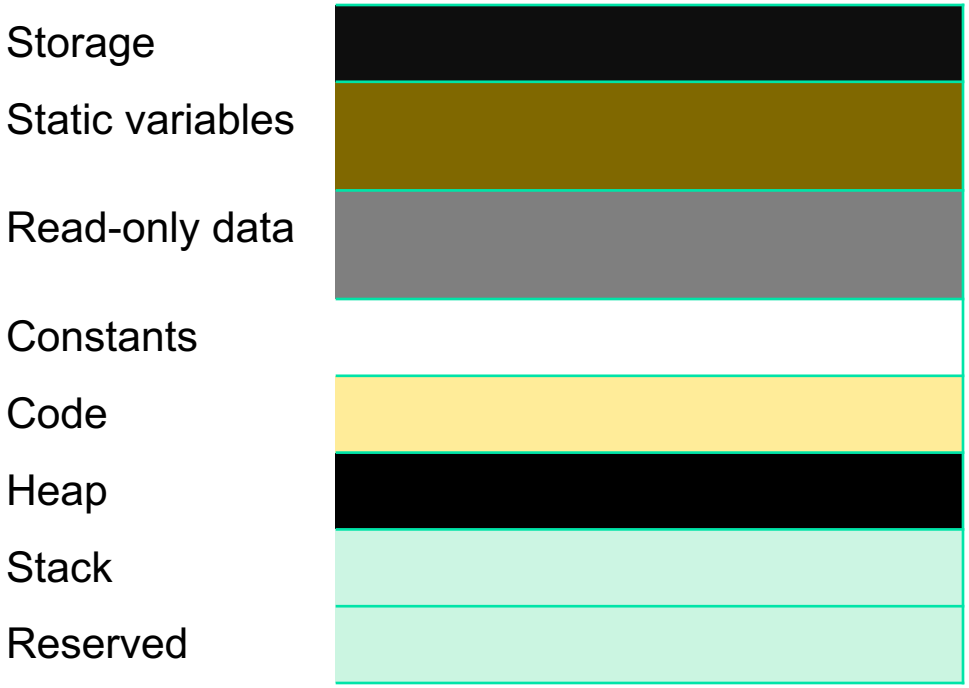
  tên_cấu_trúc.tên_thành_phần
- Ví dụ:

  SV x;

  x.MSV x.hoten x.DTB

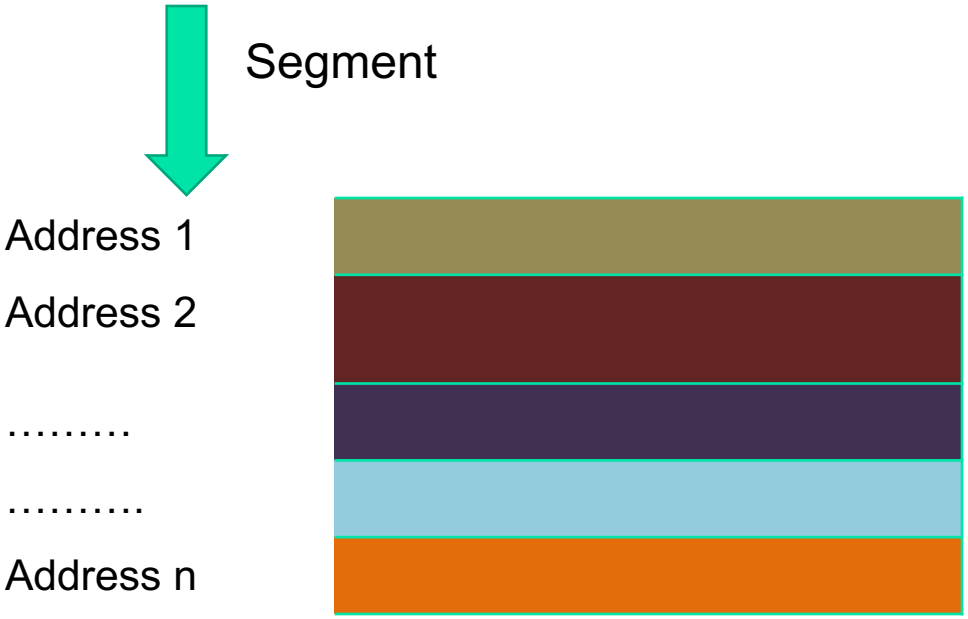
Tổ chức bộ nhớ trên máy tính

- Bộ nhớ của máy tính được tổ chức theo các phân đoạn, mỗi phân đoạn gọi là 1 segment
- Mỗi Segment thực hiện một nhiệm vụ chuyên biệt



Tổ chức bộ nhớ trên máy tính

- Để truy cập vào các vùng nhớ này máy tính sử dụng địa chỉ đoạn.
- Địa chỉ này gọi là địa chỉ phân đoạn (segment address)

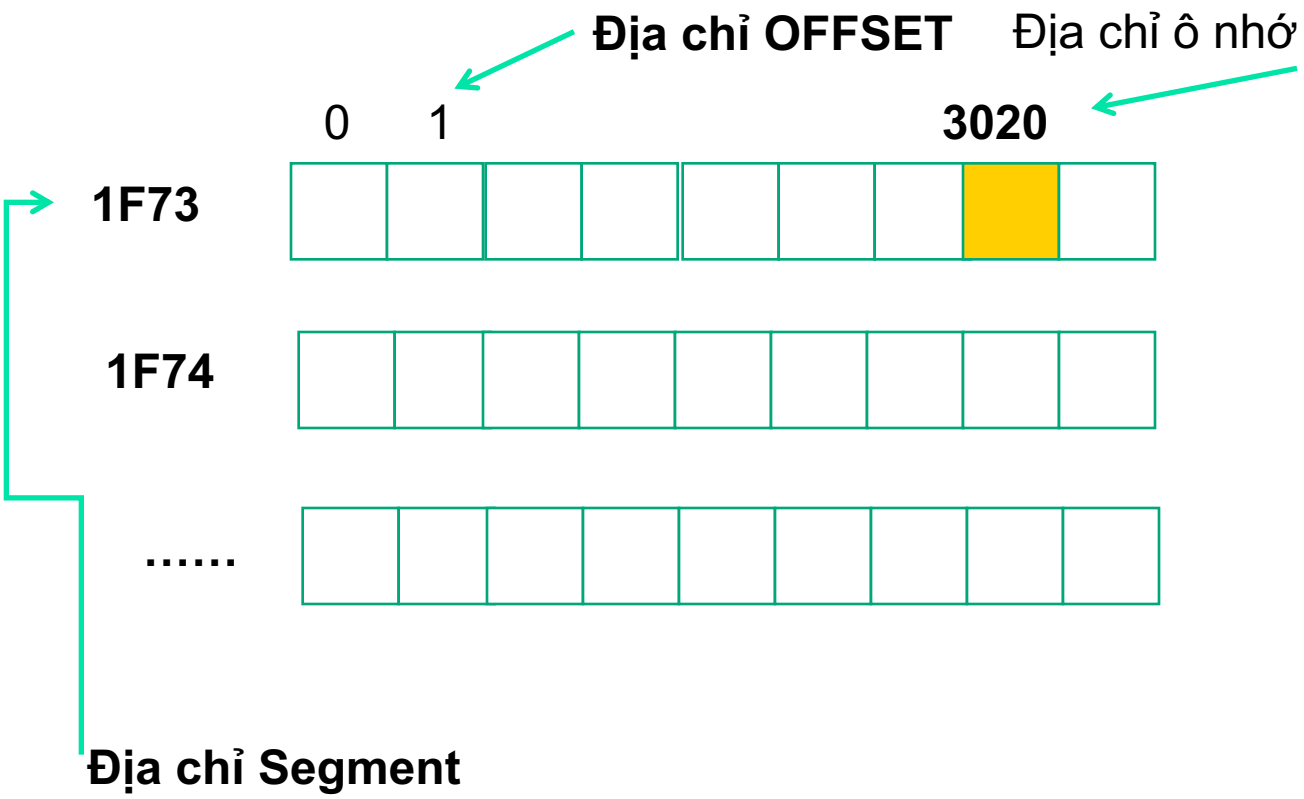


Tổ chức bộ nhớ trên máy tính

- Biến nhớ được tạo ra trong phân đoạn
- Một biến nhớ sẽ bao gồm 2 vùng địa chỉ
  - Địa chỉ đoạn gọi là địa chỉ Segment: xác định vị trí của đoạn nhớ
  - Địa chỉ nền hay còn gọi là địa chỉ Offset: xác định vị trí của ô nhớ trong đoạn nhớ

Con trỏ, địa chỉ và tham chiếu

- Ví dụ: 1 ô nhớ có địa chỉ 1F73:3020



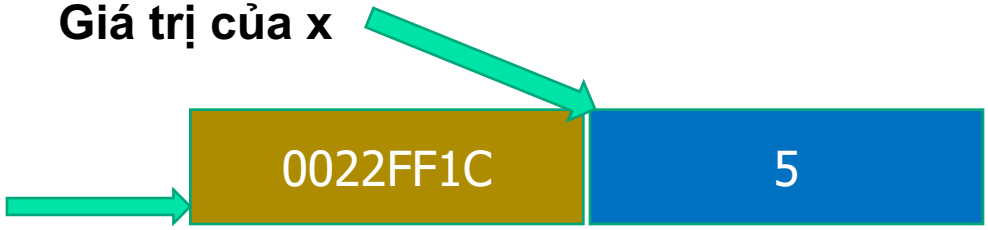
Con trỏ, địa chỉ và tham chiếu

- Mỗi biến nhớ trong chương trình gồm 2 thành phần
  - Giá trị: nội dung của biến nhớ đó chiếm giữ
  - Địa chỉ: một giá trị Hexa dùng để xác định vị trí của ô nhớ trong bộ nhớ

Con trỏ, địa chỉ và tham chiếu

- Ví dụ:

```
int x = 5;
```



Địa chỉ ô nhớ x

Trong đó:

0022: là địa chỉ Segment của x

FF1C: là địa chỉ Offset của x

Con trỏ và địa chỉ

- Để lưu trữ các giá trị trong chương trình ta dùng biến
- VD: `int x = 5; char c='a'; float y = 3.14…`
- Để truy cập vị trí của biến nhớ ta sử dụng 2 cách:
  - Tên của biến nhớ: x,y,c
  - Địa chỉ của biến nhớ.
- Vậy cần phải có một loại biến đặc biệt chỉ để lưu trữ địa chỉ ô nhớ
- Biến dùng để lưu trữ địa chỉ được gọi là con trỏ.

Kiểu con trỏ

- Con trỏ là kiểu dữ liệu dùng để lưu địa chỉ biến.
- Khai báo biến trỏ:

  Tên_kiểu *Tên_con_trỏ;

  - Ví dụ: int *p;
- Phép toán lấy địa chỉ ô nhớ: &
  - Ví dụ: int x=5; p= &x;
- Lưu ý:
  - Kích thước của biến con trỏ phụ thuộc vào kích thước của kiểu dữ liệu mà con trỏ trỏ tới
  - Khi khai báo con trỏ *p như trên thì:
    - p → giá trị địa chỉ mà con trỏ p trỏ tới
    - *p → giá trị nội dung mà con trỏ p trỏ tới

Con trỏ cấu trúc

- Con trỏ cấu trúc là con trỏ chứa địa chỉ của một biến cấu trúc
- Ví dụ

SV *p;

→ p là con trỏ cấu trúc.

- Truy nhập đến các thành phần của cấu trúc:

  Tên_con_trỏ ->Tên_trường

  Hoặc (*Tên_con_trỏ).Tên_trường

Ví dụ: p->MSV hoặc (*p).MSV

Mảng và con trỏ

- Mảng- Vùng nhớ tĩnh
  - Khi khai báo các biến dữ liệu → máy sẽ tạo vùng nhớ tĩnh cho các biến đó
  - Các vùng nhớ bị chiếm giữ trong suốt quá trình chạy chương trình
  - Có nhiều vùng nhớ không sử dụng đến do khai báo dư thừa
  - Có thể xảy ra thiếu bộ nhớ do không đủ số phần tử
- Con trỏ- Vùng nhớ động:
  - Khi có yêu cầu thì máy tính cấp phát vùng nhớ tương ứng
  - Sau khi sử dụng có thể giải phóng vùng nhớ cho chương trình

Cấp phát bộ nhớ động trong C:

- Hàm cấp phát bộ nhớ (*trong thư viện alloc.h*):
  void (datatype *)malloc(n);
  void (datatype*)calloc(int *num*, sizeof(object));
- Hàm giải phóng bộ nhớ:
  void free(void *ptr);
- Ví dụ:
  int *p;
  p = (int *)malloc(n); → cấp phát n blocks (kich thuoc moi block = kich thuoc cua kieu con tro (int)
  SV * p1;
  p1 = (SV*) calloc (1 , sizeof(SV)); → cấp fát 1 vùng nhớ có kích thước bằng kích thước của kiểu SV