BÀI 01 – CON TRỎ
KIẾN THỨC CON TRỎ CẦN NHỚ

1. TS. Nguyễn Tấn Trần Minh Khang
2. TS. Ngô Đức Thành
3. ThS. Võ Duy Nguyên
4. ThS. Nguyễn Hoàng Ngân

1. GHI NHỚ

Ghi nhớ

Miền giá trị
của một biến con trỏ là
địa chỉ ô nhớ.

2. CÚ PHÁP

2. Cú pháp

- Cú pháp:

```
KDL*<tên biến>;
```

- Ví dụ 01: `int *a;`

  Trong ví dụ trên ta nói **a** là một biến con trỏ kiểu số nguyên. Miền giá trị của biến **a** là địa chỉ ô nhớ.

- Ví dụ 02: `float *b;`

  Trong ví dụ trên ta nói **b** là một biến con trỏ kiểu số thực. Miền giá trị của biến **b** là địa chỉ ô nhớ.

2. Cú pháp

- Ví dụ 03:

```
1.struct phanso
2.{
3.    int tu;
4.    int mau;
5.};
6.typedef struct phanso PHANSO;
7.PHANSO *c;
```

- Trong ví dụ trên ta nói **c** là một biến con trỏ kiểu cấu trúc PHANSO. Miền giá trị của biến **c** là địa chỉ ô nhớ.

3. CÁCH DÙNG

3. Cách dùng

Có 2 cách sử dụng con trỏ.

- Cách 1: Sử dụng con trỏ để giữ địa chỉ của một biến.
- Cách 2: Sử dụng con trỏ để xin cấp phát và thu hồi bộ nhớ.

4. HAI TOÁN TỬ CƠ BẢN

4. Hai toán tử cơ bản

Hai toán tử cơ bản khi làm việc với con trỏ.

- Toán tử & (address-of operator): toán tử và (&) được sử dụng để lấy địa chỉ của một biến.
- Toán tử * (dereference operator): toán tử hoa thị (*) được sử dụng để lấy và cập nhật giá trị tại địa chỉ mà biến con trỏ đang giữ.

5. CẤP PHÁT – THU HỒI BỘ NHỚ

5. Cấp phát và thu hồi bộ nhớ

- Cấp phát: để cấp phát bộ nhớ cho một biến con trỏ ta có thể sử dụng các hàm: malloc, calloc, realloc, hoặc toán tử new.
- Thu hồi: để thu hồi bộ nhớ đã cấp phát cho một biến con trỏ ta dùng hàm free hoặc toán tử delete.
- Lưu ý: khi cấp phát bộ nhớ cho một biến con trỏ bằng toán tử new thì khi thu hồi bộ nhớ ta bắt buộc phải sử dụng toán tử delete.

6. CON TRỎ CẤU TRÚC

6. Con trỏ cấu trúc

- Để truy xuất đến một thành phần của biến con trỏ cấu trúc ta sử dụng toán tử mũi tên (->).

Chúc các bạn học tốt