Trường ĐH Khoa Học Tự Nhiên Tp. Hồ Chí Minh
TRUNG TÂM TIN HỌC

Lập trình iOS

Bài 2. *Các kiểu dữ liệu cơ sở trong Objective – C (Phần 2)*

Ngành Mạng và Thiết bị di động

2015

Nội dung

1. Hàm thủ tục
2. Cấu trúc điều khiển
3. Mảng cơ sở

1. Hàm thủ tục

- **Hàm thủ tục được định nghĩa như một tập hợp các câu lệnh thực hiện một hoặc nhiều chức năng, có thể có hoặc không có kết quả trả về.**
- **Cách khai báo bao gồm:**

<Kiểu dữ liệu trả về><Tên hàm>(Tham số 1, tham số 2…) {
    // Phần định nghĩa thân hàm.
}

Trong đó <Kiểu dữ liệu trả về> và <Tên hàm> bắt buộc phải có.

Ví dụ:

```
void demoFuntion(int thamSo1, double thamSo2) {
  NSLog(@"Giá trị tham số 1 %d", thamSo1);
  NSLog(@"Giá trị tham số 2 %f", thamSo2);
}
```

1. Hàm thủ tục

- **Tham số truyền vào: được hiểu như các biến dữ liệu cần thiết để thực thi một hàm, tất nhiên cũng có trường hợp hàm không có tham số truyền vào. Số lượng tham số truyền vào một hàm là không giới hạn, tuy nhiên chỉ nên truyền tham số khi cần thiết.**
  - Ví dụ hàm không có tham số truyền vào:

```
void noiXinChao() {
  NSLog(@"Xin chào!");
}
```

  - Ví dụ hàm có tham số truyền vào:

```
void noiXinChaoVoiTen(char *ten) {
  NSLog(@"Xin chào %s!" , ten);
}
```

1. Hàm thủ tục

- **Tham số trả về: dùng để kiểm tra mức độ hoàn thành của hàm đó.**
- **Tham số trả về là không nhất định và tùy thuộc vào ngữ cảnh và tính chất yêu cầu chức năng của mỗi hàm**
  - Ví dụ kiểm tra một số chẵn hay lẽ, ta có thể thực hiện như sau:

```
bool kiemTraSoChan(int n) {
  if (n % 2 == 0) {
    return true;
  }
  return false;
}
```

2. Cấu trúc điều khiển

- **Câu lệnh if-else: dùng để chia trường hợp ra để xử lý, nếu đúng thì làm gì, sai thì làm gì.**
  - Ví dụ:

```
int a = 12;
if(a > 10) //nếu a > 10 thì:
{
  NSLog(@"a = %d và a > 10 !" , a);
}else {
//Ngược lại thì:
  NSLog(@"a = %d và a <= 10 !" , a);
}
```

2. Cấu trúc điều khiển

- **Switch … case: tương tự như if, nhưng có thể dùng để xét nhiều trường hợp. Tuy nhiên chỉ sử dụng để so sánh với số**
  - Ví dụ:

```
int thang = 12;
switch (thang) {
  case 1:
    NSLog(@"Tháng giêng");
    break;
  case 2:
    NSLog(@"Tháng hai");
    break;
  default:
    break;
}
```

2. Cấu trúc điều khiển

- **Vòng lặp for: Dùng để thực thi các câu lệnh nhiều lần và biết trước số lần lặp.**
  - Ví dụ in ra từ tháng 1 đến tháng 12

```
for (int thang = 1; thang <= 12; thang ++) {
  NSLog(@"Tháng %d",thang);
}
```

- **Vòng lặp while: dùng để lặp giống như for, tuy nhiên được sử dụng khi không biết trước số vòng lặp nhưng biết được điều kiện lặp.**
  - Ví dụ:

```
int r = arc4random() % 10;
while (r!=5) {
  r = arc4random() % 10;
  NSLog(@"r = %d",r);
}
```

2. Cấu trúc điều khiển

- **Vòng lặp do… while: cũng giống như while nhưng thực hiện trước rồi mới xét điều kiện để lặp.**
  - Ví dụ:

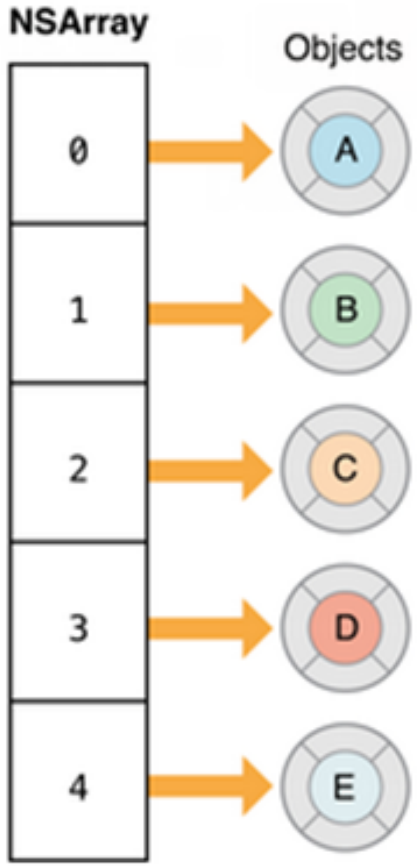
```
int r;
do{
   r = arc4random() % 10;
   NSLog(@"r = %d",r);
}while (r!=5);
```

Nội dung

1. **Hàm thủ tục**
2. **Cấu trúc điều khiển**
3. **Mảng cơ sở**
    1. Giới thiệu về NSArray và NSMutableArray
    2. Khởi tạo mảng
    3. Liệt kê phần tử mảng
    4. So sánh mảng
    5. Kiểm tra phần tử mảng
    6. Sắp xếp mảng

3.1 Giới thiệu về NSArray và NSMutableArray

- **NSArray cho phép quản lý các đối tượng**
  - Bất biến đổi (NSArray)
  - Biến đổi (NSMutableArray)
- **Các chức năng quản lý của dữ liệu tập hợp đối với đối tượng:**
  - Bất biến đổi :
    - Liệt kê dữ liệu đối tượng trong tập hợp
    - Định nghĩa đối tượng thuộc tập hợp
    - Truy xuất các phần tử trong tập hợp
  - Biến đổi:
    - Bao gồm các chức năng của dữ liệu tập hợp bất biến
    - Thêm đối tượng vào tập hợp
    - Xóa đối tượng khỏi tập hợp



3.1. Khởi tạo mảng

- **Khởi tạo mảng**
  - Cú pháp: **@[]**
  - Hoặc sử dụng phương thức **arrayWithObjects**
  - Ví dụ:

```
NSArray *array = @[@"t", @"3", @"h"];
NSArray *array2 = [[NSArray alloc] initWithObjects: @"t", @"3",
@"h",nil];
NSArray *array3 = [[NSArray alloc] initWithArray:array];
NSArray *tenPhim = [NSArray arrayWithObjects:@"Hobbit",
@"Avengers", @"Frozen", nil];
```

3.2. Liệt kê phần tử mảng

Liệt kê phần tử mảng

- Truy xuất từng phần tử theo chỉ số trong mảng hoặc có thể sử dụng các vòng lặp để truy xuất tuần tự các phần tử trong mảng
- Ví dụ:

```
NSArray *tenPhim = @[@"Hobbit", @"Avengers", @"Frozen"];
for (NSString *item in tenPhim) {
  NSLog(@"%@", item);
}
```

Hoặc

```
for (int i = 0; i < [tenPhim count]; i++) {
  NSLog(@"%d: %@", i, tenPhim[i]);
}
```

2.3 So sánh mảng

- **So sánh mảng**
  - Sử dụng phương thức **isEqualToArray** để thực hiện kiểm tra số lượng và giá trị phần tử mảng.
  - Kết quả trả về:
    - YES: hai mảng bằng nhau
    - NO: hai mảng không bằng nhau
  - Ví dụ:

```
NSArray *phimDangChieu = @[@"Hobbit", @"Avengers", @"Frozen"];
NSArray *phimSapChieu = @[@"47 Ronin", @"In the door"];
if ([phimDangChieu isEqualToArray:phimSapChieu]) {
  NSLog(@"Hai mảng bằng nhau");
}
```

2.4 Kiểm tra phần tử mảng

Kiểm tra phần tử mảng

- Sử dụng phương thức **containsObject** để thực hiện kiểm tra phần tử tồn tại trong mảng, phương thức **indexOfObject** cho biết vị trí đối tượng trong mảng.
- Ví dụ:

```
NSArray *phimDangChieu = @[@"Hobbit", @"Avengers", @"Frozen"];
if ([phimDangChieu containsObject:@"Hobbit"]) {
  NSLog(@"Hobbit – Đạo diễn: Peter Jackson");
}
NSUInteger index = [phimDangChieu indexOfObject:@"Hobbit"];
if (index == NSNotFound) {
  NSLog(@"Không tìm thấy");
} else {
  NSLog(@"Hobbit – Đạo diễn: Peter Jackson – Vị trí phim %ld",
index);
}
```

2.5 Sắp xếp mảng

Sắp xếp mảng

- Sử dụng phương thức **sortedArrayUsingComparator** kết hợp sử dụng kĩ thuật block **^NSComparisonResult(id obj1, id obj2)** để tiến hành sắp xếp mảng.
- Kết quả khi sắp xếp mặc định:
  - NSOrderedAscending : obj1 trước obj2
  - NSOrderedSame: obj1 và obj2 bằng nhau
  - NSOrderedDescending: obj1 sau obj2

2.5 Sắp xếp mảng

- Ví dụ:

```
NSArray *danhSachPhim = @[@"Hobbit", @"Avengers", @"Frozen"];
NSArray *dsSapXep = [danhSachPhim sortedArrayUsingComparator:
         ^NSComparisonResult(id obj1, id obj2) {
           if ([obj1 length] < [obj2 length]) {
             return NSOrderedAscending;
           } elseif ([obj1 length] > [obj2 length]) {
             return NSOrderedDescending;
           } else {
             return NSOrderedSame;
           }
         }];
NSLog(@"%@", dsSapXep );
```

Thảo luận