C语言高级特性教程

C语言的高级数据结构

1. 结构体的深入理解

1.1 结构体的概念

结构体（struct）是C语言中一种复合数据类型，允许你将不同类型的数据组合在一起，形成一个数据结构。这使得在程序中处理复杂数据类型时更加灵活和高效。

1.2 结构体的定义与使用

结构体的定义通常包括多个成员，每个成员可以是不同的数据类型。下面是一个结构体的定义示例，用于表示一个学生的信息：

```c
// 定义结构体
struct Student {
    char name[50];    // 学生姓名
    int age;          // 年龄
    float score;      // 成绩
};

// 使用结构体
int main() {
    struct Student stu1; // 声明结构体变量
    stu1.name = "张三";
    stu1.age = 20;
    stu1.score = 85.5;

    // 输出学生信息
    printf("学生姓名: %s\n", stu1.name);
    printf("学生年龄: %d\n", stu1.age);
    printf("学生成绩: %.1f\n", stu1.score);

    return 0;
}
```

1.3 结构体的高级操作

结构体可以嵌套使用，即一个结构体的成员可以是另一个结构体类型。此外，结构体还可以作为函数参数，返回值，以及数组的元素。

```c
// 嵌套结构体
struct Address {
    char street[100];
    char city[50];
};

struct Person {
    char name[50];
    struct Address addr;
};

// 结构体作为函数参数
void printPerson(struct Person p) {
    printf("姓名: %s\n", p.name);
    printf("地址: %s, %s\n", p.addr.street, p.addr.city);
}

int main() {
    struct Person p1;
    p1.name = "李四";
    p1.addr.street = "和平路123号";
    p1.addr.city = "北京";

    printPerson(p1); // 调用函数

    return 0;
}
```

2. 联合体的使用与陷阱

2.1 联合体的概念

联合体（union）是一种特殊的数据类型，它允许你将不同类型的数据存储在相同的内存位置。联合体的所有成员共享同一段内存，因此，同一时间只能存储一个成员的值。

2.2 联合体的定义与使用

下面是一个联合体的定义示例，用于表示一个可以存储整数或浮点数的变量：

```c
union Data {
    int i;
    float f;
};
```