数据结构01

概述

数据结构定义

如果把现实中大量复杂的问题以**特定的数据类型**和**特定的存储结构**保存到主存储器，在此基础上，为实现某个功能（如，查找某个元素，删除某个元素，对所有元素进行排序）而执行的相应操作，这个操作也叫算法

数据结构 = 个体 + 个体关系

算法 = 对存储数据的操作

算法：解题的方法和步骤

衡量算法的标准

1.时间复杂度：程序要执行的次数，而不是执行的时间

2.空间复杂度：执行过程中大概占用的最大内存

3.可读性

4.耐造性

预备知识

指针

定义：构建一个变量来储存其他变量的地址（内存单元的编号0–(4G–1)）

指针就是地址///指针变量是存放内存单元地址的变量

指针本质是一个操作受限（只能相减）的非负整数

e.g.

int *p // p 为指针变量名字，int * 表示p只能储存int类型变量的地址

int a = 5;

p = &a;

printf (a, &a, p, &p)； ————》》5，f4, f4, f7

指针与函数调用

```c
# include <stdio.h>

void f(int * p) //不是定义了一个名字叫做*p的形参，而是定义了一个形参，
{
    *p = 100;
}

int main(void)
{
    int i = 9;

    f(&i);
    printf("i = %d\n", i);

    return 0;
}
```

i = 100;

指针与数组

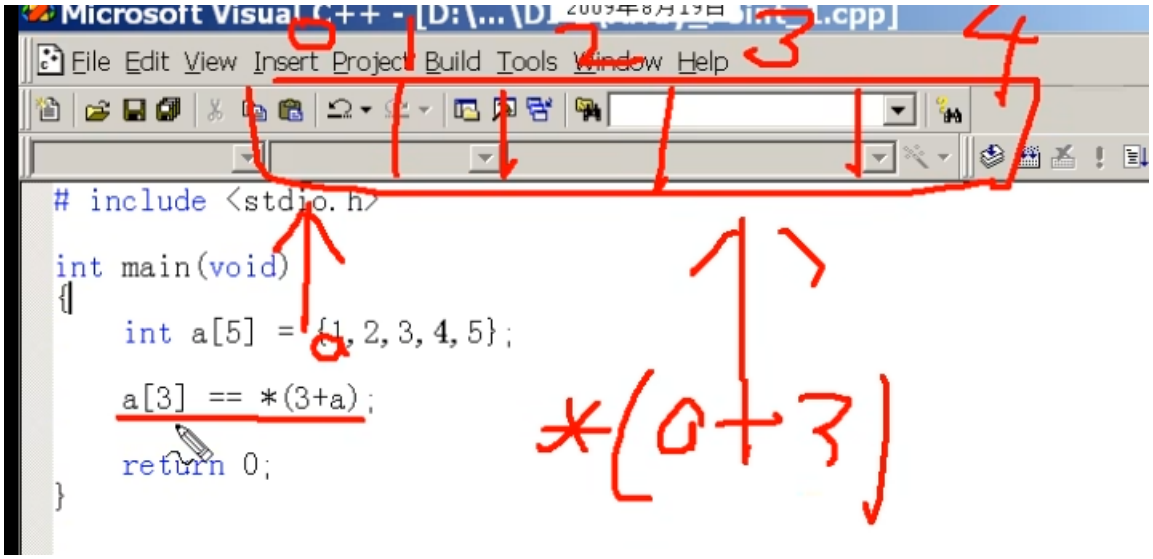
一堆数组名是个指针常量

which 存放的是数组中第一个元素的地址——》指向第一个元素

其值不能改变

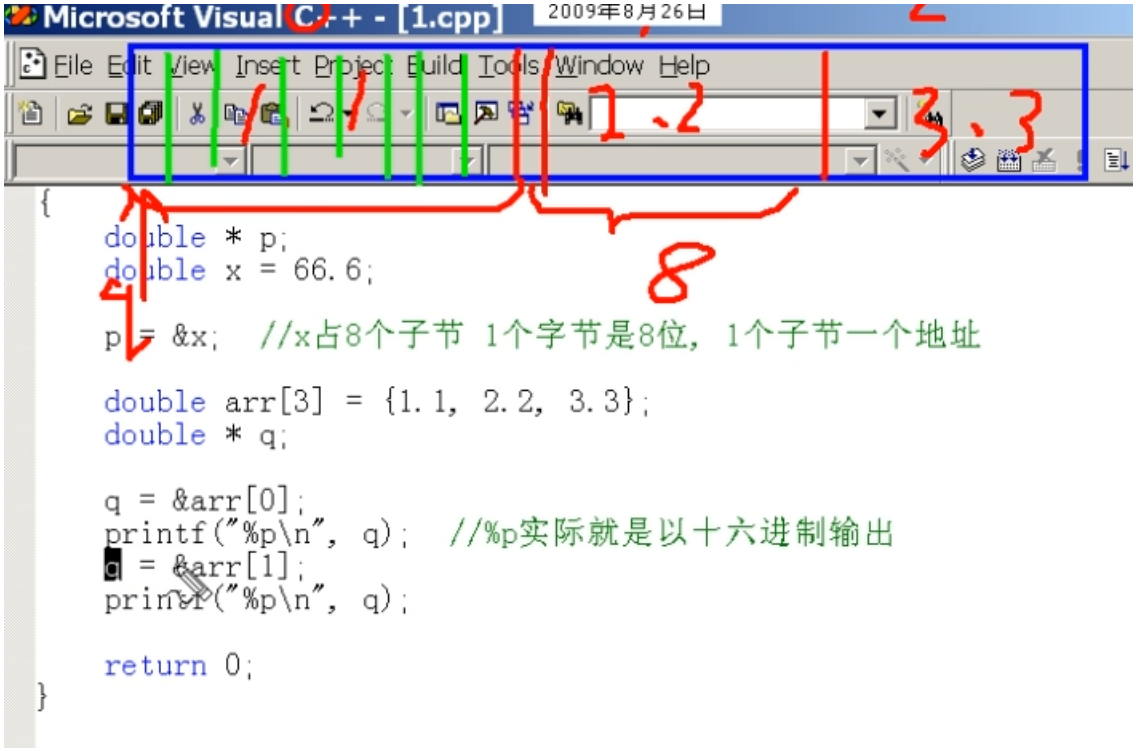
a 是数组里第一个数的地址

a[3] == *(a + 3)



指针的大小

数组指针只指向第一个元素的首部（如图）



故，指向200个字节和指向2个字节的指针大小相同

不管改什么变量，只要改所指地址就可以改变其值（**不懂**

```
int main(void)
{
    int i = 9;
    int * p = &i;// int  *p;  p = &i;

    printf("%p\n", p);
    f(&p);
    printf("%p\n", p);

    return 0;
}

void f(int ** q)
{
    *q = (int *)0xFFFFFFFF;
}
```

结构体

为什么要用结构体?

为了表达一些复杂的数据，而普通的基本变量类型无法满足要求

把一个事物的多个变量放到一起表达，避免重复（如，学生的学号、班级。姓名）

什么叫结构体?

用户根据自己需要定义的复合数据类型

和类相比，结构体没有方法

类

```
class Studnet
{
    int sid;
    String name;
    int sage;

    void inputStudent()
    {
    }

    void showStudent()
    {
    }
}
```

结构体

类型：struct Studnet

成员：int sid;

String name;

int age;

```
struct Studnet
{
    int sid;
    String name;
    int sage;
}
```

使用结构体创造新变量的两者写法

```
两种方式：
    struct Student st = {1000, "zhangsan", 20};
    struct Student * pst = &st;

    1.
        st.sid
    2.
        pst->sid
```

```c
int main(void)
{
    struct Student st = {1000, "zhangsan", 20};
    printf("%d  %s  %d\n", st.sid, st.name, st.age);

    st.sid = 99;
    //st.name = "lisi";  //error
    strcpy(st.name, "lisi");
    st.age = 22;
    printf("%d  %s  %d\n", st.sid, st.name, st.age);

    return 0;
}
```

注意事项

结构体变量不能加减乘除，但可以相互赋值

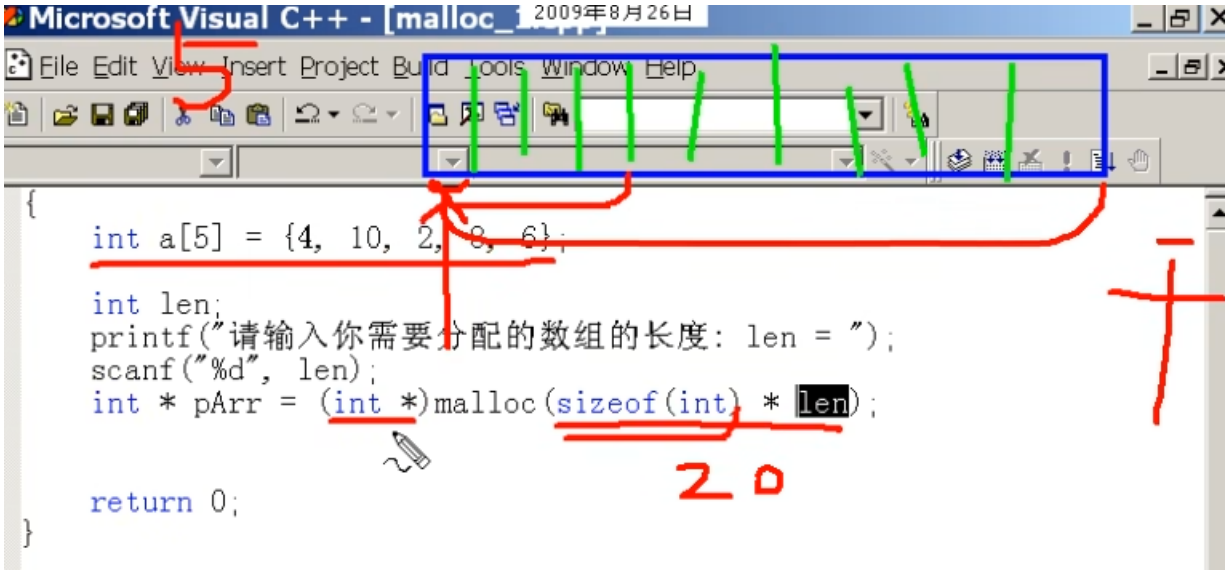
普通结构体变量和结构体指针变量作为函数传参的问题

动态内存的分配和释放

什么是静态，什么是动态的?

使用函数malloc（）——》动态

（int *）的含义：由于malloc只返回第一个字节的地址，我们不知道该数据是什么类型的（是占4个还是8个字节的数据），所以在前面强制转换为int，告诉编译器这是int数据的第一个字节的地址



```c
int main(void)
{
    int a[5] = {4, 10, 2, 8, 6};

    int len;
    printf("请输入你需要分配的数组的长度: len = ");
    scanf("%d", len);
    int * pArr = (int *)malloc(sizeof(int) * len);
    *pArr = 4;  //类似于 a[0] = 4;
    pArr[1] = 10; //类似于a[1] = 10;

    free(pArr);

    return 0;
}
```

静态动态数组的对比