



第8章 自定义数据类型





本章主要内容

- 8.1 结构体类型、变量定义与访问
- 8.2 结构体与函数
- 8.3 链表
- 8.4 共用体
- 8.5 枚举类型



8.1 结构体类型、变量定义与访问

- 为什么要定义结构体类型
- 定义结构体类型
- 定义结构体类型变量
- 访问结构体类型变量成员
- 结构体的应用



8.1.1为什么要定义结构体类型 从基本数据类型到抽象数据类型



问题域中包含的事物，如具体事物（汽车、员工、楼房、报账单等），抽象的概念、思想等都需要在程序中描述。



收益明细表	
编制单位: 牛角村	2009 年
单位: 元	
项 目	金额
本 年 收 益	
一、经营收入	20 万
加: 发包及上交收入	5 万
投资收益	
减: 经营支出	10
管理费用	5 万
二、经营收益	
加: 补助收入	2 万
其他收入	
减: 其他支出	2 万
三、本年收益	10 万



7.1.1为什么要定义结构体类型 从基本数据类型到抽象数据类型

单个属性的描述->基本数据类型

整型、浮点型、字符型、布尔等，不同语言会定义不同的基本类型。一个基本数据类型变量只能描述事物某个属性/特征，但是实际编程中，需要描述一个事物的很多属性/特征。

复杂事物（包含多个属性）的描述->?

分析要描述的属性，定义多个基本数据类型变量

大量复杂事物（包含多个属性的大量事物）的描述->?

学号	姓名	性别	出生年	数学	英语	计算机原理	程序设计
100310121	王 刚	男	1991	72	83	90	82
100310122	李小明	男	1992	88	92	78	78
100310123	王丽红	女	1991	98	72	89	66
100310124	陈莉莉	女	1992	87	95	78	90
...							



7.1.1为什么要定义结构体类型 从基本数据类型到抽象数据类型

如何用计算机程序实现下述表格的管理？

学号	姓名	性别	出生年	数学	英语	计算机原理	程序设计
100310121	王刚	男	1991	72	83	90	82
100310122	李小明	男	1992	88	92	78	78
100310123	王丽红	女	1991	98	72	89	66
100310124	陈莉莉	女	1992	87	95	78	90
...							

数组的解决方法

```
long   studentId[30];           /* 学号 */
char   studentName[30][10];     /* 姓名 */
char   studentSex[30];          /* 性别 */
int     yearOfBirth[30];         /* 出生年 */
int     scoreMath[30];           /* 数学课的成绩 */
int     scoreEnglish[30];        /* 英语课的成绩 */
int     scoreComputer[30];       /* 计算机原理课的成绩 */
int     scoreProgramming[30];    /* 程序设计课的成绩 */
```



8.1.1为什么要定义结构体类型 从基本数据类型到抽象数据类型

数组的解决方法（内存分配）

100310121	王刚	'M'	1991
100310122	李小明	'M'	1992
100310123	王丽红	'F'	1991
100310124	陈莉		
.....		

对数组赋初值时，易发生错位；
结构显得零散，不易管理；
逻辑上别扭。

72	83	78	78
88	92	89	66
98	72	78	90
87	95
.....		



8.1.1为什么要定义结构体类型 从基本数据类型到抽象数据类型

希望的内存分配图

100310121	100310122	100310123	100310124
王刚	李小明	王丽红	陈莉莉
'M'	'M'	'F'	'F'
1991	1992	1991	1992
72	88	98	87
83	92	72	95
90	78	89	78
82	78	66	90



8.1.1为什么要定义结构体类型 从基本数据类型到抽象数据类型

□ 用户自己构造数据类型-复合数据类型

- 由基本数据类型组合、迭代派生而来，表示复杂的数据对象
- 典型的代表就是“结构体”

□ 抽象数据类型（Abstract Data Type，简称ADT）

- 在复合数据类型基础上增加了对数据的操作
- C++中的类class就是一种抽象数据类型
- Class是Object-Oriented的一个重要概念



8.1.2 定义结构体类型

结构体类型描述复杂事物

- (1) 名称 **Student**
- (2) 包含哪些属性 **见表头**
- (3) 存储属性值的变量名及类型

学号: studentID long

姓名: studentName char[30]

性别: studentSex char

出生年: yearOfbirthday int

数学: math int

英语: english int

计算机原理: computer int

程序设计: program int

学号	姓名	性别	出生年	数学	英语	计算机原理	程序设计
100310121	王刚	男	1991	72	83	90	82
100310122	李小明	男	1992	88	92	78	78
100310123	王丽红	女	1991	98	72	89	66
100310124	陈莉莉	女	1992	87	95	78	90
...							



8.1.2 定义结构体类型

结构体类型描述的复杂事物

- (1) 名称 **Student**
- (2) 包含哪些属性 见表头
- (3) 存储属性值的变量名及类型

学号: studentID long

姓名: studentName char[30]

性别: studentSex char

出生年: yearOfBirth int

数学: math int

英语: english int

计算机原理: computer int

程序设计: program int

关键字

结构体名

```
struct Student{  
    long studentID;  
    char studentName[30];  
    char studentSex;  
    int yearOfBirth;  
    int math;  
    int english;  
    int computer;  
    int program;  
};
```

结构体成员

不要忘记分号!!



7.1.2 定义结构体类型

定义结构体类型后，形成一个类型的样板，用于生成结构体变量，但并未定义结构体变量，因而编译器不为其分配内存。

但是编译器会根据类型的定义，**确定该结构体变量所占用的内存空间，分配给结构体变量的内存空间至少要能够存储所有的成员。** `sizeof(struct Student)`

```
#include <stdio.h>
```

```
struct Student{  
    long studentID;  
    char studentName[30];  
    char studentSex;  
    int yearOfBirth;  
    int math;  
    int english;  
    int computer;  
    int program;  
};  
int main(){  
    return 0;  
}
```




8.1.2 定义结构体类型

struct Student是一个新的类型，每个定义的**struct Student**类型的变量，系统为其分配的空间都包含8个部分，如右图。

studentID
studentName
studentSex
yearOfBirth
math
english
computer
program

int
char[30]
char
int
int
int
int
int



8.1.3 定义结构体类型变量

定义结构体变量

- (1) 定义结构体类型的同时定义变量
定义的都是全局变量。

```
struct Student{  
    long studentID;  
    char studentName[30];  
    char studentSex;  
    int yearOfBirth;  
    int math;  
    int english;  
    int computer;  
    int program;  
} stu1, stu2;
```

- (2) 类型名 变量名;

```
struct Student stu1, stu2;  
//定义两个struct Student类型的变量
```



7.1.3 定义结构体类型变量

定义结构体变量

stu1内存存储

studentID		int
studentName		char[30]
studentSex		char
yearOfbirth		int
math		int
english		int
computer		int
program		int

stu2内存存储

studentID		int
studentName		char[30]
studentSex		char
yearOfbirth		int
math		int
english		int
computer		int
program		int



8.1.3 定义结构体类型变量

结构体变量赋值和访问

- (1) 定义时初始化
- (2) 赋值运算符对成员赋值
- (3) 通过赋值运算符整体赋值



8.1.3 定义结构体类型变量

结构体变量赋值和访问

(1) 定义时初始化

```
int main(){  
    struct Student stu1 = {100310421,"王刚",'M',1991,72,83,90,82};  
    struct Student stu2 = {100310223,"李芳",'F',2000,82,90,95,88};  
}
```

studentID	100310421
studentName	"王刚"
studentSex	'M'
yearOfbirth	1991
math	72
english	83
computer	90
program	82

studentID	100310223
studentName	"李芳"
studentSex	'F'
yearOfbirth	2000
math	82
english	90
computer	95
program	88



8.1.3 定义结构体类型变量

结构体变量赋值和访问

(2) 赋值运算符对成员赋值

结构体变量成员的访问包括两种方式：

- 通过结构体变量访问成员, 利用成员选择符 .
- 通过指向结构体变量的指针访问成员, 利用成员选择符 ->



8.1.3 定义结构体类型变量

结构体变量赋值和访问

(2) 赋值运算符对成员赋值

结构体变量成员的访问包括两种方式：

- 通过结构体变量访问成员, 利用成员选择符 .

结构体变量. 成员名 `stu1.studentID`

- 通过指向结构体变量的指针访问成员, 利用成员选择符 ->

结构体指针->成员名 `Student* ps = &stu1; ps->studentID;`



8.1.3 定义结构体类型变量

结构体变量赋值和访问

(2) 赋值运算符对成员赋值

结构体变量成员的访问：**通过结构体变量访问成员, 利用成员选择符 .**

结构体变量. 成员名

struct Student stu1;

stu1.studentID 是int变量

stu1.studentName 是char[30]数组

stu1.studentSex 是字符变量

stu1.studentID

stu1.studentName

stu1.studentSex

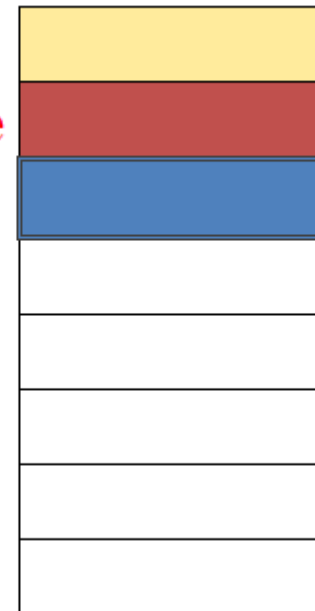
stu1.yearOfBirth

stu1.math

stu1.english

stu1.computer

stu1.program



//打印学生信息

```
#include <stdio.h>
struct Student{
    long studentID;
    char studentName[30];
    char studentSex;
    int yearOfbirth;
    int math;
    int english;
    int computer;
    int program;
};
int main(){
    struct Student stu1 = { 100310421,"王刚",'M',1991,72,83,90,82};
    printf("%s\n", stu1.studentName);
    printf("%d\n", stu1.studentID );
    printf("%c\n", stu1.studentSex );
    printf("%d\n", stu1.yearOfbirth );
    printf("%d\n", stu1.math );
    printf("%d\n", stu1.english );
    printf("%d\n", stu1.computer );
    printf("%d\n", stu1.program );
}
```

C:\Users\Administrator\Desktop\Untitled1.e

王刚
100310421
M
1991
72
83
90
82

//打印学生信息

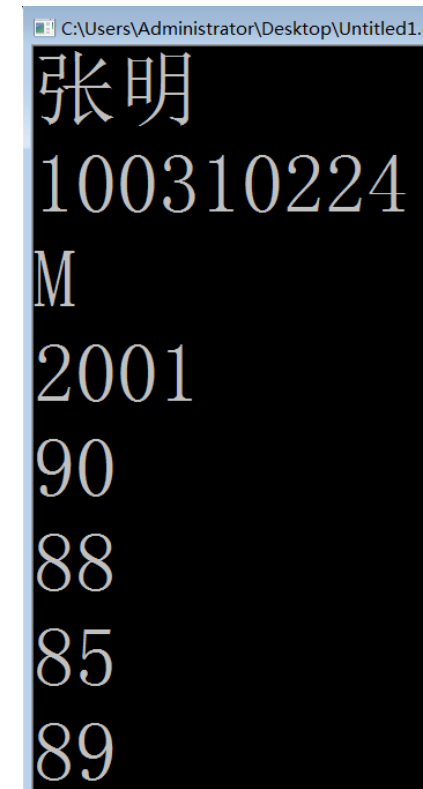
```
#include <stdio.h>
```

//struct Student定义见上例，此处为节约篇幅，省略.

```
int main(){
    struct Student stu1;
    strcpy(stu1.studentName, "张明");
    stu1.studentID = 100310224;
    stu1.studentSex = 'M';
    stu1.yearOfBirth = 2001;
    stu1.math = 90;
    stu1.english = 88;
    stu1.computer = 85;
    stu1.program = 89;
    printf("%s\n", stu1.studentName);
    printf("%d\n", stu1.studentID );
    printf("%c\n", stu1.studentSex );
    printf("%d\n", stu1.yearOfBirth );
    printf("%d\n", stu1.math );
    printf("%d\n", stu1.english );
    printf("%d\n", stu1.computer );
    printf("%d\n", stu1.program );
}
```

stu1内存存储

studentID	100310224
studentName	"张明"
studentSex	'M'
yearOfBirth	2001
math	90
english	88
computer	85
program	89



C:\Users\Administrator\Desktop\Untitled1.

张明
100310224
M
2001
90
88
85
89



8.1.3 定义结构体类型变量

结构体变量赋值和访问

(2) 赋值运算符对成员赋值

结构体变量成员的访问：通过结构体变量指针访问成员, 利用成员选择符 ->

结构体变量指针->成员名

```
struct Student stu1, *ps;
```

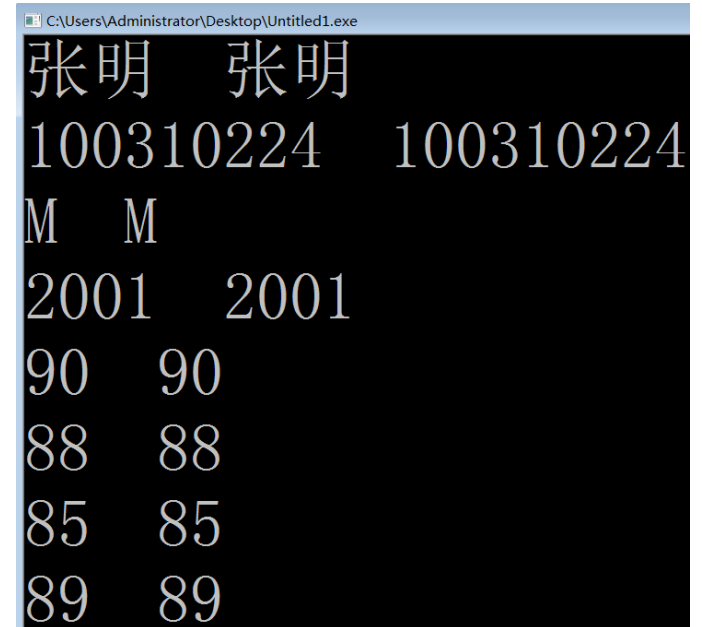
```
ps = &stu1;
```

ps->studentName 字符数组

ps->studentID int变量

//打印学生信息

```
#include <iostream>
using namespace std;
//struct Student定义见上例，此处为节约篇幅，省略.
int main(){
    struct Student stu1, *ps = &stu1;
    strcpy(ps->studentName, "张明");
    ps->studentID = 100310224;
    ps->studentSex = 'M';
    ps->yearOfbirth = 2001;
    ps->math = 90;
    ps->english = 88;
    ps->computer = 85;
    ps->program = 89;
    printf("%s %s", ps->studentName , stu1.studentName);
    printf("%d %d", ps->studentID , stu1.studentID );
    printf("%c %c", ps->studentSex , stu1.studentSex );
    printf("%d %d", ps->yearOfbirth , stu1.yearOfbirth );
    printf("%d %d", ps->math , stu1.math );
    printf("%d %d", ps->english , stu1.english );
    printf("%d %d", ps->computer , stu1.computer );
    printf("%d %d", ps->program , stu1.program );
}
```



```
张明 张明
100310224 100310224
M M
2001 2001
90 90
88 88
85 85
89 89
```

stu1内存存储

studentID	100310224
studentName	"张明"
studentSex	'M'
yearOfbirth	2001
math	90
english	88
computer	85
program	89

//分析程序

```
#include <iostream>
using namespace std;
//struct Student定义见上例，此处为节约篇幅，省略.
int main(){
    struct Student stu1, *ps = &stu1;
    strcpy(ps->studentName, "张明");
    (*ps).studentID = 100310224;
    (*ps).studentSex = 'M';
    stu1.yearOfBirth = 2001;
    stu1.math = 90;
    (&stu1)->english = 88;
    (&stu1)->computer = 85;
    (&stu1)->program = 89;
    printf("%s %s", ps->studentName , stu1.studentName);
    printf("%d %d", ps->studentID , stu1.studentID );
    printf("%c %c", ps->studentSex , stu1.studentSex );
    printf("%d %d", ps->yearOfBirth , stu1.yearOfBirth );
    printf("%d %d", ps->math , stu1.math );
    printf("%d %d", ps->english , stu1.english );
    printf("%d %d", ps->computer , stu1.computer );
    printf("%d %d", ps->program , stu1.program ); }
```



```
张明 张明
100310224 100310224
M M
2001 2001
90 90
88 88
85 85
89 89
```

stu1内存存储

studentID	100310224
studentName	"张明"
studentSex	'M'
yearOfBirth	2001
math	90
english	88
computer	85
program	89



8.1.3 定义结构体类型变量

结构体变量赋值和访问

(3) 通过赋值运算符整体赋值

```
struct Student stu1,stu2={ 100310224,“张明",'M',2001,90,88,85,89};
```

```
//...
```

```
stu1 = stu2;
```

将stu2的内容完整拷贝到stu1的存储区域

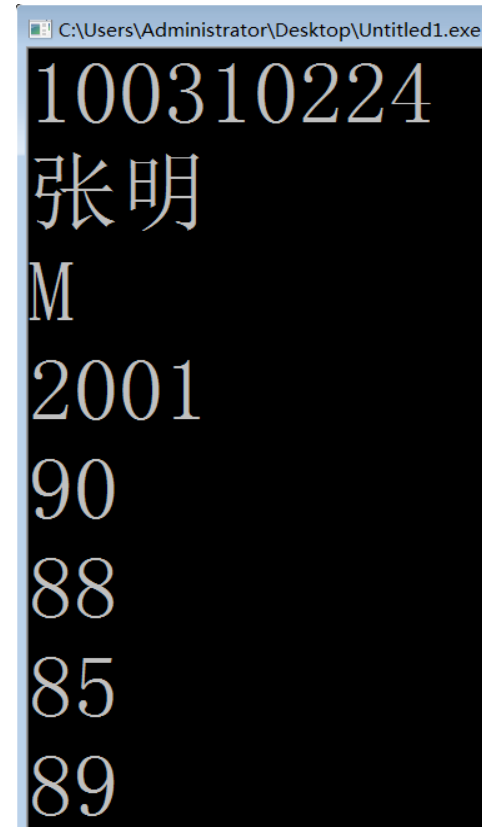
studentID		←	100310224
studentName		←	“张明”
studentSex		←	' M '
yearOfBirth		←	2001
math		←	90
english		←	88
computer		←	85
program		←	89

//分析程序

```
#include <stdio.h>
```

//struct Student定义见上例，此处为节约篇幅，省略。

```
int main(){  
    struct Student stu1, stu2 = {100310224,"张明",'M',2001,90,88,85,89};  
    stu1 = stu2;  
    printf("%d\n", stu1.studentID );  
    printf("%s\n", stu1.studentName );  
    printf("%c\n", stu1.studentSex );  
    printf("%d\n", stu1.yearOfBirth );  
    printf("%d\n", stu1.math );  
    printf("%d\n", stu1.english );  
    printf("%d\n", stu1.computer );  
    printf("%d\n", stu1.program );  
  
    return 0;  
}
```



```
C:\Users\Administrator\Desktop\Untitled1.exe  
100310224  
张明  
M  
2001  
90  
88  
85  
89
```



8.1.3 结构体嵌套

一个结构体类型变量可以作为另一个结构体类型的成员。

为某汽车设计制造企业开发一套系统，进行汽车设计流程的管理。定义结构体类型描述汽车。



汽车一般由发动机、底盘、车身和电气设备等四个基本部分组成。

发动机由曲柄连杆机构、配气机构和燃料供给系、冷却系、润滑系、点火系、起动系组成。

底盘由传动系、行驶系、转向系和制动系四部分组成。

车身包括壳体、车门、车窗、车身内外装饰件和车身附件、座椅以及通风、暖风、冷气、空气调节装置等。

电气设备由电源和用电设备两大部分组成。电源包括蓄电池和发电机；用电设备包括发动机的启动系、汽油机的点火和其他用电装置。

```

struct 汽车{
    曲柄连杆机构 a;
    配气机构 b;
    燃料供给系 c;
    冷却系 d;
    润滑系 e;
    点火系 f;
    起动系 g;
    传动系 h;
    行驶系 i;
    制动系 j;
    壳体 k;
    车门 l;
    车窗 m;
    车身内外装饰件 n;
    车身附件 o;
    座椅 p;
    通风 q;
    暖风 r;
    冷气 s;
    空气调节装置 t;
    蓄电池 u;
    发电机 v;
    用电装置 w;
};

```



很多事物的组成不能仅仅通过简单数据类型的成员来描述，他们的组成必须要逐级用另一个复合数据类型来描述，这样才能更加方面的认识事物。

```

struct 发动机{
    //...
};
struct 车身{
    //...
};
struct 底盘
    //...
};
struct 电气设备{
    //...
};
struct 汽车{
    struct 发动机 eg;
    struct 车身 bo;
    struct 底盘 ch;
    struct 电气设备 ed;
};

```



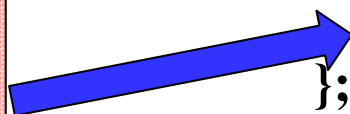
8.1.3 结构体嵌套

定义结构体类型，描述具有以下属性的学生。

学号	姓名	性别	出生日期			数学	英语	计算机 原理	程序 设计
			年	月	日				

```
struct Student{  
    long studentID;  
    char studentName[30];  
    char studentSex;  
    int yearOfbirth;  
    int monthOfbirth;  
    int dayOfbirth;  
    int math;  
    int english;  
    int computer;  
    int program;  
};
```

```
struct Student{  
    long studentID;  
    char studentName[30];  
    char studentSex;  
    int yearOfbirth;  
    int monthOfbirth;  
    int dayOfbirth;  
    int score[4];  
};
```





8.1.3 结构体嵌套

定义结构体类型，描述具有以下属性的学生。

学号	姓名	性别	出生日期			数学	英语	计算机 原理	程序 设计
			年	月	日				

//定义日期结构体

```
struct Date{  
    int year;  
    int month;  
    int day;  
};
```

//定义学生结构体，包含日期类型的成员

```
struct Student{  
    long studentID;  
    char studentName[30];  
    char studentSex;  
    Date birthday;  
    int score[4];  
};
```





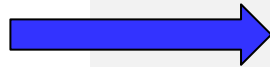
8.1.3 结构体嵌套

//定义日期结构体

```
struct Date{  
    int year;  
    int month;  
    int day;  
};
```

//定义学生结构体，包含日期类型的成员

```
struct Student{  
    long studentID;  
    char studentName[30];  
    char studentSex;  
    Date birthday;  
    int score[4];  
};
```

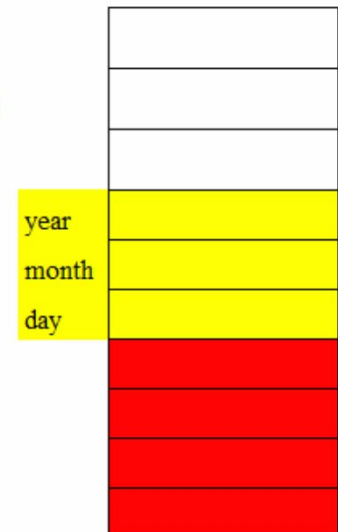


嵌套了struct Date类型成员的struct Student类型变量内存结构

studentID
studentName
studentSex

birthday

score



//分析程序输出

```
#include <stdio.h>
```

```
struct Date{ //要放在Student前
```

```
    int year;
```

```
    int month;
```

```
    int day;
```

```
};
```

```
struct Student{
```

```
    long studentID;
```

```
    char studentName[30];
```

```
    char studentSex;
```

```
    Date birthday;
```

```
    int score[4];
```

```
};
```

```
int main(){
```

```
    struct Student stu1 = {100310224,"张明",'M',{2001,10,8},{90,88,85,89}};
```

```
    printf("%d %s %c", stu1.studentID , stu1.studentName , stu1.studentSex );
```

```
    printf("%d %d %d", stu1.birthday.year , stu1.birthday.month , stu1.birthday.day );
```

```
    printf("%d %d %d %d", stu1.score[0] , stu1.score[1], stu1.score[2], stu1.score[3] );
```

```
    return 0;
```

```
}
```

stu1.birthday.year

stu1.birthday

stu1.score

stu1.score[0]

studentID
studentName
studentSex

	100310224
	张明
	M
year	2001
month	10
day	8
score[0]	90
score[1]	88
score[2]	85
score[3]	89

访问时，通过选择符逐级细化，直到基本类型的成员

//分析程序输出

```
#include <stdio.h>
```

```
struct Date{
```

```
    int year;
```

```
    int month;
```

```
    int day;
```

```
};
```

```
struct Student{
```

```
    long studentID;
```

```
    char studentName[30];
```

```
    char studentSex;
```

```
    Date birthday;
```

```
    int score[4];
```

```
};
```

```
int main(){
```

```
    struct Student stu1 = {100310224,"张明",'M',{2001,10,8},{90,88,85,89}};
```

```
    printf("%d %s %c", stu1.studentID , stu1.studentName , stu1.studentSex );
```

```
    printf("%d %d %d", stu1.birthday.year, stu1.birthday.month, stu1.birthday.day );
```

```
    printf("%d %d %d %d", stu1.score[0] , stu1.score[1] , stu1.score[2], stu1.score[3] );
```

```
    return 0;
```

```
}
```

studentID
studentName
studentSex

birthday

score

	100310224
	张明
	M
year	2001
month	10
day	8
	90
	88
	85
	89



//分析程序输出

#include <stdio.h>

//Date,Student的定义见前例,此处为节省篇幅,

int main(){

struct Student stu1 = {100310224,"张明",'M',
struct Student stu2;

stu2 = stu1; //结构体变量整体赋值

strcpy(stu2.studentName, "李明");

printf("%d,%s %c", stu2.studentID , stu2.studentName , stu2.studentSex);

printf("%d %d %d", stu2.birthday.year , stu2.birthday.month , stu2.birthday.day);

printf("%d %d %d %d", stu2.score[0] , stu2.score[1] , stu2.score[2] ,stu2.score[3]);

//通过指针访问

struct Student *ps = &stu1;

printf("%d %s %c", ps->studentID , ps->studentName , ps->studentSex);

printf("d %d %d", **ps->birthday.year** , **ps->birthday.month** , **ps->birthday.day**);

printf("%d %d %d %d", **ps->score**[0], ps->score[1], ps->score[2], ps->score[3]);

return 0;

}

```
100310224 李明 M
2001 108
90 88 85 89
100310224 张明 M
2001 108
90 88 85 89
```

//分析程序输出

```
#include <stdio.h>
```

```
struct Date{
```

```
    int year;
```

```
    int month;
```

```
    int day;
```

```
};
```

```
struct Student{
```

```
    long studentID;
```

```
    char studentName[30];
```

```
    char studentSex;
```

```
    Date birthday;
```

```
    int score[4];
```

```
};
```

```
int main(){
```

```
    struct Student stu1 = {100310224,"张明",'M',{2001,10,8},{90,88,85,89}};
```

```
    struct Student *ps = &stu1;
```

```
    printf("%d%s %c", ps->studentID, ps->studentName, ps->.studentSex);
```

```
    printf("%d %d %d", ps->birthday.year, ps->birthday.month, ps->birthday.day);
```

```
    printf("%d %d %d", ps->score[0], ps->score[1], ps->score[2], ps->score[3]);
```

```
    return 0;
```

```
}
```





7.1.4 结构体中的指针成员

//定义日期结构体

```
struct Date{  
    int year;  
    int month;  
    int day;  
};
```

//定义学生结构体，包含指向日期类型的指针成员

```
struct Student{  
    long studentID;  
    char studentName[30];  
    char studentSex;  
    Date* birthday;  
    int score[4];  
};
```

struct Student类型变量内存结构

studentID
studentName
studentSex
birthday
score



->?

//分析程序

```
#include <stdio.h>
```

```
struct Date{
```

```
    int year;
```

```
    int month;
```

```
    int day;
```

```
};
```

```
struct Student{
```

```
    long studentID;
```

```
    char studentName[30];
```

```
    char studentSex;
```

```
    Date* birthday;
```

```
    int score[4];
```

```
};
```

```
int main(){
```

```
    struct Student stu1 = {100310224,"张明",'M',NULL,{90,88,85,89}};
```

```
    stu1.birthday->year = 2001;
```

```
    stu1.birthday->month = 10;
```

```
    stu1.birthday = new Date;
```

```
    stu1.birthday->year = 2001;
```

```
    stu1.birthday->month = 10;
```

```
    stu1.birthday->day = 8;
```

```
    return 0;
```

```
}
```

代码是否正确？

//分析程序

```
#include <stdio.h>
```

//Date, Student定义见前例。

```
int main(){
    struct Student stu1 = {100310224,"张明",'M',NULL,{90,88,85,89}};
    stu1.birthday = (struct Date*)malloc(sizeof(struct Date));
    stu1.birthday->year = 2001;
    stu1.birthday->month = 10;
    stu1.birthday->day = 8;
    struct Student stu2; stu2 = stu1;
    printf(“%d %d %d”, stu2.birthday->year, stu2.birthday->month,
stu2.birthday->day);
    stu2.birthday->year = 2003;
    printf(“%d”, stu1.birthday->year); //??????

    stu2.birthday = (struct Date*)malloc(sizeof(struct Date));
    stu2.birthday->year = 2005;
    printf(“%d”, stu2.birthday->year);

    return 0;
}
```

需要考虑，每个结构体变量中的指针成员是否需要指向自己独立的一份空间。指针成员指向的空间需要通过动态分配。



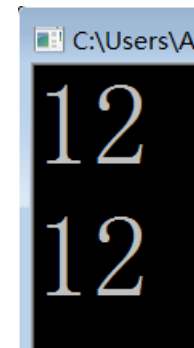
7.1.4 结构体变量及成员的存储

结构体变量所占用的内存是所有成员变量的内存总和吗？

```
#include <stdio.h>
```

```
struct sample{  
    char m1;  
    int m2;  
    char m3;  
};
```

```
int main(){  
    printf(“%d\n”, sizeof(struct sample) );  
    struct sample s;  
    printf(“%d\n”, sizeof(s) );  
    return 0;  
}
```



Why?





8.1.4 结构体变量及成员的存储

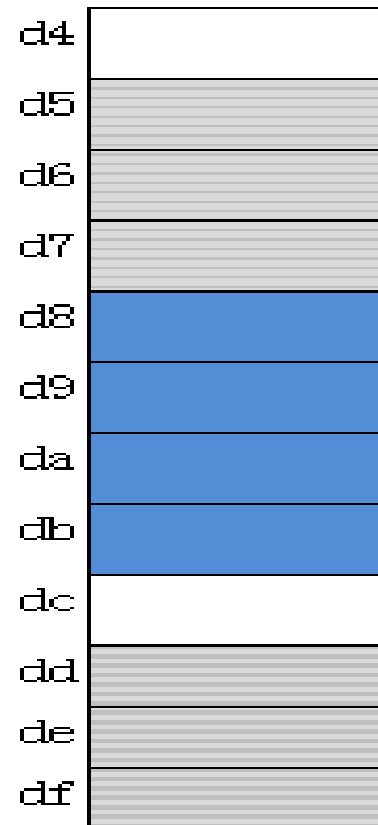
结构体变量所占用的内存>=所有成员变量的内存总和

```
#include <stdio.h>
```

```
struct sample{  
    char m1;  
    int m2;  
    char m3;  
};
```

```
int main( ){  
    struct sample s1;  
    printf(“%d\n”, (void*)&s1.m1 );  
    printf(“%d\n”, &s1.m2 );  
    printf(“%d”, (void*)&s1.m3);  
}
```

0x28fed4
0x28fed8
0x28fedc





8.1.4 结构体变量及成员的存储

结构体变量所占用的内存 \geq 所有成员变量的内存总和

事实上，所有数据类型在内存中都是从偶数地址开始存放的
且结构所占的实际空间一般是按照机器字长对齐的
不同的编译器、平台，对齐方式会有变化
结构体变量的成员的存储对齐规则是与机器相关的
具有特定数据类型的数据项大小也是与机器相关的
所以一个结构体在内存中的存储格式也是与机器相关的



8.1.5 结构体变量数组

写一个班级学生基本信息管理的程序，学生基本信息如下，每个班级的学生不超过50人。

学号	姓名	性别	出生年	数学	英语	计算机原理	程序设计
100310121	王 刚	男	1991	72	83	90	82
100310122	李小明	男	1992	88	92	78	78
100310123	王丽红	女	1991	98	72	89	66
100310124	陈莉莉	女	1992	87	95	78	90
...							

```
struct Student s[50];
```



8.1.5 结构体变量数组

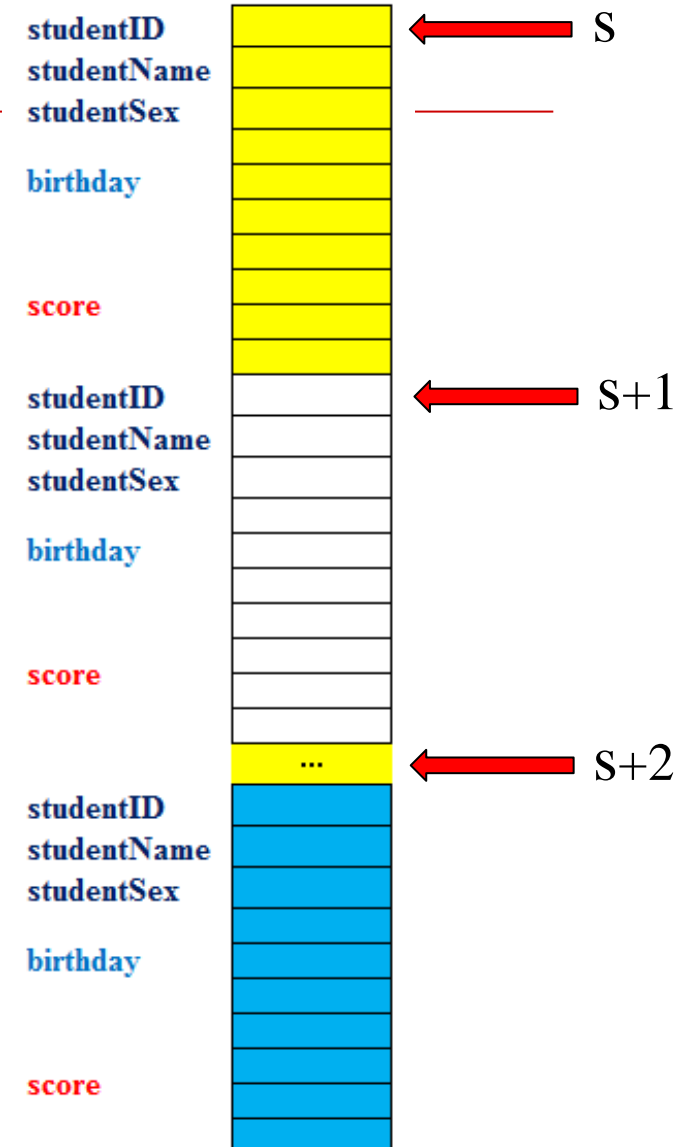
```
struct Student s[50];
```

s为struct Student*类型常量

```
for(int index=0; index<50; index++){  
    printf("%s\n", (s+index)->studentName );  
    printf("%d\n", s[index].studentID );  
}
```

计算数组s的元素个数:

`sizeof(s)/sizeof(struct Student)`





作业

S

1. 分析、运行课堂所有代码。
2. 用结构体实现员工管理系统。



河海大学 计算机与信息学院

欢迎交流

