清華大學
Tsinghua University

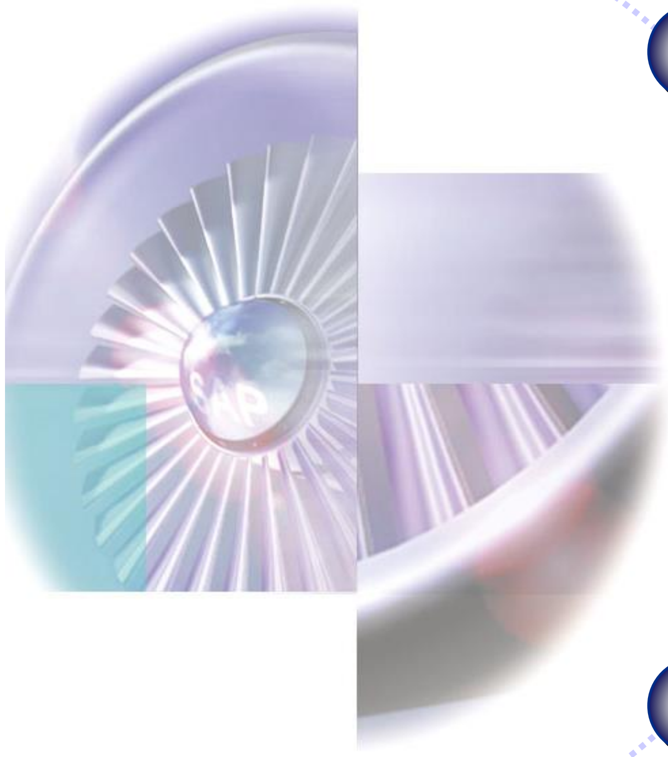
程序设计基础
Fundamental of Programming

清华大学软件学院

刘玉身

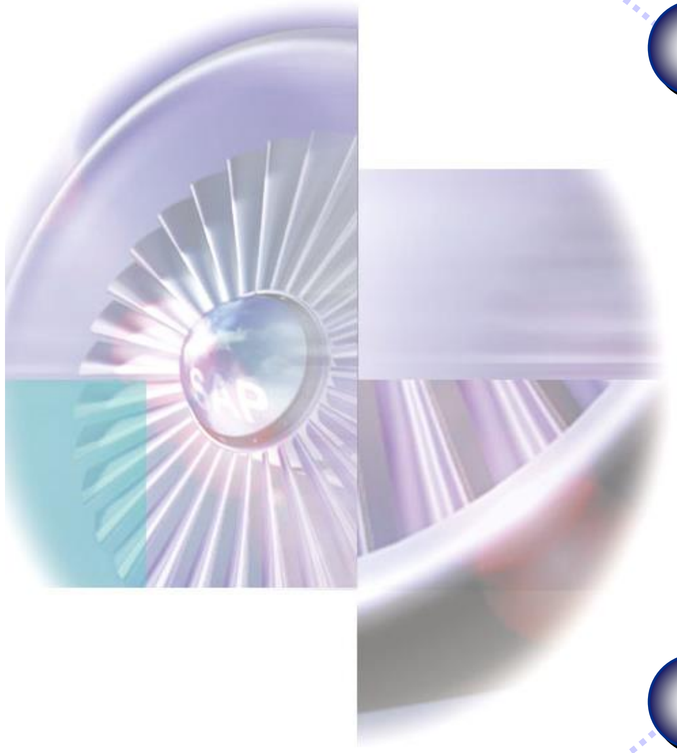
liuyushen@tsinghua.edu.cn

Lecture 7: 结构体



1. 结构体的定义和使用
2. 结构体数组与指针
3. 结构体作为函数参数
4. 链表

Lecture 7: 结构体



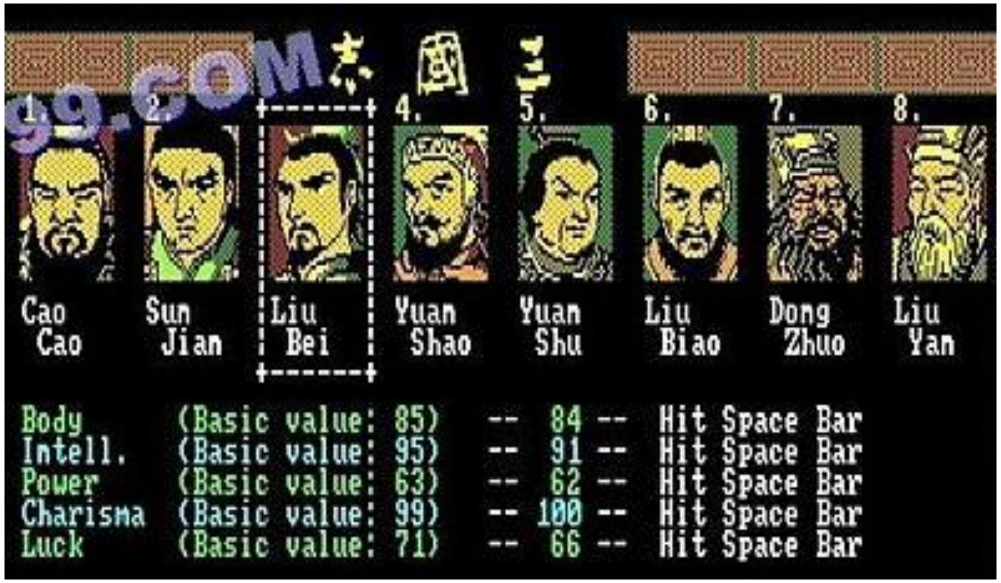
1 结构体的定义和使用

2 结构体数组与指针

3 结构体作为函数参数

4 链表

引言



```
1.     2.      4.     5.     6.     7.      8.
Cao    Sun    Liu    Yuan   Yuan   Liu    Dong   Liu
Cao    Jian   Bei    Shao   Shu    Biao   Zhuo   Yan

Body      (Basic value: 85)  --  84 --  Hit Space Bar
Intell.   (Basic value: 95)  --  91 --  Hit Space Bar
Power     (Basic value: 63)  --  62 --  Hit Space Bar
Charisma  (Basic value: 99)  -- 100 --  Hit Space Bar
Luck      (Basic value: 71)  --  66 --  Hit Space Bar
```

刘备的人马

姓名	体力	智力	武力	魅力	运气
刘备	84	91	62	100	66
诸葛亮	80	100	70	95	80
关羽	95	88	99	85	80
张飞	95	85	99	70	70
赵云	100	90	95	90	80
...					

新君主的人马

姓名	体力	智力	武力	魅力	运气
新君主	84	91	90	90	85
随从	80	80	80	80	80

人才最宝贵！

什么数据结构？

1. 单个人物
2. 所有人物

【结构体】

由一个或多个变量（类型可以相同，也可以不同，称为字段或成员变量）所组成的一个组合项。

A structure is a collection of one or more variables, possibly of different types, grouped together under a single name for convenient handling.

结构体的定义

定义一个结构体(struct)的步骤通常是：

1. 定义一个新的结构体类型，并指明它内部的各个成员变量；
2. 使用该类型来定义相应的结构体变量。

结构体类型的定义只在程序的开头出现一次，而结构体变量的声明可以根据需要在程序中出现多次。

定义一个结构体类型

```
struct  <结构体标记>
{
        类型名1   成员变量1;
        ……
        类型名n   成员变量n;
};
```

合法标识符
可省:无名结构体

struct是关键字,
不能省略

成员类型可以是
基本型或构造型

定义结构体变量:

```
struct  <结构体标记>  变量1, 变量2, …;
struct General
{
     char Name[20];        // 姓名
     int  Body;            // 体力
     int  Intelligence;    // 智力
     int  Power;           // 武力
     int  Charisma;        // 魅力
     int  Luck;            // 运气
};
```

分号

结构体类型定义描述结构的组织形式，不分配内存

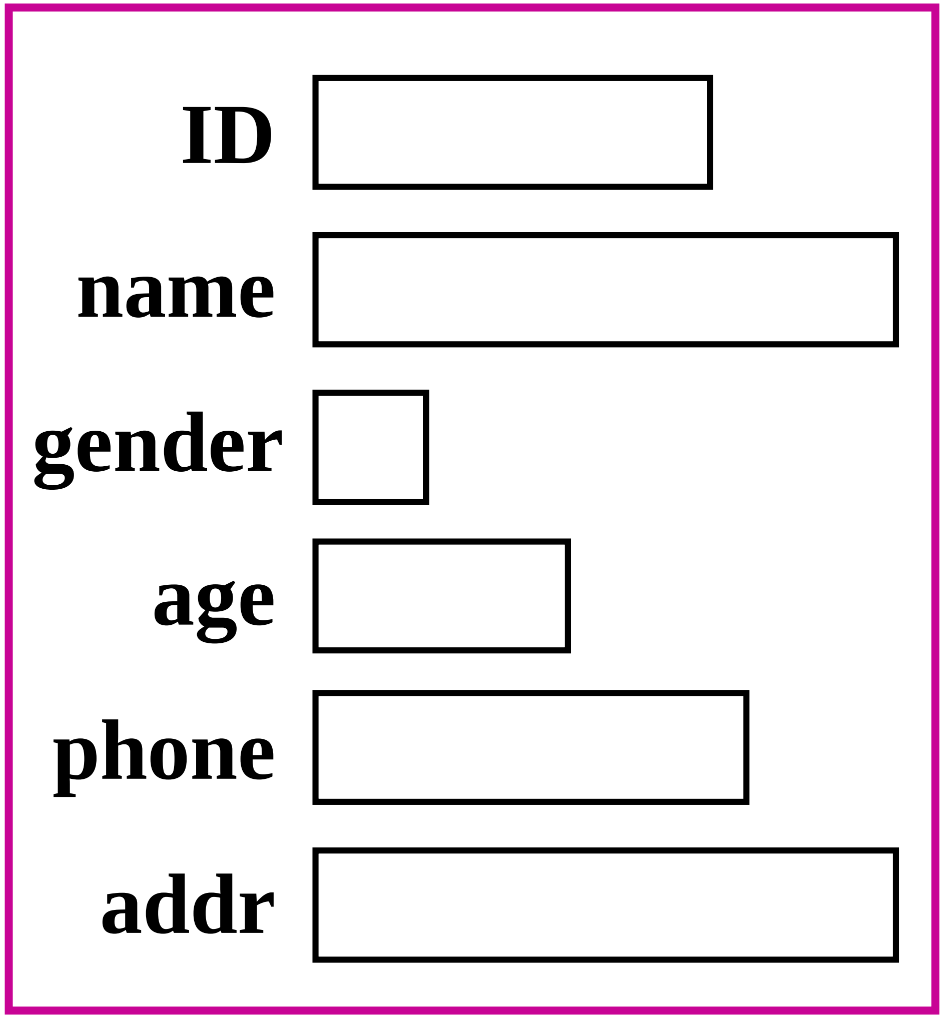
```
struct General LiuBei;
struct General GuanYu, ZhangFei;
```

在定义结构体变量后，将为它分配相应的内存空间。

结构体变量的长度 = ∑其各个成员变量长度之和?

x

```
struct  student
{
      char  ID[7];
      char  name[20];
      char  gender;
      int  age;
      char  phone[9];
      char  addr[30];
} x;
```



sizeof(x) = 72   7 + 20 + 1 + 4 + 9 + 30 = 71?

结构体变量的长度

```
struct  ID
{
      char    ch;
      double dd;
} y;
```

sizeof(y) = 16 ?

ANSIC标准中并没有规定: 相邻声明的变量在内存中一定相邻。

1. 在Win32下，结构体大小为结构体中最宽基本类型成员大小的整数倍，如有需要，编译器会在最末一个成员之后填充字节;
2. GNU GCC原则不同;
3. 结构体变量的长度 >= 其成员变量长度之和。
4. C语言结构体内存对齐问题

结构体变量的使用

对一个结构体变量的最基本的操作就是去访问它的各个成员变量。访问的方式为：

结构体变量名.成员变量名

一个成员变量就是一个普通的变量，可以对它进行各种通常的变量操作。

```c
struct General x;

strcpy(x.Name, "刘备");

if(x.Charisma > 95)
    printf("%s是仁君", x.Name);

if(x.Power > 90 && x.Intelligence < 70)
    printf("%s是一介武夫", x.Name);
```

Can't And Can

- 不能做什么？
  - ☺ 不能直接比较两个结构体变量，
    struct student x, y；~~(x > y);~~ ~~(x != y);~~
  - ☺ 不能用**scanf/printf**来输入或输出整个结构体变量。

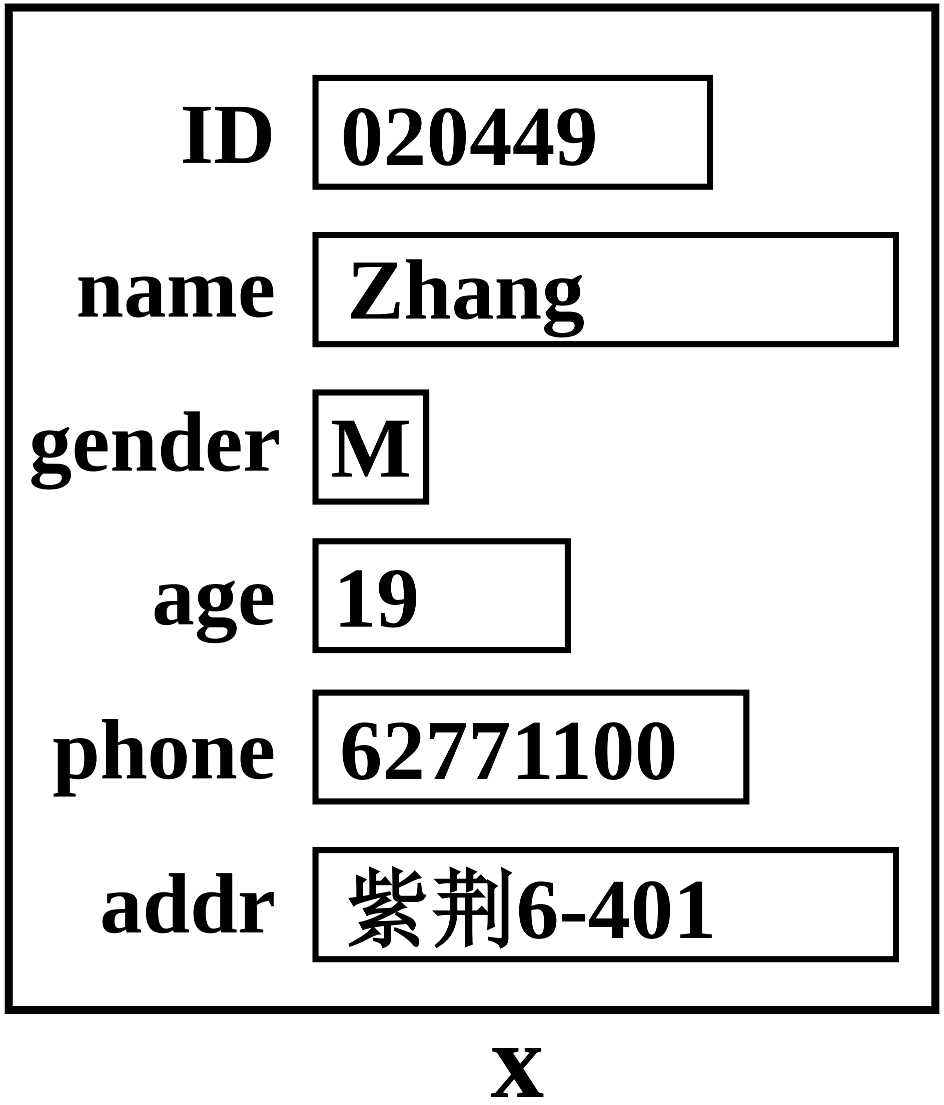
Can't And Can

- 能做什么？
  - ☺ 能够进行结构体变量的整体赋值，如：
    struct student x, y; y = x; （why?）
  - ☺ 能够定义一个返回值类型为结构体类型的函数，把被调用函数当中的某个结构体变量的值返回给主调函数。

```
struct  student  x, y;
…
y  =  x;
```

等价于

```
strcpy(y.ID,   x.ID);
strcpy(y.name,  x.name);
y.gender  =  x.gender;
y.age  =  x.age;
strcpy(y.phone,  x.phone);
strcpy(y.addr,   x.addr);
```



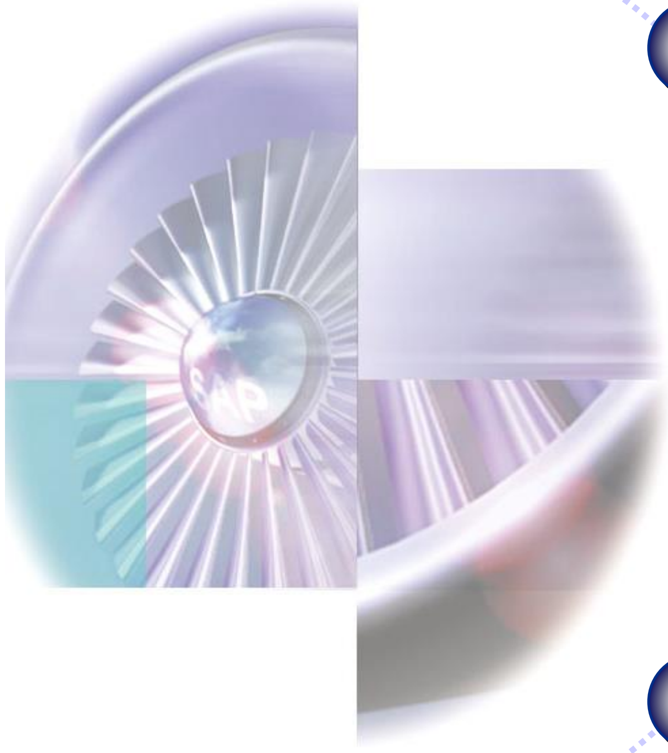
示例1:统计候选人得票

```c
#include <stdio.h>
#include <string.h>
struct Person
{
    char name[20];
    int count;
} p[3]={"Li", 0, "Zhang", 0, "Wang", 0}; //初始化
int main()
{
    int i, j;
    char str[20];
    for(i = 1; i <= 10; i++)
    {
      scanf("%s", str);
      for(j = 0; j < 3; j++)
          if(strcmp(str, p[j].name) == 0)
            p[j].count++;
    }
    for(i = 0; i < 3; i++)
      printf("%-5s: %d\n", p[i].name, p[i].count);
}
```

```
Li
Zhang
Wang
Li
Li
Zhang
Wang
Wang
Wang
Wang
```

```
Li    : 3
Zhang: 2
Wang : 5
```

Lecture 7: 结构体

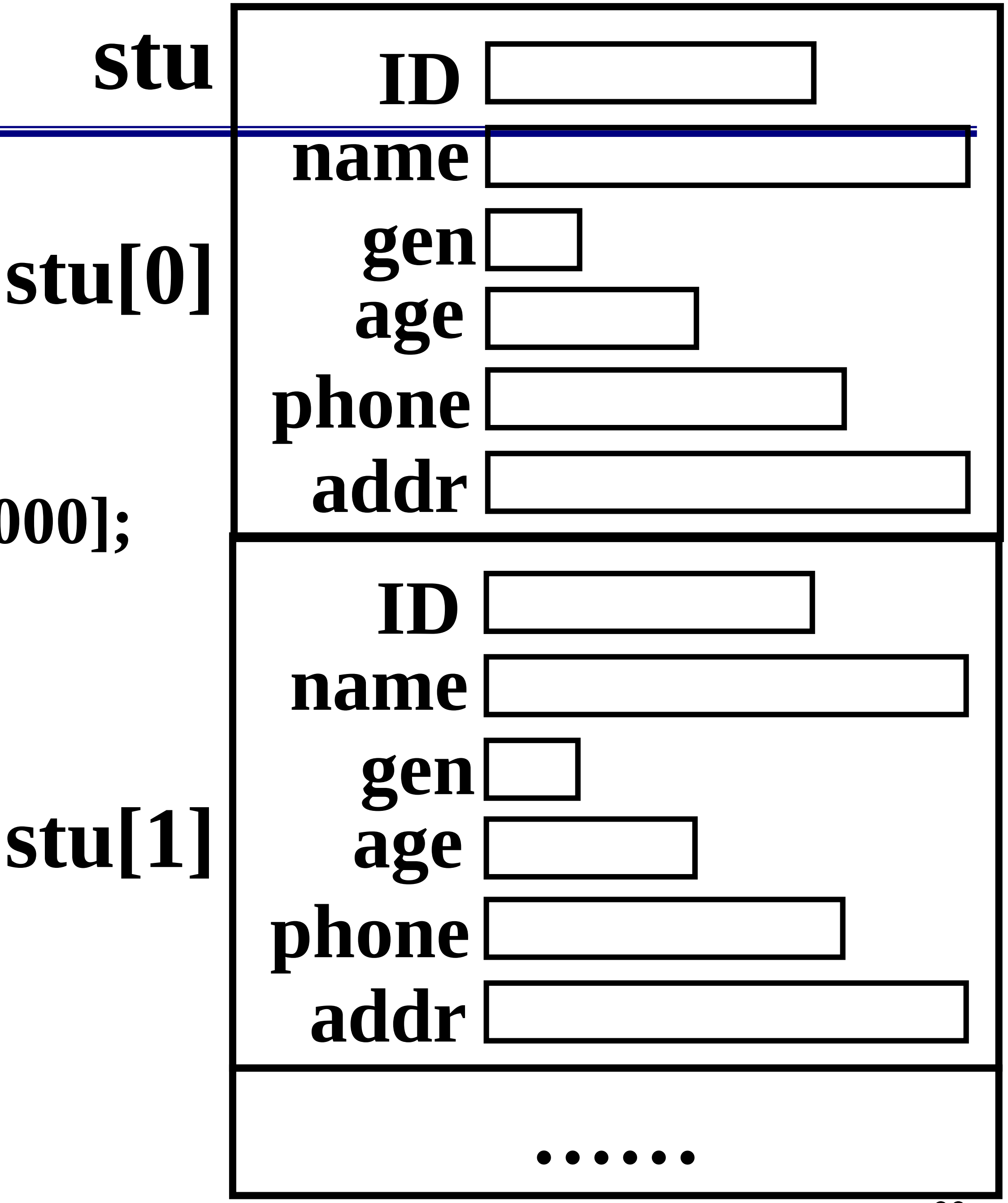


1. 结构体的定义和使用
2. 结构体数组与指针
3. 结构体作为函数参数
4. 链表

结构体数组

结构体变量

结构体数组

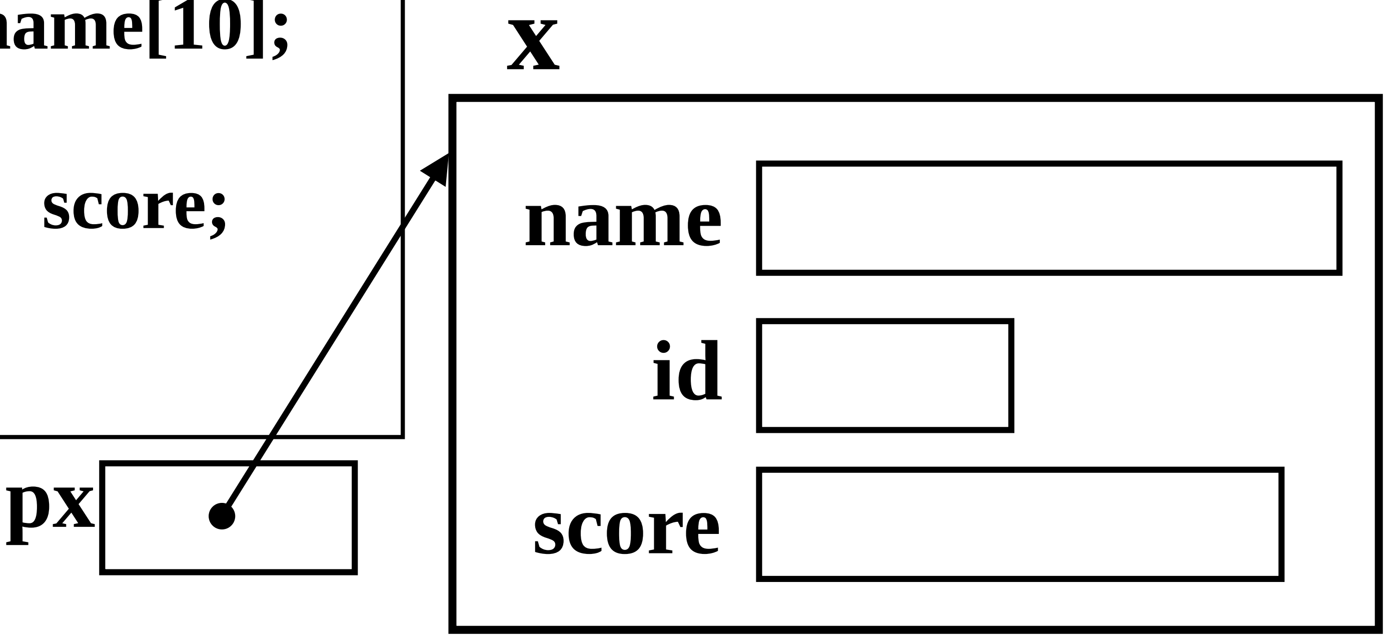


```
struct student stu[1000];
```

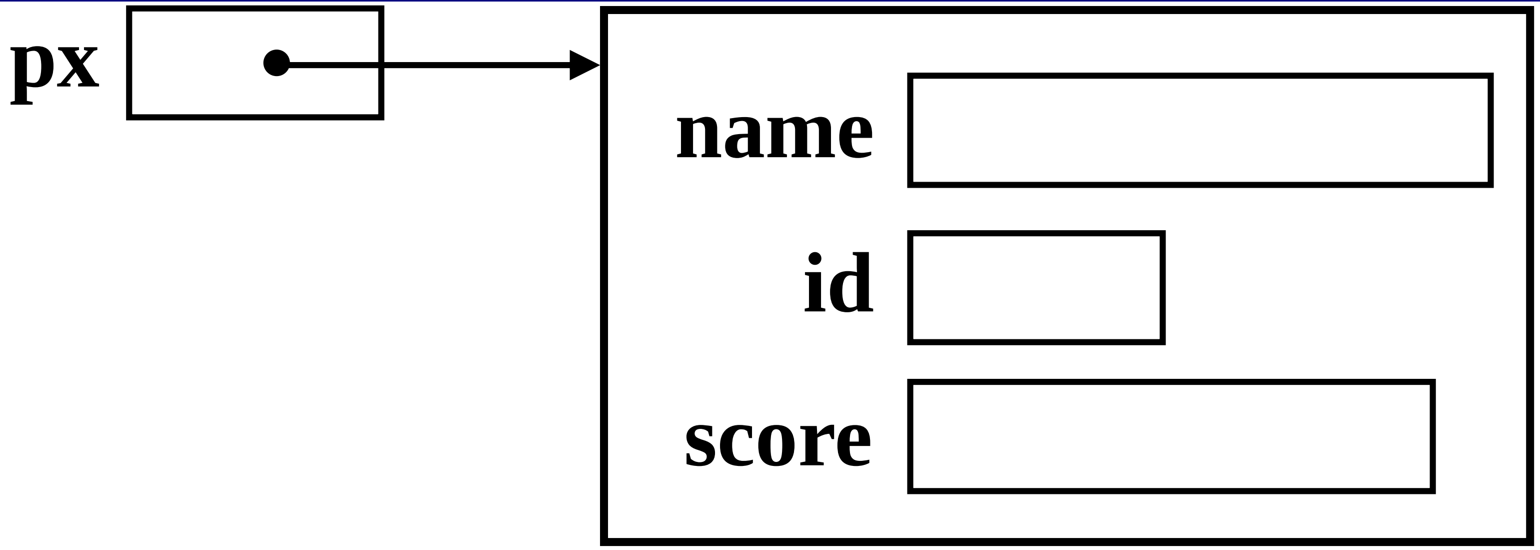
结构体与指针

新类型：基类型为结构体类型的指针类型。

```
struct StudentRecord
{
        char   name[10];
        int   id;
        double   score;
} x,  *px;
px  =  &x;
```



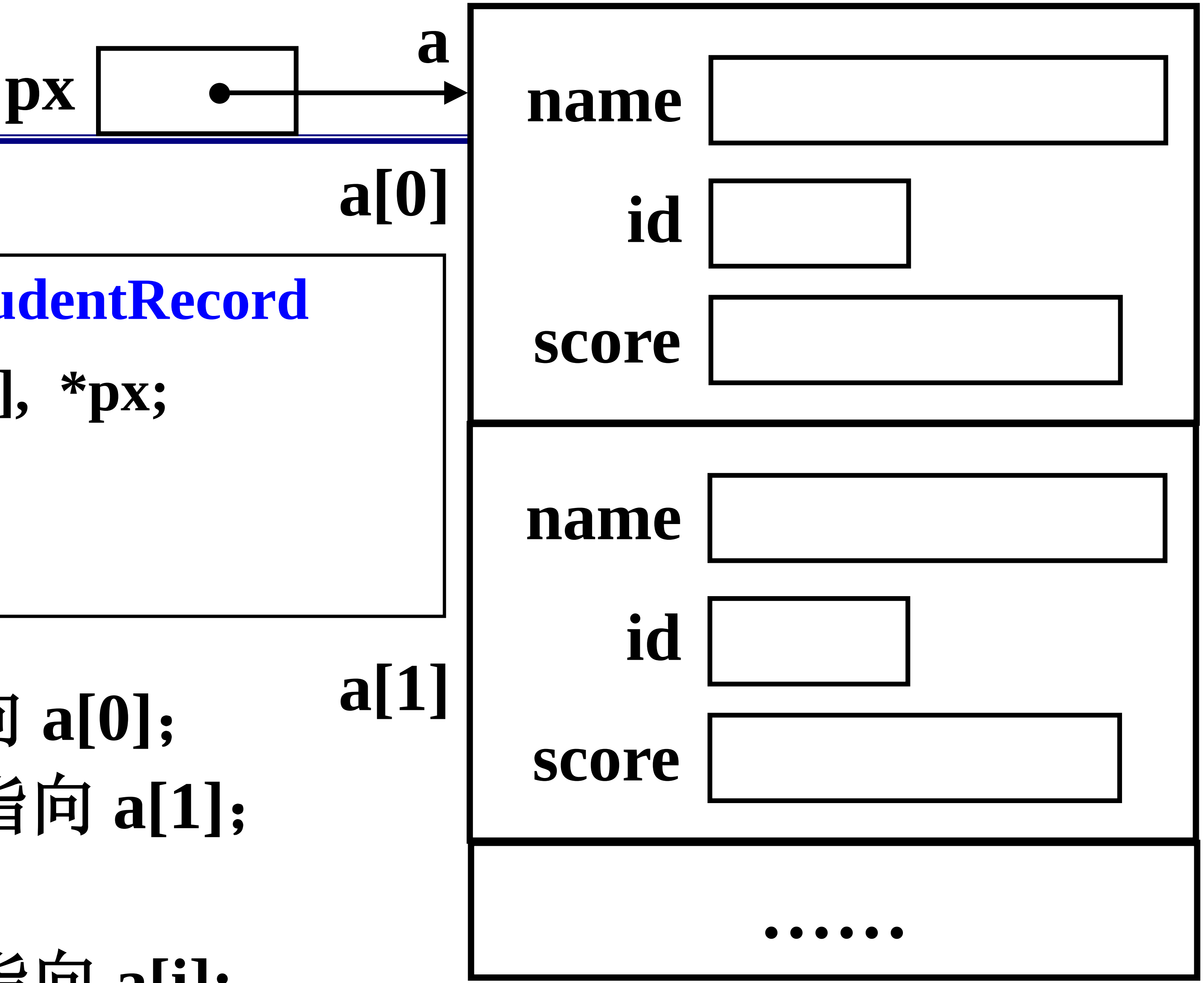
x = *px



结构体及指针应用广泛！

如何来访问x的成员变量？

1. x.成员变量名，如：x.name, x.id;
2. (*px).成员变量名，如：(*px).name, (*px).id;
3. px->成员变量名， “->”称为指向运算符或箭头运算符，如：px->name, px->id。



```
struct  StudentRecord
        a[10],  *px;
…
px  =  a;
```

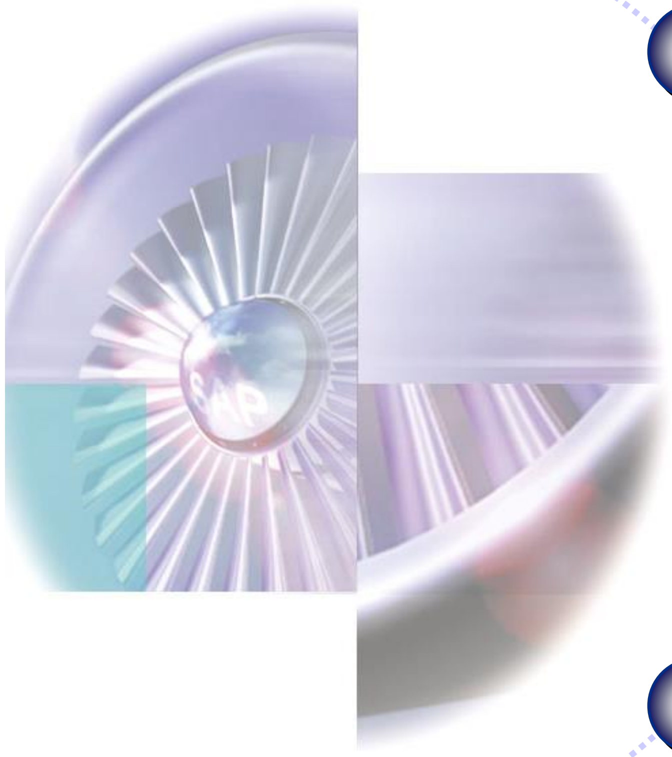
px 指向 a[0];

px + 1指向 a[1];

…

px + i 指向 a[i];

Lecture 7: 结构体



1 结构体的定义和使用

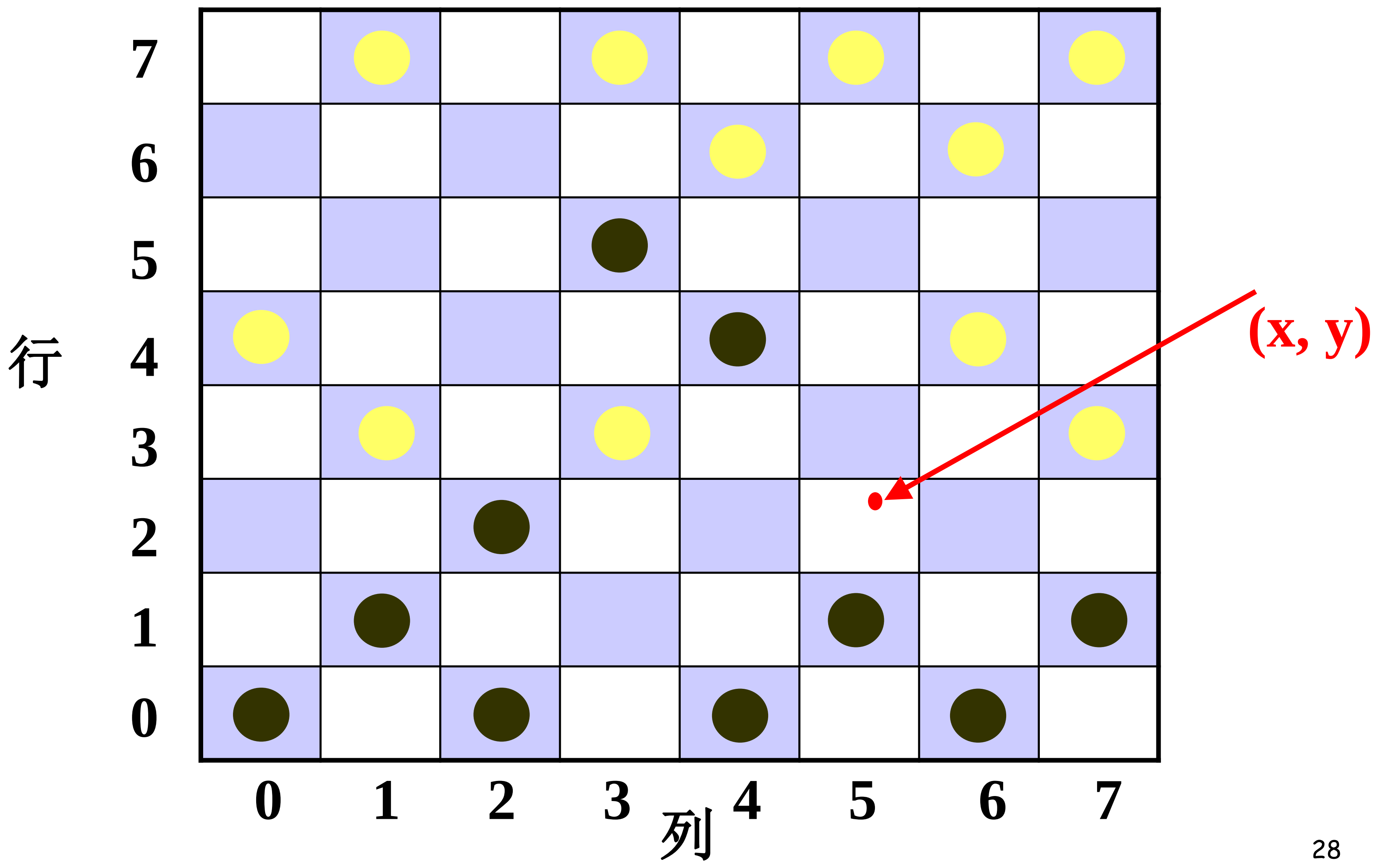
2 结构体数组与指针

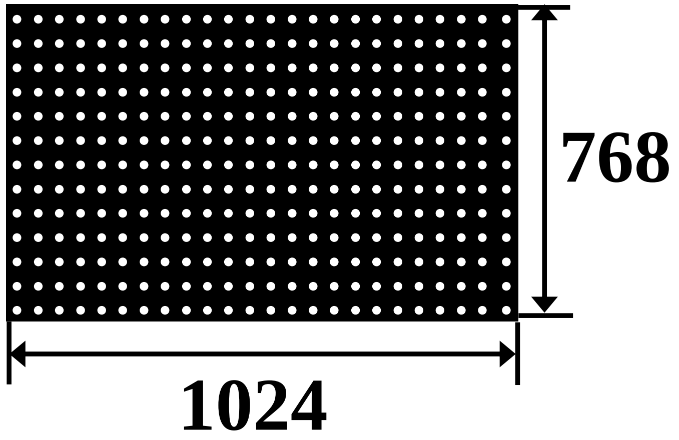
3 结构体作为函数参数

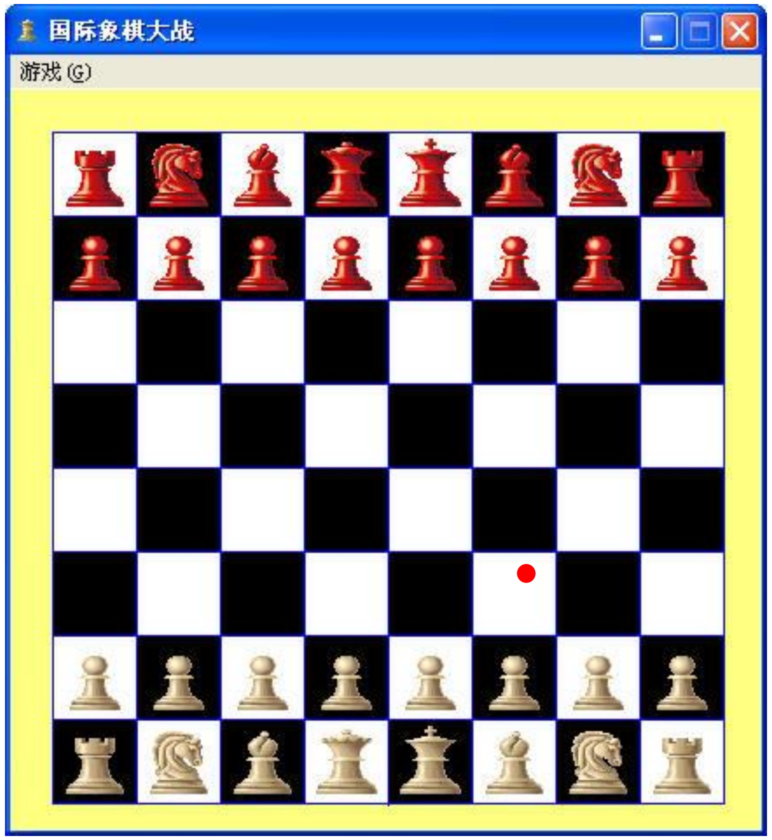
4 链表

- 函数参数的传递
  - ☺ 普通变量：传值；
  - ☺ 数组：传地址；
  - ☺ 结构体：传值！

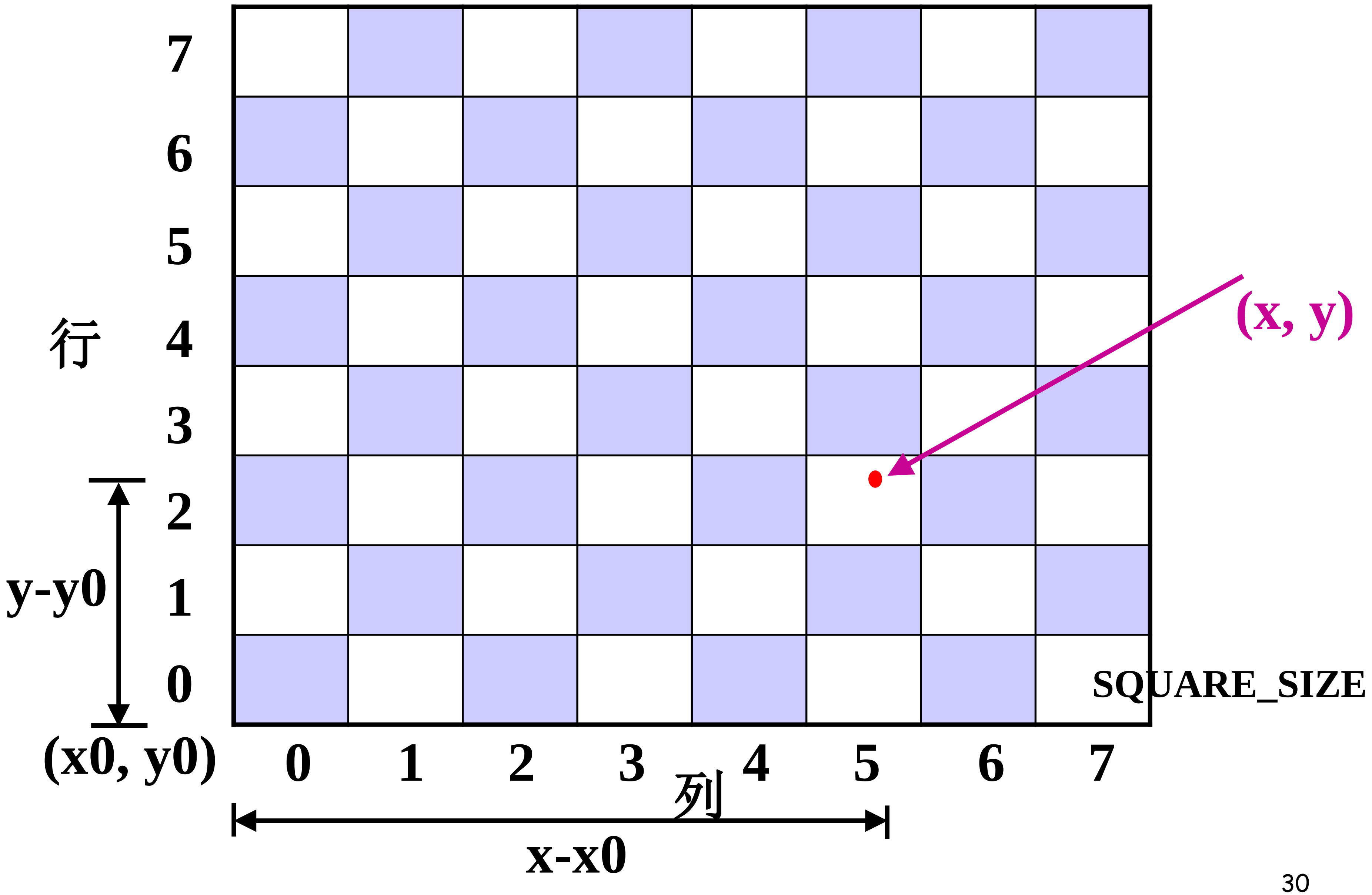
问题：屏幕坐标到棋盘坐标的转换




768
1024


国际象棋大战
游戏 (G)

问题分析：



```
struct ScreenCoor //屏幕坐标
{
    int x, y;
};

struct BoardCoor  //棋盘坐标
{
    int row, col;
};
```

```
#define x0 40
#define y0 20
#define SQUARE_SIZE 10
void main()
{
    struct ScreenCoor s;
    struct BoardCoor  b;
    scanf("%d %d", ________); // 输入屏幕坐标
    screen_to_board(________); // 函数调用
    printf("%d %d\n", b.row, b.col);
}

void screen_to_board(??screen, ??board)
{
    ____________________________;
    ____________________________;
}
```

```
#define x0 40
#define y0 20
#define SQUARE_SIZE 10
void main()
{
    struct ScreenCoor s;
    struct BoardCoor  b;
    scanf("%d %d", &s.x, &s.y); // 输入屏幕坐标
    screen_to_board(s, &b);     // 函数调用
    printf("%d %d\n", b.row, b.col);
}

void screen_to_board(struct ScreenCoor screen,
                     struct BoardCoor *board)
{
   board->row = (screen.y - y0) / SQUARE_SIZE;
   board->col = (screen.x - x0) / SQUARE_SIZE;
}
```

typedef:定义类型

功能：用自定义名字为已有数据类型命名

形式：**typedef** type name;

分号

```
typedef   int          INTEGER;
typedef   float        REAL;

// 类型定义后，与已有类型一样使用
INTEGER    a, b, c;
REAL        f1, f2;
```

补充说明

- **typedef** 没有创造新数据类型
- **typedef** 是定义类型, 不能定义变量
- **typedef** 与 **#define** 不同

#define	**typedef**
预编译时处理	编译时处理
简单字符置换	为已有类型命名

```
typedef struct date
{
    int month;
    int day;
    int year;
} DATE;

struct date  d[2];
DATE  birthdays[100], *p;
```

枚举类型

枚举类型的定义格式：

```
enum  枚举类型名 {变量1, 变量2, …};
```

示例：
```
enum  weekday {Sun, Mon, Tue, Wed, Thu, Fri, Sat};
```

- 枚举常量是一种符号常量，起始值从 0 开始，递增值为 1
- 可以用 = 为枚举常量定值
- 枚举定义中的标识符必须唯一
- 用枚举类型可以定义各种离散的、非数值型数据

```
enum  weekday {Sun, Mon=100, Tue, Wed, Thu, Fri, Sat};
```
此时，Sun 为0，Mon~Sat 分别为100 ~105。

枚举变量的使用举例：

```
weekday  day ;
day = Mon;              // OK
day = Sunday;           // Error
day = 2;                // Error
day = (weekday)2;       // OK
//如一定要把数值赋予枚举变量，则必须用强制类型转换
```

枚举元素（枚举成员）不是字符常量也不是字符串常量，使用时不要加单、双引号。

枚举类型作用：以一系列字符串，来表示整型数字，起到更方便阅读维护代码的效果。

枚举变量的使用

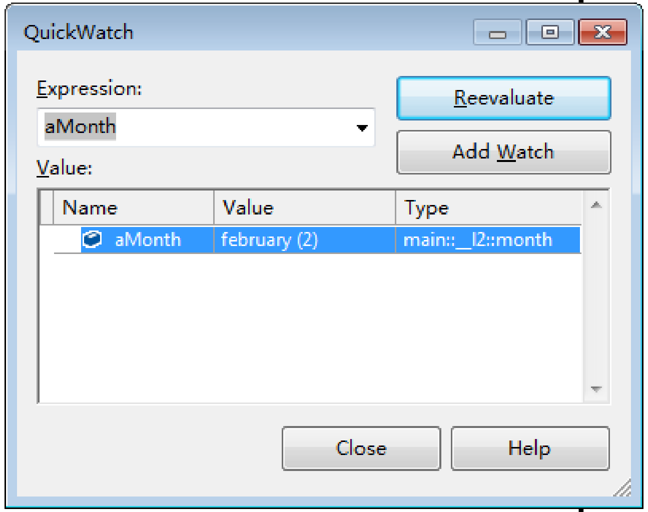
问题描述：

输入月份（整数）。

请编写一个程序，利用枚举型变量输出对应月份的天数。

枚举变量的使用

```
// Program to print the number of days in a month
#include <stdio.h>
int main ()
{
    enum month { january = 1, february, march, april, may, june,
            july, august, september, october, november, december};
    enum  month   aMonth;
    int days;
    printf ("Enter month number: ");
    scanf ("%i", &aMonth);
switch (aMonth ) {
case january: case march: case may: case july:
case august: case october: case december:
        days = 31;  break;
case april: case june: case september: case november:
        days = 30;  break;
case february:
        days = 28;  break;
default:
        printf ("bad month number\n");
        days = 0;   break;
}
if ( days != 0 )
        printf ("Number of days is %i\n", days);
if ( aMonth == february )
        printf ("...or 29 if it’s a leap year\n");
return 0;
}
```



Expression: aMonth

Name	Value	Type
aMonth	february (2)	main::_I2::month

```
Enter month number: 2
Number of days is 28
...or 29 if it’s a leap year
```

结构体应用

问题描述：

输入“今天”的日期（mm dd yyyy）。

请编写一个程序，利用结构体表示日期（struct date），来计算“明天”的日期（mm/dd/yyyy）。

问题分析

1. 如何用结构体来描述“日期”？
2. 如何针对今天的日期来计算明天的日期?

示例2: 明天的日期

```
#include <stdio.h>
int main ()
{
    struct date
    {
            int month;
            int day;
            int year;
    };
    struct date today, tomorrow;
    const int daysPerMonth[12] = { 31, 28, 31, 30, 31, 30, 31, 31, 30,
                                   31, 30, 31 };
    printf ("Enter today's date (mm dd yyyy): ");
    scanf ("%i%i%i", &today.month, &today.day, &today.year);
    if ( today.day != daysPerMonth[today.month - 1] ) {
        tomorrow.day = today.day + 1;
        tomorrow.month = today.month;
        tomorrow.year = today.year;
    }
    else if ( today.month == 12 ) { // end of year
        tomorrow.day = 1;
        tomorrow.month = 1;
        tomorrow.year = today.year + 1;
    }
    else { // end of month
        tomorrow.day = 1;
        tomorrow.month = today.month + 1;
        tomorrow.year = today.year;
    }
    printf ("Tomorrow's date is %i/%i/%i.\n", tomorrow.month,
        tomorrow.day, tomorrow.year );
    return 0;
}
```

```
Enter today's date (mm dd yyyy): 11 30 2013

Tomorrow's date is 12/1/2013.
```

还有何问题？ *thinking…*

问题分析(2)

1. 如何考虑闰年闰月的情况？
2. 如何判断闰年？
3. 如何编写子函数计算“明天”的日期?
4. 结构体变量如何作为参数和返回值传递？

示例2: 明天的日期（v2）

```
#include <stdio.h>
struct date
{
    int month;
    int day;
    int year;
};
// Function to find the number of days in a month
int numberOfDays (struct date d);

int main ()
{
    struct date today, tomorrow;
    printf ("Enter today's date (mm dd yyyy): ");
    scanf ("%i%i%i", &today.month, &today.day, &today.year);
```

1. 定义struct date是全局类型
2. struct date做为函数参数

```
    if ( today.day != numberOfDays (today) ) {
            tomorrow.day = today.day + 1;
            tomorrow.month = today.month;
            tomorrow.year = today.year;
    }
    else if ( today.month == 12 ) { // end of year
            tomorrow.day = 1;
            tomorrow.month = 1;
            tomorrow.year = today.year + 1;
    }
    else { // end of month
            tomorrow.day = 1;
            tomorrow.month = today.month + 1;
            tomorrow.year = today.year;
    }
    printf ("Tomorrow's date is %i/%i/%i.\n", tomorrow.month,
            tomorrow.day, tomorrow.year);
    return 0;
}
```

```
int isLeapYear (struct date d);
// Function to find the number of days in a month
int numberOfDays (struct date d)  {
    int days;
    const int daysPerMonth[12] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30,
    31 };
    if ( isLeapYear (d) == 1 && d.month == 2 )
        days = 29;
    else
        days = daysPerMonth[d.month - 1];
    return days;
}
// Function to determine if it's a leap year
int isLeapYear (struct date d) {
    int leapYearFlag;
    if ( (d.year % 4 == 0 && d.year % 100 != 0) || d.year % 400 == 0 )
        leapYearFlag = 1; // It's a leap year
    else
        leapYearFlag = 0; // Not a leap year
    return leapYearFlag;
}
```

```
Enter today's date (mm
dd yyyy): 02 28 2008

Tomorrow's date is
2/29/2008.
```

2008年是闰年

示例2: 明天的日期（v3）

```
#include <stdio.h>
struct date
{
    int month;
    int day;
    int year;
};
struct date  dateUpdate (struct date today);
int main (){
    struct date  thisDay, nextDay;
    printf ("Enter today's date (mm dd yyyy): ");
    scanf ("%i%i%i", &thisDay.month, &thisDay.day,
            &thisDay.year);
    nextDay = dateUpdate (thisDay);
    printf ("Tomorrow's date is %i/%i/%i.\n", nextDay.month,
            nextDay.day, nextDay.year );
    return 0;
}
```

定义一个子函数:
其形参和返回值都是struct date

```
int numberOfDays (struct date d);
// Function to calculate tomorrow's date
struct date dateUpdate (struct date today) {
    struct date tomorrow;
    if ( today.day != numberOfDays (today) ) {
            tomorrow.day = today.day + 1;
            tomorrow.month = today.month;
            tomorrow.year = today.year;
    }
    else if ( today.month == 12 ) { // end of year
            tomorrow.day = 1;
            tomorrow.month = 1;
            tomorrow.year = today.year + 1;
    }
    else { // end of month
            tomorrow.day = 1;
            tomorrow.month = today.month + 1;
            tomorrow.year = today.year;
    }
    return tomorrow;
}
```

```
Enter today's date (mm
dd yyyy): 02 28 2008

Tomorrow's date is
2/29/2008.
```

Lecture 8 - Summary

- **Topics covered:**
  - **Defining and using Structures**
  - **Functions and Structures**
  - **Initializing Structures. Compound Literals**
  - **Arrays of Structures**
  - **Structures Containing Structures and/or Arrays**
  - **Enumerated Data Types**
  - **The typedef Statement**