复习课

周水庚

计算机学院

2017年12月28日

考试时间/地点

- 时间: 2018年1月10日（周三）下午13：00-15：00
- 地点: ???
- 方式：闭卷考试

C语言关键词与函数

- C语言中有一些保留使用的关键词
- 预处理命令行中的词不是关键词，但也是保留使用的

- 除了main函数和sizeof()内置函数外，大部分编程调用的是环境提供的函数，不是C语言所固有的

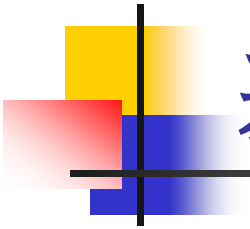
字符与整数

- 字符变量可以赋整数值
  - char c=9; int i=c,j; j=c-0; printf("%d,%d", i,j);
  - char c='9';int i=c, j; j=c-'0'; printf("%d,%d", i,j);

- 字符表示：转义字符
  - \ddd /* 8进制表示
  - \xdd /* 16进制表示
  - char c='\12'

整数的表示

- int i1=100, j1=-100, i2=0xaaaa, j2=-0xaaaa
- printf(“%x, %x”, i1, j1);
  - 64, ffffff9c
- printf(“%x, %x”, i2, j2);
  - aaaa, ffff5556



基本输入输出

- 掌握scanf()和printf()基本使用方法
  - char *s, x[10]; int a[20]
  - scanf(“%s”,s);
  - scanf(“%s”,x);
  - scanf(“%c”,&x[0]);
  - scanf(“%d”,&a[1]);

控制条件与循环终止

- int x; if (x==0)x=1 else x=0; printf("%d",x);
- int x; if (x=0)x=1 else x=2; printf("%d",x);

- int x,y; for(x=0,y=10;x=0;x++,y--)x++;
- int x,y; for(x=0,y=3;x=y;x++,y--)x++;

- for()循环语句中的break与continue
  - for(i=0;i<10;i++) if(i==5)break;
  - for(i=0;i>=0;i++) if(i!=5)continue;

一维数组初始化

- int e[5] = {0, 1, 2};
- int g[] = {5, 6, 7, 8, 9};
- int c[5] = {0, 1, 2, 3, 4, 5};

二维数组初始化

- 按行给二维数组的全部元素赋初值

```
int a1[2][3]={ {1, 2, 3},{4, 5, 6}};
```

- 按元素存储顺序给数组元素赋初值

```
int a2[2][3] = {1, 2, 3, 4, 5, 6};
```

- 按行给数组的部分元素赋初值

```
int a3[2][3]={ {1, 2}, {0, 5}}; /*其余均为 0*/
```

- 按元素存储顺序给前面部分元素赋初值

```
int a4[2][3] = {1, 2, 3, 4}; /*其余均为 0*/
```

- 按元素存储顺序，给数组部分或全部元素赋初值，并且不指定第一维的元素个数

```
int a5[][3] = {1, 2, 3, 4, 5};/* 两行 */
```

- 用按行赋初值方法，对各行的部分或全部元素赋初值，并省略第一维的元素个数

```
int a6[][3] = {{0, 2}, {}};/*  两行 */
```

指针

- 有不同类型的指针变量，但在同一计算机系统中，所有指针变量值占用相同的存储空间
  - 在**32**位机器中，一个地址占**4byte**
- 除了形式指针变量，指针变量定义后，必须赋初始值地址后才能使用
  - int *p, *q, i, j;
  - i=10; q=&j;
  - *p=i; *q=10;
  - *p=NULL;*p=i;

```
f(int *i,int *j){
  int k;
  k=*i;
  *i=*j;
  *i=k;
}
```

数组与指针

- **数组名可以作为一个地址使用**，表示数组存储空间开始的地址（第一个数组元素地址），但**数据名不是一个指针变量**
  - **int a[10],*p1=a+1,*p2=a++,*p3=2*a;**
  - **int i, a[10],*p=a+3; /* p=&a[3];**
  - ***(p-2)=1; /* a[1]=1;**
  - **p[2]=2; /* a[5]=2;**
  - **p[-1]=-1; /* a[2]=-1;**
  - **i=p[2]**(p-2); /* i=2**

数组与指针

- int x[][4]={0,1,2,3,4,5,6,7,8,9};
  - 0,1,2,3
  - 4,5,6,7
  - 8,9,0,0
- int *p1=&x[1][1]; /* *p1=5 */
- int y=*p1**p1++; /* *p1*p1;p1++, p1=&x[1][2] */
- int z=*(p1+1)+p1[1]; /* *(p1+1)=p1[1]=x[1][3] */
- int w=*p1+p1[-2]; /* p1[-2]=x[1][0] */
- y=25;z=14;w=10;

字符数组与字符指针(1)

- 字符数组初始化
  - char s[]={‘F’, ‘u’, ‘d’, ‘a’, ‘n’};
  - char s[]={“Fudan”};
  - char s[]=“Fudan”;
  - char s1[5]=“Fudan”;
- 字符指针存储字符常量，指向字符串
  - char *cp1, *cp2=“Fudan”;
  - cp1=“Fudan”; *cp1=“Fudan”;

字符数组与字符指针(2)

- 字符数组存储字符串，可以整体输出
  - char s[]="Fudan";
  - printf("%s",s);
- 而仅仅存储字符的字符数组不能整体输出
- 字符数组可以直接输入字符串
  - char s[10];
  - scanf("%s",s); /* 输入的字符串不能超过9个有效字符

字符数组与字符指针(3)

- char str1[10] = "fudan";
- char str2[20] = "fudan\0university";
- char str3[20] = "fudan\01university";
- char str4[20] = "fudan\012university";
- char str5[20] = "fudan\0123university";
- char str6[20] = "fudan\x123university";
- int i1, j1,i2,j2,i3,j3,i4,j4,i5,j5,i6,j6;
- i1=sizeof(str1) ; i2=sizeof(str2) ; i3=sizeof(str3) ; i2=sizeof(str4) ; i2=sizeof(str5) ; i2=sizeof(str6);
- j1=strlen(str1) ; j2=strlen(str2) ; j3=strlen(str3) ; j4=strlen(str4) ; j5=strlen(str5) ; j6=strlen(str6);
- printf("%d,%d , %d,%d , %d,%d , %d,%d , %d,%d , %d,%d", i1,j1 , i2,j2 , i3,j3 , i4,j4 , i5,j5 , i6,j6);

函数与函数调用

- 函数不可以嵌套定义，但可以递归调用
- 调用函数时，只能把实参的值传递给形参，形参的值不能传递给实参
- 例子：int i, f(int x, int y);
- 调用
  - i=f(3, 4);
  - i=f(2+1, 2*2);
  - i=f((4, 3), (3,4));

文本文件与二进制文件

- 假设在某一计算机中，int用4个bytes表示
- int i=100, j=-100
  - 那么，i和j在二进制文件中，只占4bytes
  - 而在文本文件中，则分别占3bytes, 4bytes
- 假设在某一计算机中，float用4个bytes表示
- float f1=100.2345, f2=-1.115
  - 那么，f1和f2在二进制文件中，只占4bytes
  - 而在文本文件中，则分别占8bytes, 6bytes

数组元素排序

- **冒泡排序**
  - **每一轮要相邻数据两两比较并交换位置。其实可以不用每次交换位置，每一轮只要把最大或最小的挑出来即可**
  - **假设有int x[M]，M是一个正整数，要对其中的元素从小到大排序**

```
int i, j, min, temp;
for(i=0;i<M-1;i++){
 for(j=i+1,min=i;j<M;j++)
  if(x[j]<x[i])min=j;
  if(i!=min){
   temp=x[i]; x[i]=x[min]; x[min]=x[i];
  }
}
```

从一个字符串中找一个子串

```
int findsubstring(char *str, char *substr){
 int i, j; char *s, *t;
  if (str==null || substr==null) return -1;
  for(i=0;str[i];i++){
    s=str+i, t=substr;
    for(;*s==*t&&t!=null&&s!=null;s++,t++);
    if(!t)return i;
   }
   return 0;
}
```

递归函数：求4!

```
long fac (int n ) {
  if ( n == 0 )  return 1L;
  else
   return (long)n*fac (n-1);
}
```

递归函数举例

```
#include  <stdio.h>
void fun(int *x, int i, int j)
{
        if(j <= i)
                fun(x, i-3, 2+j);
        *x= *x+2*j;
}
void main()
{
        int x=1;
        fun(&x, 8, 2);
        printf("%d\n", x);
}
```

1: x=1
2: fun(&x,8,2)
7: x=21+2*2=25

3: fun(&x,5,4)
6: x=13+2*4=21

4: fun(&x,2,6)
5: x=1+2*6=13

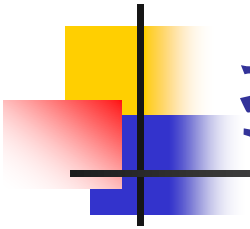
倒序输出一个正整数

```
int x, i, j

for (j=0;x>0; x/=10){
   i=x%10;
   j++;
    printf("%c", '0'+i);
}
```

顺序输出一个正整数

```
void show-int(int n)
{ char c;
   c = n % 10 + '0';
   if (n >= 10) show-int(n/10);
   printf("%c", c);
}
```



指针/地址值

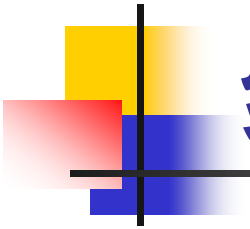
- 如果一个计算机是32位，则它的地址空间大小为2^32。每个地址用32位，即4 bytes表示
  - int x, *y, *z[10];
  - size of y =4; size of z =40;

结构

```
struct  pNode {
   float   pos[2];
   struct  pNode  * next;
} p
```

size of p =?

链表操作要熟练：
1）构建链表；
2）搜索；
3）插入、删除表元
4）链表合并等



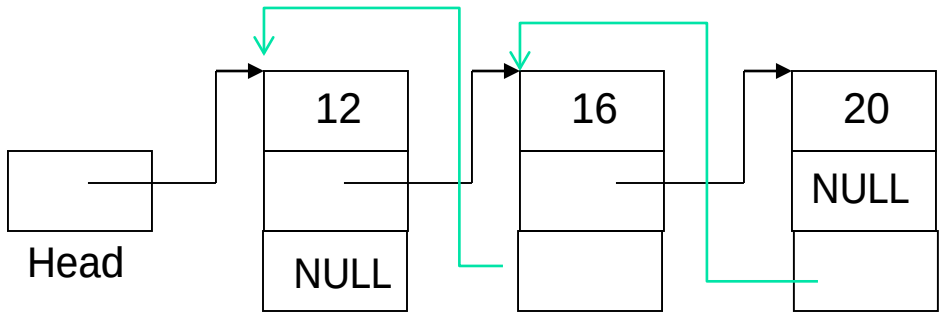
结构

- struct person{
- struct {
- char *first-name;
- char *last-name;
- }name
- int age;
- }p1,*p2

- P1.name.first-name=“Tom”; p1.age=18;
- P2->name.first-name=“Jerry”, p2->age=18;

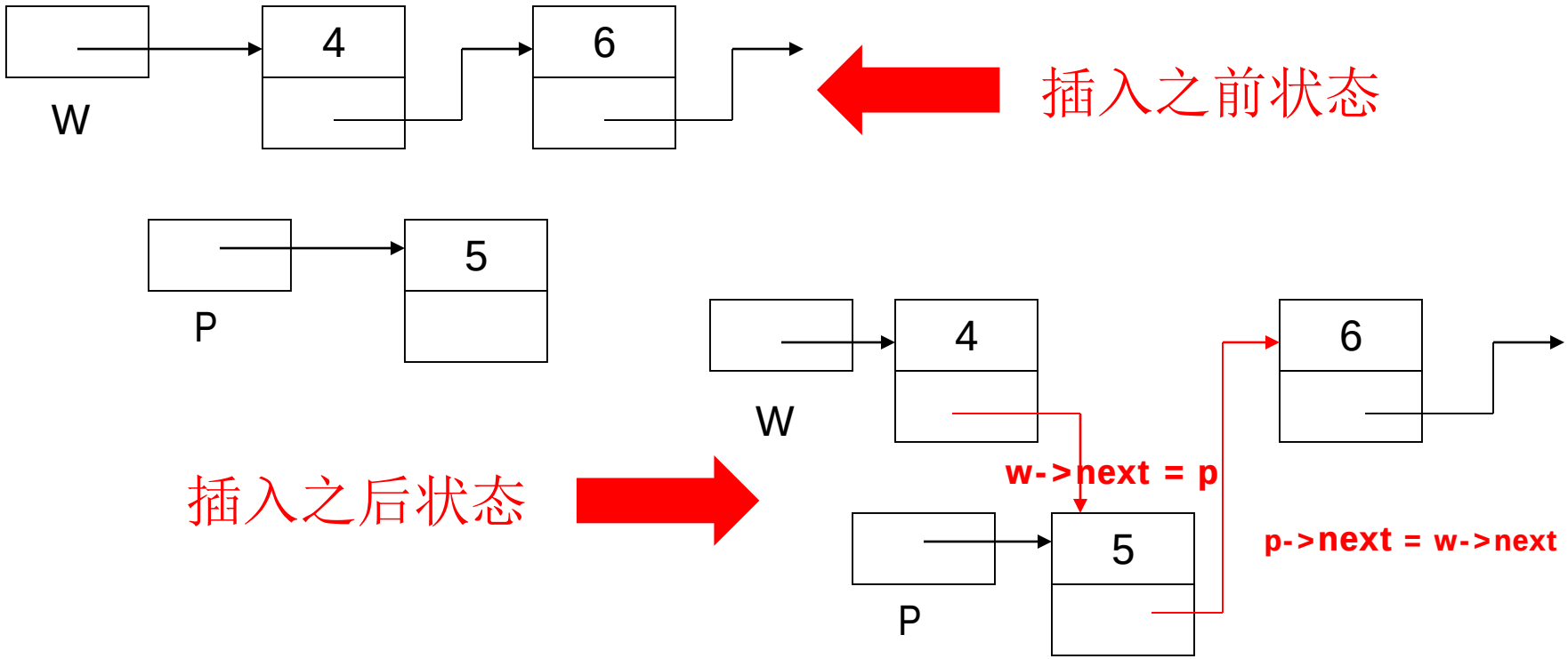
双向链表

```
struct pNode {
    int key;
    struct pNode * next;
    struct pNode * last;
} p
```

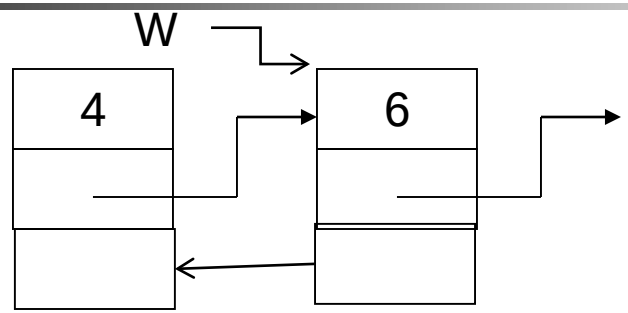


在单向链表中插入表元

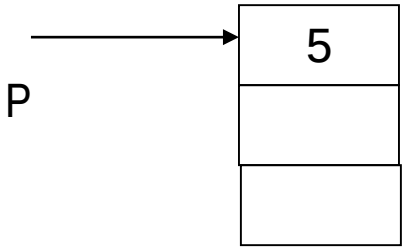
- 向链表插入一个新表元(续)



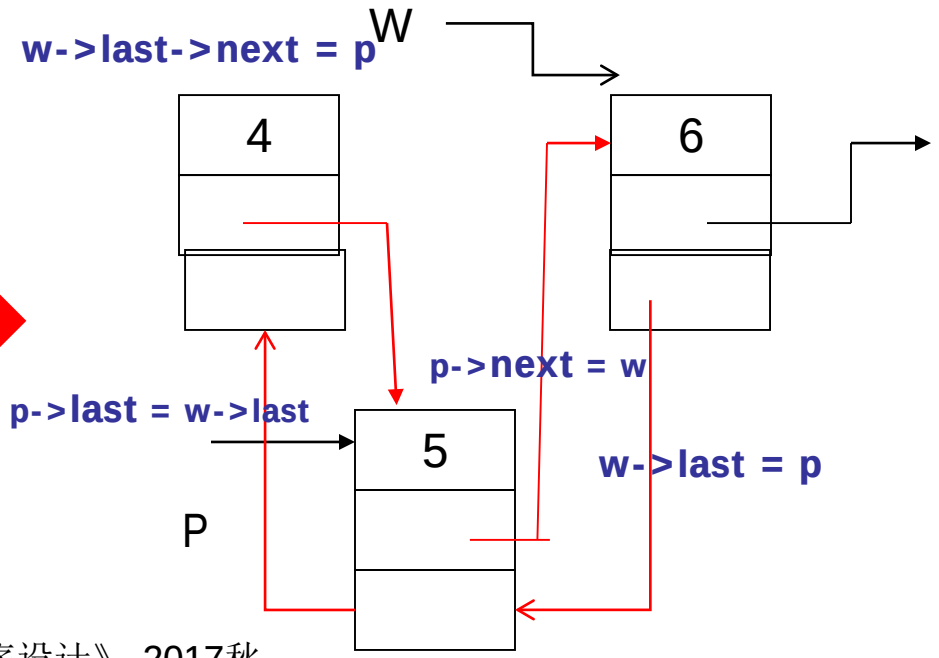
在双向链表中插入表元



插入之前状态



插入之后状态



课本中的几个算法

- 冒泡排序，有三个版本
  - 仔细看课件
- 二分搜索算法
  - 课件中也有
- 求两个整数的最大公因数