知识点1【作业讲解】

知识点2【for循环】

循环嵌套循环（了解）

作业：

1、求出1~100中能被7整除的数。

知识点3【while 循环、do...while】

while

do....while();

goto 跳转

作业：重复输入1~7的数值判断 星期几，注意输入0，就结束程序，如果是0或1~7以外的数，请提示“请重新输入有效数值”。

总结：for while 我们如何选择呢?

知识点4【数组的概述】

知识点5【一维数值数组】

1、数组的定义

案例：遍历数组：

案例：数组的初始化

全部初始化

部分初始化

扩展：初始化

数组的空间大小：（重要）

数组元素的操作

案例：定义数组 5个元素 每个元素int类型 获取键盘输入

作业：定义数组 5个元素 每个元素int类型 获取键盘输入 请求出数组的最大值、最小值、平均值。

知识点6【二维数组】

二维数组的定义

二维数组的遍历：

二维数组的初始化

分段初始化：用{}里面的{}明确的表示一行

连续初始化:放满一行 才能放下一行

作业：定义int arr[3][4]的二维数组，获取键盘输入，并求出每一行的平均值。

知识点1【作业讲解】

作业1：

```
void test01()
{
 char ch =0x85 ;//1000 0101>>4
 //逻辑右移: 1000 0101>>4 == 0000 1000 == 0x08
 //算术右移: 1000 0101>>4 == 1111 1000 == 0xf8
 ch = ch>>4;
 printf("ch = %#x\n",ch);//0xf8是算术右移

 //逻辑右移、算术右移是编译器决定 我们只能测试 不能改变
}
int main(int argc,char *argv[])
{
 test01();
 return 0;
}
```

作业：

```
void test02()
{
 //需求: 请输入1~7的数 判断是星期几
 int date = 0;
 printf("请输入1~7的数:");
```

```
6   scanf("%d", &date);
7
8   switch(date)
9   {
10   case 1:
11   printf("星期一\n");
12   break;
13   case 2:
14   printf("星期二\n");
15   break;
16   case 3:
17   printf("星期三\n");
18   break;
19   case 4:
20   printf("星期四\n");
21   break;
22   case 5:
23   printf("星期五\n");
24   break;
25   case 6:
26   printf("星期六\n");
27   break;
28   case 7:
29   printf("星期日\n");
30   break;
31   default:
32   printf("请输入有效数值\n");
33   break;
34
35   }
36
37  }
38  int main(int argc,char *argv[])
39  {
40   test02();
41   return 0;
42  }
```

知识点2【for循环】

```
for( 初始语句; 循环条件 ; 步进条件)
{
 //循环语句
}
//初始语句:只在循环开始时 执行一次
//循环条件:每次循环都要执行 如果循环条件为真 进入循环体 如果为假 退出循环
//步进条件:每次循环结束的时候 要执行的语句
```

案例：求1~100的和

```
void test03()
{
 //1+2+3+4+ ...+ 100
 int i=0;
 int sum = 0;

 for(i=1 ; i<=100 ; i++)//i++ ==> i=i+1
 {
 sum = sum+i;
  }

  printf("sum = %d\n",sum);//5050

 }
```

案例：

```
void test03()
{
 //1+2+3+4+ ...+ 100
 int i=1;
 int sum = 0;

 //如果变量 提前初始化了 for的初始化语句 可以省略
 for( ; ; )//建议别这么做
 {
  //如果for省略循环条件 必须在循环语句中实现 退出循环的条件
  if(i>100)
  break;//跳出循环

  sum = sum+i;
  i++;//如果for省略步进条件 必须在循环语句中实现 步进动作
  }
  printf("sum = %d\n",sum);//5050
```

```
18
19 }
20 int main(int argc,char *argv[])
21 {
22  test03();
23  return 0;
24 }
```

案例：

```
1 void test04()
2 {
3  for(;;)
4  {
5  for(;;)
6  {
7  break;//只能跳出离它最近的一层循环
8  }
9  }
10 }
```

案例：

```
1 void test05()
2 {
3  int i=0;
4  int sum = 0;
5  for(i=1 ;i<=100 ; i++)
6  {
7  //sum = sum+i;
8  if(i == 50)
9  continue;//结束本次循环 立即从当前位置直接 进入下一次循环
10
11  sum += i;
12  }
13
14  printf("sum = %d\n",sum);//5000 仅仅是少加了一个50
15 }
```

案例：求出0~100的所有偶数的和

```
1 void test06()
2 {
3  //求0 2 4 6 8......100的和
4  int i=0;
5  int sum = 0;
```

```
 for(; i<=100 ; i=i+2)//步进条件不是单纯的 ++i
 {
 sum += i;
 }

  printf("sum = %d\n",sum);//2550
}
```

循环嵌套循环（了解）

```
void test07()
{
 int i=0;
 int j=0;

 //总结 外层循环的次数 * 内层循环的次数 == 总循环次数
 for(i=0;i<10;i++)//0~9 =10
 {
 for(j=0;j<10; j++)//0~9 =10
  {
  printf("i=%d, j=%d\n", i,j);//循环100次
  }
  }
}
```

作业：

1、求出1~100中能被7整除的数。

知识点3【while 循环、do...while】

while

```
while(循环条件)
{
 //循环语句
}
//如果“循环条件”为真 就进入循环体执行循环语句
```

注意：

1、while没有初始化语句 用户提前初始化好。

2、while没有步进语句 用户必须在循环语句中 写好 步进语句。

案例：

```c
void test08()
{
 int i=1;//提前初始化
 int sum = 0;//局部变量如果不初始化 内容不确定

 while(i<=100)
 {
 sum += i;
 i++;//不仅条件
  }
  printf("sum = %d\n", sum);//5050
 }
```

do....while();

```c
do
{
 //循环语句;
}while(循环条件);
```

先执行循环语句 再判断循环条件是否为真 如果为真 进行下次循环 如果为假 直接退出循环

```c
void test09()
{
 int num = 0;
 do
 {
 printf("ok\n");
 }while(num>0);
}
```

goto 跳转

```c
void test10()
{
 printf("-------001-------\n");
 printf("-------002-------\n");
 goto here;
 printf("-------003-------\n");
 printf("-------004-------\n");
 printf("-------005-------\n");
here:
  printf("-------006-------\n");
  printf("-------007-------\n");

```

```
13  }
```

运行结果：

```
--------001--------
--------002--------
--------006--------
--------007--------
```

作业：重复输入1~7的数值判断 星期几，注意输入0，就结束程序，如果是0或1~7以外的数，请提示“请重新输入有效数值”。

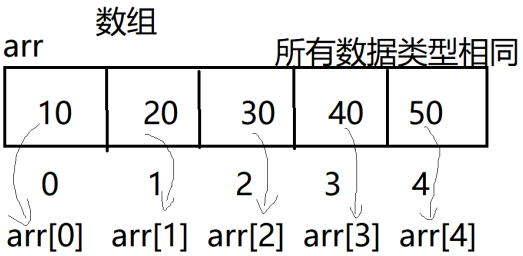
总结：for　while 我们如何选择呢?

如果循环的次数是确定的 建议选择for。

如果循环的次数不确定，知道退出的条件　建议选择while.

知识点4【数组的概述】

```
int num1 = 10;
int a = 20;
int hehe = 30;
int heihe = 40;
int lala = 50;
```



具有相同类型的变量 用连续的空间存储起来 这样的数据结构 就是数组。

知识点5【一维数值数组】

1、数组的定义

需求：请定义一个数组 该数组 有10个元素 每个元素为int

在定义的时候：

a、arr[]　arr和[]结合是数组

b、将确定的元素的个数放入 []中

c、用元素的类型 定义一个普通变量

d、从上往下整体替换。

int arr[10];

注意：

1、数组名 arr 不能和其他变量名 同名

2、数组的元素小标是从0开始：0~9

3、数组的元素 分别是：arr[0]、arr[1]~arr[9], 如果访问arr[10]数组越界。

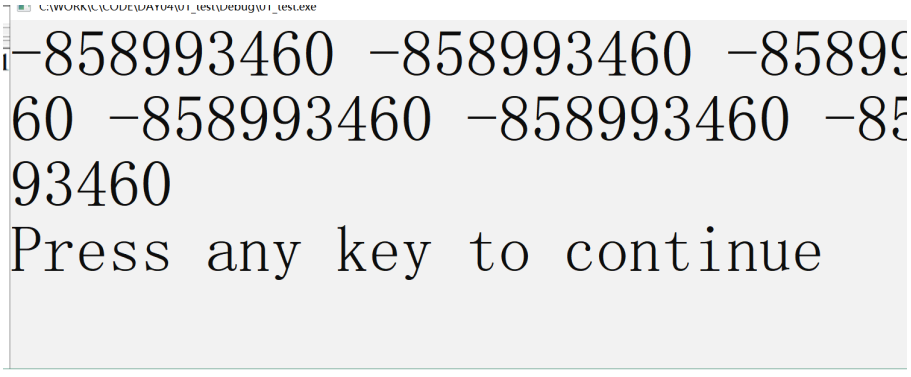
4、数组的元素 等价于 普通变量

5、在定义数组的时候 ,[]里面的值 不能是变量（c89）。

案例：遍历数组：

```
void test01()
{
 //局部数组 如果不初始化 内容不确定
 int arr[10];
 int i=0;

 //遍历数组
 for(i=0;i<10; i++)
 {
  printf("%d ",arr[i]);//输出的都是不确定的值
  }
  printf("\n");
}
```

运行结果：



案例：数组的初始化

初始化：定义的时候 给变量 或数组元素 赋值的动作 叫初始化。

全部初始化

```
int arr[5] = {10,20,30,40,50};
//如果是 全部初始化 数组元素的个数 可以省略 实际的元素个数由 初始化个数决定
```

```
int arr[] = {10,20,30,40,50};//少用
//错误显示
int arr[];//错误无法确定数组的大小
```

部分初始化

```
//部分初始化 未被初始化 部分自动补0 (重要)
int arr[5] = {10,20,30};

//初始化数组 常见的操作（将数组的所有元素清零）
int arr[5] = {0};//只初始化arr[0]=0,未被初始化部分 自动补0
int arr[5] = {2};//2 0 0 0 0
```

扩展：初始化

```
int arr[5]={[2]=3, [4]=7};//0 0 3 0 7
//[2]=3将数组的第2个元素 初始化为3
//[4]=7将数组的第4个元素 初始化为7
```

数组的空间大小：（重要）

```
//arr数组名 作为数组类型 代表的是数组空间的总大小
int arr[5] = {10,20,30,40,50};
int n = 0;

//数组的总大小 = 元素的个数 * 每个元素的大小
printf("数组的总大小=%d\n",sizeof(arr));//20

//数组元素的大小 arr[0]是数组的第0个元素
printf("数组元素的大小 = %d\n",sizeof(arr[0]));//4

 //数组元素的个数 = 数组总大小/每个元素的大小
 n = sizeof(arr)/sizeof(arr[0]);
 printf("数组元素的个数为:%d\n", n );
```

案例：

```
void test05()
{
 int arr[] = {10,20,30,40,50,1,2,10};
 int n = sizeof(arr)/sizeof(arr[0]);//是求数组的元素个数
 int i=0;

 //遍历数组
 for(i=0;i<n; i++)
 {
```

```
 printf("%d ",arr[i]);//输出的都是不确定的值
 }
 printf("\n");
}
int main(int argc,char *argv[])
{
 test05();
 return 0;
}
```

运行结果：

```
10 20 30 40 50 1 2 10
Press any key to continue
```

案例：

```
void test06()
{
 int arr1[5]={1,2,3};
 int arr2[]={1,2,3};
 char arr3[5]={1,2,3};
 int arr4[]={[5]=7};

 printf("%d\n",sizeof(arr1));//20
 printf("%d\n",sizeof(arr2));//12
 printf("%d\n",sizeof(arr3));//5
 printf("%d\n",sizeof(arr4));//24 //vc++6.0不支持
}
int main(int argc,char *argv[])
{
 test06();
 return 0;
}
```

数组元素的操作

数组的元素 等价于 普通变量

```
void test07()
{
 int arr[5]={1,2,3,4,5};
```

```
 int n = sizeof(arr)/sizeof(arr[0]);
 int i=0;

 //给数组元素 赋值
 arr[0] = 100;

  //num++
  arr[1]++;//arr[1]=arr[1]+1;

  //scanf("%d",&num)
  scanf("%d",&arr[2]);

  for(i=0;i<n;i++)
  {
  printf("%d ",arr[i]);
  }
  printf("\n");
}
```

案例：定义数组 5个元素 每个元素int类型 获取键盘输入

```
void test08()
{
 //数值数组 只能逐个元素操作 不能整体操作
 int arr[5]={0};
 int n = sizeof(arr)/sizeof(arr[0]);
 int i=0;

 printf("请输入%d个int数据\n", n);
 for(i=0;i<n;i++)
  {
  scanf("%d",&arr[i]);//&arr[i] 表示取第i个元素的地址
  }

  for(i=0;i<n;i++)
  {
  printf("%d ", arr[i]);//arr[i] 访问数组的第i个元素
  }
}
```

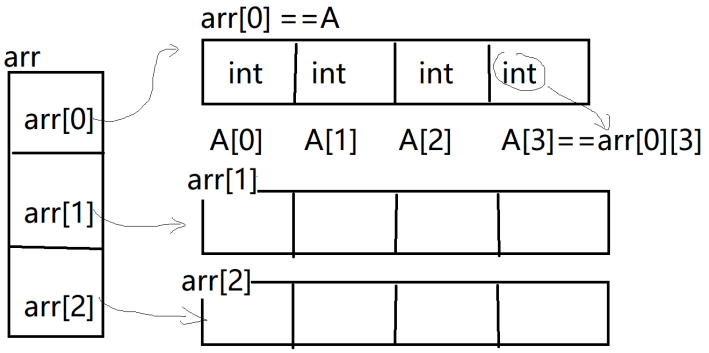
作业：定义数组 5个元素 每个元素int类型 获取键盘输入 请求出数组的最大值、最小值、平均值。

知识点6【二维数组】

二维数组的定义

请定义一个数组 该数组有3个元素
每个元素 又是一个数组，此数组
有4个元素 每个元素为int类型

int arr[3][4]; //数组的数组




```
int arr[3][4];
//第一个[]里面的值 表示行数 ， 第二个[]里面的值 表示列数
//arr数组有3行4列
//3行：0~2行 4列：0~3列
```

arr	0	1	2	3
0	arr[0][0]	arr[0][1]	arr[0][2]	arr[0][3]
1	arr[1][0]	arr[1][1]	arr[1][2]	arr[1][3]
2	arr[2][0]	arr[2][1]	arr[2][2]	arr[2][3]

二维数组的遍历：

```
int i=0;
for(i=0;i<3;i++)
{
 //遍历第i行
 int j = 0;
 for(j=0;j<4; j++)
 {
 printf("%d ",arr[i][j])
 }
}
```

案例：

```
void test01()
{
 int arr[3][4];//局部数组不初始化元素内容不确定
 int i=0,j=0;

 for(i=0;i<3;i++)//行的变化
 {
 for(j=0;j<4;j++)//列的变化
 {
  printf("%d ",arr[i][j]);
  }
  printf("\n");
  }
}
```

运行结果：

```
"C:\WORK\C\CODE\DAY04\03_test\Debug\03_test.exe"
-858993460 -858993460 -858993460 -858993460
-858993460 -858993460 -858993460 -858993460
-858993460 -858993460 -858993460 -858993460
Press any key to continue
```

二维数组的初始化

分段初始化：用{}里面的{}明确的表示一行

```
int arr[3][4]={ {1,2,3,4}, {5,6,7,8}, {9,10,11,12} };
```

连续初始化:放满一行 才能放下一行

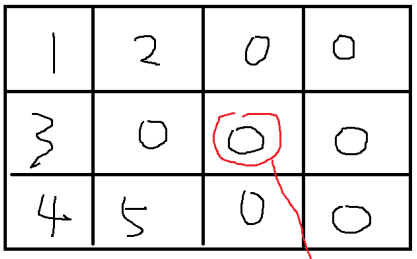
```
int arr[3][4]={1,2,3,4,5,6,7,8,9,10,11,12};
```

案例：

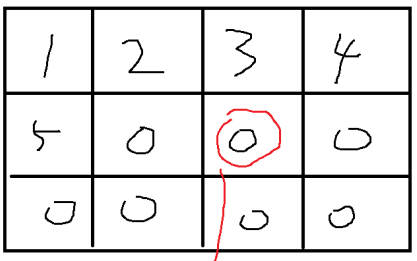
```
void test01()
{
 int arr1[3][4]={{1,2},{3},{4,5}};
 int arr2[3][4]={1,2,3,4,5};

 printf("%d\n", arr1[1][2]+arr2[1][2]);
}
int main(int argc,char *argv[])
{
```

```
10    test01();
11    return 0;
12  }
```

int arr1[3][4]={{1,2},{3},{4,5}};



int arr2[3][4]={1,2,3,4,5};



printf("%d\n", arr1[1][2]+arr2[1][2]); = 0

作业：定义int arr[3][4]的二维数组，获取键盘输入，并求出每一行的平均值。