



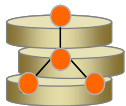
第六单元（4）：变量的作用域与存储类别

教学内容	变量的作用域和存储类别，内部函数和外部函数，工程文件
教学目标	
应知	变量的作用域与生存期
	变量的存储类别
	内部函数、外部函数的使用与工程文件
应会	程序中变量的作用范围
	能够分析程序中静态变量的值
	学会建立工程文件进行函数调用
难点	静态变量在程序中的作用
	单步断点调试方法的使用

教学方法

- 讨论式教学，分析方法采用单步断点调试，通过观察程序运行和程序分析学习全局变量和局部变量，静态存储和动态存储

教学过程



1. 全局变量和局部变量

- 分析结果说明全局变量与局部变量的作用域

```
int a,b;
main()
{ int i,j;
// extern c;    //外部变量
void deel(); void my();
clrscr();
a=3;
b=5;
printf("a=%d,b=%d",a,b);
deel();
my();
}
int c=200;
void deel()
{ int a,b;
a=100;
b=100;
```



```
printf("\ndeel function  a:%d,b:%d,c:%d",a,b,c);
}
void my()
{ printf("\nmy function\n%d  %d  %d",a,b,c);
}
```

2. 静态存储和动态存储

■ 分析结果说明静态存储的特点

```
(1) f(int a)
{ auto int b=0;
static c=3;
b=b+1;
c=c+1;
return(a+b+c);
}
main()
{int a=2,i; clrscr();
for(i=0;i<3;i++)
    printf("%5d",f(a));
}
```

结果为 7 8 9

```
(2) int fac(int n)
{static int f=1;
f=f*n;
return f;}
main()
{ int i;
for(i=1;i<=5;i++)
    printf("%d! = %d\n",i,fac(i)); }
```

3. 定义一个二维字符数组 `string[3][10]`，分别在 `prog1.c` 中和 `prog2.c` 中定义函数打印出二维字符数组的第 1 和第 2 个字符串。(外部函数、内部函数、工程概念)

4. 分析程序结果：

```
(1)
main()
{int k=4,m=1,p;clrscr();
p=func(k,m);
printf("%3d",p);}
```



```
p=func(k,m);
printf("%3d\n",p);
}

func(int a,int b)
{static int m=0,i=2;
 i+=m+1;
 m=i+a+b;
 return m;
}

(2).
void func(int);
main()
{int k=4;
 func(k);
 func(k);
}
void func(int a)
{ static int m=0;
 m+=a;
 printf("%3d",m);
}
```



学生容易出错的地方

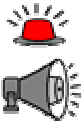
- 静态变量在程序中的值
- 全局变量与局部变量同名时起作用的那个变量

问题与讨论

- 在什么情况下用静态变量?
- 什么是全局变量(外部变量)
- 什么是局部变量(内部变量),与全局变量有什么区别
- 什么是外部函数
- 什么是内部函数,与外部函数有什么区别
- 什么是静态变量
- 什么是动态变量,与静态变量有什么区别



小结（可由问题与讨论方式给出）



- 变量从作用域可分为外部变量和内部变量，即全局变量和局部变量
- 变量从生存期的角度可分为静态变量和动态变量
- 函数也有内部函数与外部函数，访问时有所不同



课后任务

- 编写函数 `int prime(int n)` 判断 `n` 是否是素数（只能被 1 和该数自己整除的数），将判断结果返回，在主函数中调用函数 `prime()`，求出 1—100 之内的所有素数。