# 第八章 函数

本章学习目标：

* 理解生命周期和作用域的概念
* 了解存储类型说明符，掌握extern、static的用法
* 了解类型限定符的作用
* 理解程序的内存布局，掌握动态内存分配相关系统函数的用法
* 理解函数指针的作用，掌握函数指针的用法。

## 8.1 实践题

**一、生存周期和作用域**

**实验目的**

1. 理解生存周期的概念。
2. 掌握变量的作用域。

**实验步骤**

步骤1：建立一个VS2012工程，在工程里添加一个源文件main.c；

步骤2：在main.c里输入如下内容：

|  |
| --- |
| 1 #include <stdio.h>  2 #include <stdlib.h>  3  4 int x = 3;  5 void varScope1();  6 void varScope2();  7 int main(void)  8 {  9 int y = 11;  10 varScope1();  11 varScope2();  12 printf("x = %d y = %d\n",x,y);  13 {  14 int x = 30,y = 35;  15 printf("x = %d,y = %d\n",x,y);  16 }  17 varScope1();  18 varScope2();  19 printf("x= %d\n",x);  20  21 system("pause");  22 return 0;  23 }  24  25 int y = 5;  26  27 void varScope1()  28 {  29 int x = 20;  30 printf("x=%d,y=%d \n",x,y);  31 x++,y++;  32 printf("x=%d,y=%d \n",x,y);  33 }  34  35 void varScope2()  36 {  37 static int x = 40;  38 printf("x=%d,y=%d \n",x,y);  39 x++,y++;  40 printf("x=%d,y=%d \n",x,y);  41 } |

步骤3：编译运行，结果如图1所示：

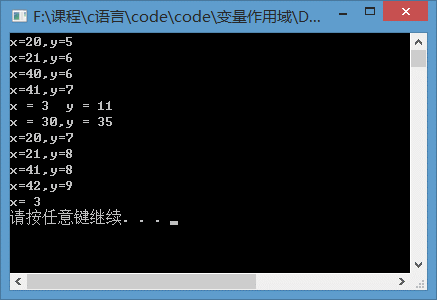


图1

请根据运行结果填写下表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Printf行号 | 输出 | 输出使用的X和y的行号 | X和y作用域 | X和y的生存周期 |
| 30 | x=20;y=5 | x: 29 y: 25 | x:从29~33行  y:从25~文件结束 | X从varScope1运行开始到varScope1运行结束；y从程序运行开始到程序结束 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**实验结果/结论**

1. **实验结论**

* 复合语句内定义的是局部变量，只能在复合语句里使用；
* 局部变量的作用域是从定义开始到定义该局部变量的块结束；
* 全局变量的作用域是从定义到本源文件结束，可以使用extern将全局变量进行前向声明，那么全局变量的作用域就是从声明到本源文件结束。
* 局部变量的生存周期是从函数执行开始，到函数执行结束。
* 函数的作用域和全局变量类似，从定义开始到源文件结束，但可通过函数声明，扩大作用域。

**二、 类型限定符**

**实验目的**

1. 掌握static和extern的用法
2. 掌握多源文件的划分和引用

**实验步骤**

步骤1：建立一个VS2012工程，在工程里添加一个源文件main.c；

步骤2：在main.c里输入如下内容：

1 #include <stdio.h>

2 #include <stdlib.h>

3 #include <math.h>

4 int gcb(int a,int b);//最大公约数

5 int lcm(int a,int b);//最小公倍数

6 int main(void)

7 {

8

9 extern int x,y;

10

11 printf("%d和%d的最大公约数为：%d\n",x,y,gcb(x,y));

12 printf("%d和%d的最小公倍数为：%d\n",x,y,lcm(x,y));

13

14 system("pause");

15 return 0;

16 }

17

18 int x = 12,y = 32;

19

20 int lcm(int a,int b)

21 {

22 return a \* b/gcb(a,b);

23 }

24 int gcb(int a,int b)

25 {

26 a = abs(a);

27 b = abs(b);

28 while(b)

29 {

30 int tmp = a%b;

31 a = b;

32 b = tmp;

33 }

34 return a;

35 }

步骤3：编译运行，结果如图2所示：

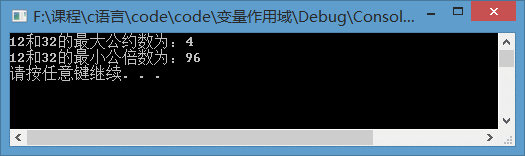


图2

步骤4：将第9行注释，编译运行，错误如图3所示。

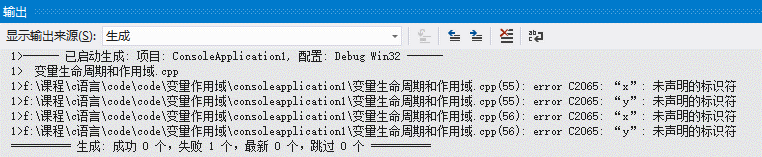


图3

请给出错误的理由：

步骤5：取消注释，并新建一个头文件(factor.h)和一个源文件(factor.c)

|  |  |
| --- | --- |
| factor.h | Factor.c |
| 1 #ifndef \_FACTOR\_H  2 #define \_FACTOR\_H  3  4 //最大公约数  5 int gcb(int a,int b);  6  7 //最小公倍数  8 int lcm(int a,int b);  9  10 #endif  11 extern int x,y; | 1 #include <math.h>  2 #include "factor.h"  3  4 int lcm(int a,int b)  5 {  6 return a \* b/gcb(a,b);  7 }  8 int gcb(int a,int b)  9 {  10 a = abs(a);  11 b = abs(b);  12 while(b)  13 {  14 int tmp = a%b;  15 a = b;  16 b = tmp;  17 }  18 return a;  19 }  20 int x = 12,y = 32; |

步骤6：修改maic.c内容。

|  |
| --- |
| main.c |
|  |
| 1 #include <stdlib.h>  2 #include "factor.h"  3  4 int main(void)  5 {  6 printf("%d和%d的最大公约数为：%d\n",x,y,gcb(x,y));  7 printf("%d和%d的最小公倍数为：%d\n",x,y,lcm(x,y));  8  9 system("pause");  10 return 0;  11 } |

步骤7：编译运行结果同步骤3一样。虽然x和y定义在factor.c中，但在factor.h中使用extern声明了变量x和y，则在其他文件中可以引用全局变量x和y。

步骤8：如果将fact.c中第20行，x和y的定义前加上static：

static int x = 12,y = 32;则连接出错，如图4所示。

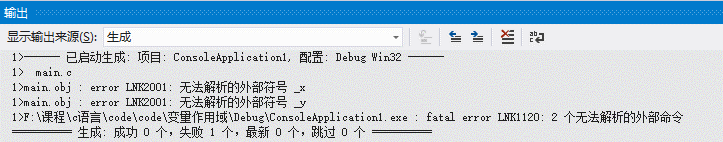


图4

static会将全局变量或函数的作用域限定在文件内，其他文件无法引用。你可以将lcm函数的返回值前加上static，同样会报出连接错误。

**实验结果/结论**

1. **实验原因**

* 步骤8中出现连接错误的原因是，存储类型说明符static将全局变量的作用域限制在本源文件里，不能导出到其他源文件里使用，所以连接器在连接factor.c的目标文件和main.c的目标文件时，不能从factor.c的目标文件导出所需要的全局变量名，导致了连接错误。如果没有加static，全局变量的默认存储类型说明符为extern，可以将一个源文件里的全局变量导出到其他源文件使用，所以我们通过引入头文件factor.h，可以在main函数里使用factor.c中定义的全局变量。
* 给函数返回值类型加上存储类型说明符static后，该函数只能在本源文件使用，不能导出到其他文件使用。否则函数的存储类型说明符为extern，可以导出到其他源文件使用。

2. **实验结论**

* 存储类型说明符static除了可以声明静态局部变量外，还可以将全局变量的作用域限制到本源文件，使它无法在其他源文件里使用。
* extern可以对全局变量进行声明，在一个源文件里扩大全局变量的作用域；并且可以将全局变量导出到其他源文件使用。
* static和extern可以应用到函数的类型声明，效果同全局变量。
* 划分多源文件可以提高程序的结构化，使代码结构清楚，易维护，复用。
* 划分多源文件时，可以把功能相近的一组函数放到同一个源文件里实现，然后在头文件里声明，其他源文件可以通过加载该头文件使用相应函数。

## 11.2 理论题

### A类

**一、填空题**

**二、选择题**

1.下面说法正确的是（）。

A. 局部变量的有效范围从定义处到文件结束

B. 在函数体外定义的变量一定不是局部变量

C. 局部变量可以是动态的也可以是静态的

D. 局部变量只能是在函数体内定义的变量

2. 下面说法正确的是（）。

A. 全局变量可以是动态的也可以是静态的

B. 形式参数是局部变量

C. 在不同函数中可以使用相同名字的变量

D. 在函数内定义的变量只在本函数范围内有效

3. 在一个Ｃ源程序文件中，若要定义一个只允许本源文件中所有函数使用的全局变量，则该变量需要使用的存储类别是（　　　　　）

A.extern B.register C.auto D.static E. restrict

4.下面有关typedef说法正确的是（ ）

A．和#define一样定义了一个常量；

B．typedef也属于预处理；

C．typedef创造了一个新的类型；

D．typedef int a[10];中的a是数组类型。

5. 以下叙述中，错误的是\_\_\_\_\_\_。

A、不同函数中可以使用相同名字的变量

B、在函数外部定义的变量是全局变量

C、形式参数是局部变量

D、在main函数体内定义的变量是全局变量

**三、综合题**

1.任意输入一个正整数，输出一个有该整数各位数组组成的最大数。函数原型：

int maxNum(int num) ;

2. 在西方民俗星象学上星期五和数字13都代表着坏运气，两个不幸的日期如果重叠被认为是很不好的一天。只要任何一月十三号又恰逢是星期五，这个日期就被称作“黑色星期五”。对于给定年份，编写一个函数，判断该年份是否有黑色星期五，如果有则输出日期

3.编写一个递归函数，计算满足下述定义的整数序列的第n项。

函数的原型为：int findn(int n);

### B类

1. **填空题**
2. 已知有程序段，请在横线处补齐代码，并给出输出结果。

#include <stdio.h>

;

int main(void){

int a= 2 ;

int i ;

printf("%d ",a+x) ;

for(i = 0,x = 1 ;i<5 ;i++){

int x = i + 1 ;

}

printf("%d ",x) ;

return 0 ;

}

int x = 10 ;

输出结果为：

1. 请使用typedef对int (\*a[10])(int)进行简化：
2. 请阅读下面代码段，补齐代码。

void f1(){

....

}

void f2(){

....

}

void f3(){

....

}

void menu(){

= {f1,f2,f3} ;

int choice ;

do{

printf("1. Choice1") ;

printf("2. Choice2") ;

printf("3. Choice3") ;

printf("0. exit") ;

printf("请输入你的选择：") ;

scnaf("%d",&choice) ;

if(choice>0&&choice<=3){

}

else if(0==choce){

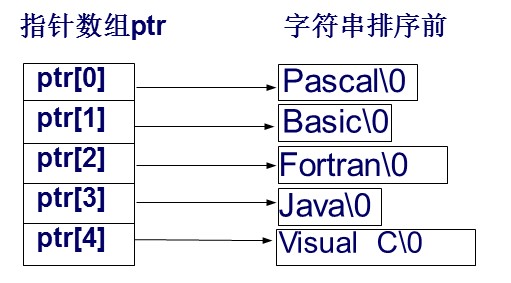
break ;

}

}while(1) ;

}

4. 下面这段代码要完成字符串排序功能，指针数组ptr必须动态生成，请补齐代码。

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

void sort(char\*ptr,int len){

for (i=0; i<len-1; i++){

for (j = 0; j<len-1-i; j++){

if ( ) {

;

;

;

}

}

}

}

int main(void){

const int len = 5 ;

int i = 0 ;

char tmp[100] ;

char\* p ;

char \*ptr = malloc(len \* sizeof( )) ;

if(!ptr){

exit(0) ;

}

while(i<len){

printf("请输入一个字符串：") ;

scanf("%s",tmp) ;

p = malloc( ) ;

if(!p){

exit(0) ;

}

;

;

ptr[ ] = p ;

}

}

**二、选择题**

1. 以下程序的正确运行结果是（　　　　　）

int main(void)

{

int a=2,i;

for(i=0;i<3;i++)

printf(“%4d”,f(a));

}

int (int a)

{

int b=0; static int c=3;

b++; c++;

return(a+b+c);

}

A.7 7 7 B.7 10 13 C.7 9 11 D.7 8 9

2. 请阅读下面的代码段，给出程序的运行的结果（ ）

#include <stdio.h>

void num()

{

extern int x, y;

int a = 15; b = 10;

x = a - b;

y = a + b;

}

int x, y;

int main(void)

{

int a = 7, b = 5;

int x,y ;

x = a + b;

y = a - b;

num();

printf("%d, %d\n", x, y);

}

A、12,2 B、不确定 C、5,25 D、1,12

3. 若有以下程序

#include

void f（int n）;

main（）

{

void f（int n）;

f（5）;

}

void f（int n）

{ printf（"%d＼n",n）; }

则以下叙述中不正确的是

A．若只在主函数中对函数f进行说明，则只能在主函数中正确调用函数f

B．若在主函数前对函数f进行说明，则在主函数和其后的其他函数中都可以正确调用函数f

C．对于以上程序，编译时系统会提示出错信息：提示对f函数重复说明

D．函数f无返回值，所以可用void将其类型定义为无返回值型

**三、综合题**

1.程序改错。

extern int x =10 ;

void fun1(int a){

return fun2(x+a) ;

}

int fun2(){

retun x + 10 ;

}

int x = 10 ;

int main(void){

const int b = 5 ;

printf("%d\n",fun1(b)) ;

fun1(b) ;

b++ ;

printf("%d\n",fun2()) ;

}

2.请阅读下面的代码段，指出其中每个对象属于程序的那个区域。

#include<stdio.h>

int a = 0 ;

void func(int b){

printf("%d",b);

}

int main(void){

static int c = 10 ;

int \*p = malloc(sizeof(int)) ;

char \*name = "tom”;

void (\*pf)(int) = func;

………

return 0;

}

请将下列对象放到指定区域：a,b,c,p,\*p,name, \*name,pf,\*pf

全局区

文字常量区

堆区

栈区

代码区

3. 按照函数原型语句“void p(int n);”编写一个递归函数显示出如下图形，此图形是n=5的情况。

1

22

333

4444

55555

## 本章答案

### A类

**一、填空题**

1.12，2.略

**二、选择题**

**三、综合题**

### B类

**一、填空题**

**二、选择题**

**三、综合题**