第 5 章　循环结构程序设计

第一单元　基础知识

一、知识点梳理

内　　容	描　　述	备　　注

for 语句	for(表达式 1; 表达式 2; 表达式 3）
	{
	　　循环体语句
	}
		适合循环次数已知、计数控制的循环

while 语句	while(循环表达式)
	{
	　　循环体语句
	}
		当型循环结构，适合循环次数未知、条件控制的循环。循环体中通常包含改变循环表达式的语句

do-while 语句	do { 　　循环体语句 } while(循环表达式)	直到型循环结构，循环至少执行一次，适合循环次数未知、条件控制的循环。循环体中通常包含改变循环表达式的语句
break 语句	用于退出 switch 或一层循环结构	
continue 语句	用于结束本次循环，继续执行下一次循环	

二、基础题

1．以下程序段____。

```
x=-1;
do
{ x=x*x; }
while (!x);
```

A）是死循环　　B）循环执行两次　　C）循环执行一次　　D）有语法错误

2．以下不是无限循环的语句是____。

A）for(y=0,x=1;x>y;y++) i=x;　　B）for(; ;x++);

C）while(1) {x++;}　　D）for(i=10; ;i--) sum=sum+i;

3．下面程序段的运行结果是____。

```
for(i=1;i<=5;)
  printf("%d",i);
```

```
  i++;
```

A）12345　　B）1234　　C）15　　D）无限循环

4．以下程序段的运行结果是____。

```
int a,y;
a=10;y=0;
do
{  a+=2; y+=a;
   printf("a=%d y=%d\n",a,y);
   if (y>20) break;
}while(a=14);
```

A）a=12 y=12
　　a=14 y=16
　　a=16 y=20
　　a=18 y=24

B）a=12 y=12
　　a=16 y=28

C）a=12 y=12
　　a=14 y=26

D）a=12 y=12
　　a=14 y=44

5．在执行以下程序时，如果从键盘上输入 ABCdef<回车>，则输出结果为____。

```
#include<stdio.h>
int main()
{   char ch;
    while((ch=getchar())!='\n')
    {  if(ch>='A' && ch<='Z')ch=ch+32;
       else if(ch>='a' && ch<'z')ch=ch-32;
       printf("%c",ch);
    }
    printf("\n");
    return  0;
}
```

A）ABCdef　　B）abcDEF　　C）abc　　D）DEF

6．下列选项中，关于 for 循环描述正确的是____。

A）for 循环只能用于循环次数已经确定的情况

B）for 循环的循环体可以是一个复合语句

C）在 for 循环中，不能用 break 语句跳出循环体

D）for 循环的循环体不能是一个空语句

7．设 i、j、k 均为 int 型变量，则执行完下面的 for 循环后，k 的值为____。

```
for(i=0,j=10;i<=j;i++,j--) k=i+j;
```

A）12　　B）10　　C）11　　D）9

8．有以下程序段：

```
int k=0;
while(k=1)k++;
```

while 循环执行的次数是____。

A）无限次　B）有语法错，不能执行　C）一次也不执行　D）执行一次

9．C 语言中 while 和 do-while 循环的主要区别是____。

A）do-while 的循环体至少无条件执行一次

B）while 的循环控制条件比 do-while 的循环控制条件严格

C）do-while 允许从外部转到循环体内

D）do-while 的循环体不能是复合语句

10．下列选项中，叙述正确的是____。

A）continue 语句的作用是结束整个循环的执行

B）只能在循环体内和 switch 语句体内使用 break 语句

C）在循环体内使用 break 语句或 continue 语句的作用相同

D）从多层循环嵌套中退出时，只能使用 goto 语句

11．下列选项中，对下面程序段描述正确的是____。

```
for(t=1;t<=100;t++)
{   scanf("%d", &x);
    if(x<0)continue;
    printf("%d\n", t);
}
```

A）当 x<0 时，整个循环结束　B）当 x>=0 时，什么也不输出

C）printf()函数永远不会执行　D）最多允许输出 100 个非负整数

12．以下程序的输出结果是____。

```
#include"stdio.h"
int main()
{   int i;
    for(i=1;i<=5;i++)
    if(i%2) printf("*");
    else continue;
    printf("#");
    printf("$\n");
}
```

A）***#$　B）#*#*#*$　C）*#*#*#$　D）**#*$

13．以下程序的输出结果是____。

```
#include"stdio.h"
int main()
{   unsigned int num, k;
```

```
    num=26;k=1;
    do {
        k*=num%10;
        num/=10;
        } while(num);
    printf("%d\n", k);
}
```

A）2　　B）12　　C）60　　D）18

14．有以下程序：

```
#include"stdio.h"
int main()
{   int n1, n2;
    scanf("%d", &n2);
    while(n2!=0)
    {   n1=n2%10;
        n2=n2/10;
        printf("%d", n1);
    }
}
```

程序运行后，如果从键盘上输入 1298，则输出结果是____。

A）8921　　B）89　　C）21　　D）1298

15．以下程序运行后的输出结果是____。

```
#include"stdio.h"
int main()
{   int s=0, k;
    for(k=7;k>=0;k--)
    {  switch(k)
       {
          case 1:
          case 4:
          case 7: s++; break;
          case 2:
          case 3:
          case 6:break;
          case 0:
          case 5: s+=2; break; }
    }
    printf("s=%d\n", s);
}
```

A）s=5　　B）s=1　　C）s=3　　D）s=7

第二单元　实验

一、常见错误小结

常见错误实例	错 误 描 述	错 误 类 型

while(i<=n)
{　 sum=sum+i;
　 i++;　 }
	在循环开始前，未对计数器变量、累加器变量初始化
	正确写法：
	i=1;sum=0;
	while(i<=n)
	{　 sum=sum+i;
	　 i++; }
		运行错误

i=1;sum=0;
while(i<=n)
　 sum=sum+i;
　 i++;
	for、while 循环的循环体未加{ }
	正确写法：
	i=1;sum=0;
	while(i<=n)
	{　 sum=sum+i;
	　 i++; }
		运行错误

for(i=1;i<=n;i++);
{　…　}
或
while(i<=n);
{　…　}
	for/while 语句后面加分号，使得 for/while 循环变成了空循环	运行错误

i=1;sum=0;
while(i<=100)
{　 sum=sum+i;
}
	循环体中缺少改变循环条件的语句，导致死循环
	正确写法：
	i=1;sum=0;
	while(i<=100)
	{　 sum=sum+i;
	　 i=i+1;　 }
		运行错误

do{ 　… }while(i<=100)	while 后面缺少分号 正确写法： do{ 　… }while(i<=100);	编译错误
for(i=1,i<n,i++)	用逗号分隔 for 语句中的 3 个表达式 正确写法： for(i=1; i<n; i++)	编译错误

二、编程技能——学会程序排错

程序调试是一个修正错误的过程。为了修正错误，需要根据错误的迹象，分析错误的

原因，确定错误的准确位置，这主要依靠的是排错技术。排错过程开始于一个测试用例的执行，当测试结果与期望输入有出入时就表现出了错误征兆，此时应先找出错误的原因，再修正错误。

常用的排错方法如下。

（1）缩减输入数据，设法找到导致失败的最小输入。

（2）采用注释的方法“切掉”一些代码，减少有关的代码区域，调试无误后再还原被“切掉”的代码。

（3）在合适的地方安排打印语句。

（4）以单步方式运行关键代码。

找到程序的错误后，就要考虑如何进行改错。改错时要注意如下几个问题。

（1）不要急于修改错误，先思考一下，修改相关代码行会不会引发其他问题。

（2）程序中可能“潜伏”着同一类型的错误，这时要改正所有的类似错误。

（3）改错后要立即进行回归测试（regression test），即回过头来，对以前修复过的 bug 重新进行测试，看该 bug 是否会重新出现，以免修复一个 bug，又引起其他新的 bug。

【例 5-1】 编程计算 1～200 中所有能被 3 整除但不能被 5 整除的整数之和。

源代码如下：

```
#include<stdio.h>
int main()
{   int i,sum=0;
    for(i=1;i<=200;i++)
         if(i%3==0&&i%5!=0) sum=sum+i;
    printf("s=%d\n",sum);
    return 0;
}
```

程序运行结果如下：

```
s=5268
```

运行结果是否正确？可以用如下方法测试。

① 缩小数据范围，例如，将测试范围改成 1～20，将语句修改为 for(i=1;i<=20;i++)，检测输出结果是否正确。

② 增加打印语句，以检查每次循环的中间结果，代码如下。

```
#include<stdio.h>
int main()
{   int i,sum=0;
    for(i=1;i<=200;i++)
         if(i%3==0&&i%5!=0)
         {   sum=sum+i;
             printf("%5d",i);//输出每个符合条件的 i 的值
         }
    printf("\ns=%d\n",sum);
    return 0;
}
```

程序运行结果如下：

```
   3    6    9   12   18   21   24   27   33   36   39   42   48   51   54   57
  63   66   69   72   78   81   84   87   93   96   99  102  108  111  114  117
 123  126  129  132  138  141  144  147  153  156  159  162  168  171  174  177
 183  186  189  192  198
s=5268
```

③ 以单步方式运行代码。

首先，将光标定位在 for 语句，选择“组建”→“开始调试”→Run to Cursor 命令，如图 5-1 所示。

图 5-1　Run to Cursor 命令

页面显示如图 5-2 所示，按 F11 键单步执行命令或单击图标 {}，程序会一步一步执行，窗口中 i 和 sum 的值也会随之改变。若要终止调试，选择“调试”菜单下的 Stop Debugging 命令。具体的调试流程，可以查看附录 A 中的“三、程序调试——逻辑错误”。

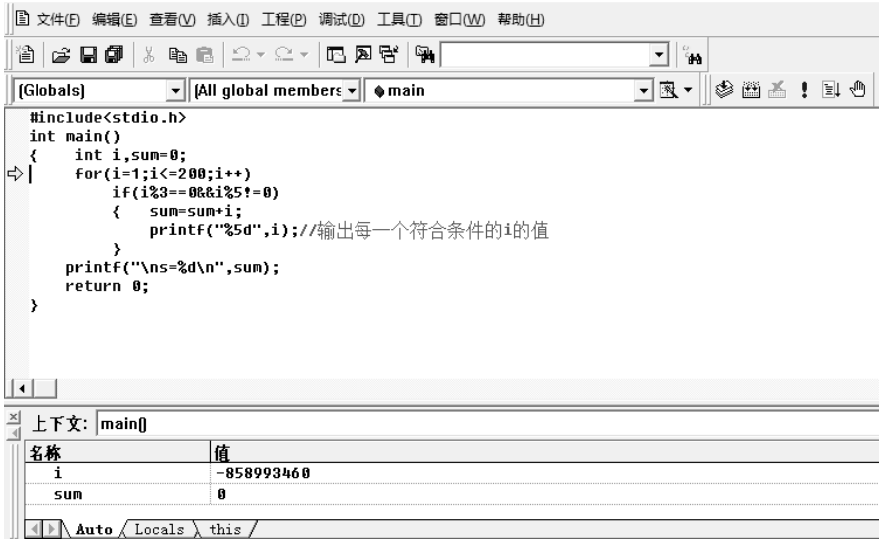

图 5-2　程序单步运行窗口

三、实验指导

实验 5　循环结构程序设计-单重循环

1. 实验目的

（1）掌握用 while 语句、do-while 语句和 for 语句实现单重循环的方法。

（2）掌握在程序设计中用循环的方法实现一些常用算法（如求和、求积和分类统计等）。

（3）学习调试程序的一些技巧。

2. 实验内容和步骤

【基础部分】理解单重循环结构

（1）给定程序 c5-1-1.c 的功能是：在屏幕中输出如图 5-3 所示图形。分析程序的运行结果，并回答问题。

图 5-3　程序 c5-1-1.c 的运行结果

```
/*c5-1-1.c */
#include<stdio.h>
int main()
{
    int i=1;
    while(i<=5)
    {
        printf("*****\n");
        i=i+1;
    }
    return 0;
}
```

① 画出该程序的流程图。

② 编辑并运行该程序，查看程序运行的结果。

③ 将 c5-1-1.c 稍作改动，去掉程序中的一对花括号，并另存为 c5-1-2.c。程序代码如下：

```
/* c5-1-2.c */
#include<stdio.h>
```

```
int main()
{
    int i=1;
    while(i<=5)
    printf("*****\n");
    i=i+1;
    return 0;
}
```

运行程序，观察程序运行的结果，分析出现这个结果的原因。

【提高部分】掌握单重循环结构程序设计

1）程序填空

说明：程序中有多个空，如（1）、（2）等需要补充完整。将程序中的__（1）__、__（2）__等删除后，在相应的位置填入正确答案并调试直到得到正确结果为止。

注意：不要随意改动程序，不得增行或删行，也不得更改程序的结构！

（1）给定程序 c5-2-1.c 的功能是：计算 1～M（M 为偶数）的奇数之和及偶数之和。完善程序，使程序运行结果如图 5-4 所示。

```
偶数之和是：  2550
奇数之和是：  2500
```

图 5-4　程序 c5-2-1.c 的运行结果

```
/*   c5-2-1.c    */
#include <stdio.h>
#define  M   100
int main()
{
    int a,b,i;
    a=0;b=0;
    /**************found************/
    for(i=1;__（1）__;i+=2)
    {
        a=a+i;
        /**************found************/
        __（2）__;
    }
    printf("偶数之和是： %d\n",b);
    printf("奇数之和是： %d\n",a);
    return 0;
}
```

（2）给定程序 c5-2-2.c 的功能是：从键盘输入整数，分别计算所输入的正整数的和、负整数的和。当键盘输入为 0 时，结束输入并输出计算结果。完善程序，使程序运行结果

如图 5-5 所示。

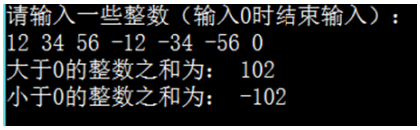

图 5-5　程序 c5-2-2.c 的运行结果

```
/*   c5-2-2.c    */
#include <stdio.h>
int main()
{
    int  x, sum1, sum2;
    /**************found************/
        （1）
    printf("请输入一些整数(输入 0 时结束输入)：\n");
    scanf("%d", &x);
    while ( x != 0 )
    {
        /**************found************/
            （2）
            sum1 = sum1+x;
        else
            sum2 =sum2+x;
        scanf("%d", &x);
    }
    printf("大于 0 的整数之和为： %d\n", sum1);
    printf("小于 0 的整数之和为： %d\n", sum2);
    return 0;
}
```

（3）给定程序 c5-2-3.c 的功能是：计算正整数 num 的各位上的数字之积。

例如，若从键盘输入 252，则运行结果如图 5-6（a）所示；若从键盘输入 202，则运行结果如图 5-6（b）所示，请完善程序。

（a）　　（b）

图 5-6　程序 c5-2-3.c 的运行结果

```
/*   c5-2-3.c    */
#include <stdio.h>
int main( )
{
    int num,k;
    /************found************/
```