实验一 熟悉C程序的运行与调试方法

1.1实验目的

（1）熟悉DevC++、CLion等C语言的开发环境；

（2）掌握C程序的编辑、编译、链接和运行方法。

（3）通过运行简单的C程序，学会在集成开发环境中调试程序的方法。

（4）提高程序查错和排错的能力。

1.2实验内容及要求

1．按下面的实验步骤编辑、编译和运行程序，使程序输出正确结果。

【**程序说明**】若一个数能表示成某个数的平方的形式，这个数即为完全平方数。有一个数，它加上100后是一个完全平方数，再加上168也是一个完全平方数，该数是多少？下面的程序找出在100000内满足该条件的数。程序输出结果应该为：

21

261

1581

【**源程序**】 下面给出原程序代码。

#include <math.h> /\*数学函数库的头文件\*/

#include <stdio.h>

int main()

{

long x,y;

for(i=1;i<100000;i++)

{

x=sqrt(i+100); /\*sqrt是求平方根的函数\*/

y=sqrt(i+268);

if(x\*x=i+100)&&(y\*y=i+268)

printf("\n%d",i);

}

return 0;

}

【**实验步骤**】整个实验步骤包括建立项目、编辑源程序、编译和运行程序。

1. 创建一个项目（project）。
2. 编辑源文件和保存。
3. 编译并修改错误。

编译程序时，编译器会在日志窗口中给出error和warning信息，提示在哪个文件的哪一行有什么错误。error信息是语法错误，即程序的编写违反了C语言的语法规则。warning是警告信息，通常不影响程序执行，但是有些警告信息常常预示着隐蔽较深的实际错误，必须予以重视。

可以通过双击日志窗口的提示信息跳转到对应的源代码处进行修改，分析出错误原因并纠正，修改完后保存，然后重新编译，直至编译成功。

1. 运行。

2．按下面的实验步骤编辑、编译、运行和调试程序，使程序输出正确结果。

【**程序说明**】输入一段正文，统计字符数和行数。例如，输入（↙️代表回车）：

Asd↙️

Xcvb↙️

输出为：

字符数:9,行数:2

【**源程序**】 下面给出原程序代码。

#include <stdio.h>

int main()

{

char c,numchar,numline;

numchar=0;

numline=0;

printf("输入若干行文本，行首输入Ctrl+z结束。\n");

while(c=getchar() !=EOF)

{

numchar++;

if(c=='\n')

numline++;

}

printf("字符数:%d,",numchar++);

printf("行数:%d\n",numline);

return 0;

}

【**实验步骤**】整个实验步骤包括建立项目、编辑、编译、运行和调试程序。

1. 根据程序说明，阅读程序，理解编程思路和方法。
2. 创建一个项目。
3. 编辑源文件和保存。
4. 编译并运行程序。

运行时输入题目所给样例，此时会发现输出结果为“字符数:9,行数:0”，显然结果是不对的。

下面应用断点、单步跟踪等调试方法对程序进行调试，找出错误。

1. 设置断点。

在第7行设置一个断点。

1. 启动调试器。

打开观察窗口观察变量c、numchar和numline的值，根据程序当前运行情况，判断其值是否正确。

1. 单步执行。

从断点开始单步执行每一行程序。单步执行到第10行（也可以在第10行设置另外一个断点，直接运行到下一个断点）。观察变量c的值是多少，思考结果对不对。

**分析**：观察窗口看到的c的值为1，根据程序的功能，应该将键盘输入的第一个字符给c，程序执行的结果显然不符合预期，问题应该出在表达式c=getchar() !=EOF。可见，通过单步执行程序同时观察变量值的变化，很快就能聚焦到有错误测代码行。仔细分析该表达式，结果为什么是1？如何改正？

1. 结束调试，纠正该错误，重新编译运行程序直至结果正确。

3．按下面的实验步骤编辑、编译、运行和调试程序，使程序最终输出正确结果。

【**程序说明**】求1000内的完数，如果一个数等于他的因子之和，则称该数为“完数”（又称“完全数”或“完美数”）。例如，6的因子为1、2、3，而6=1+2+3，因此6是“完数”。函数isPerfect用来判断一个整数是否完数，是则返回1，否则返回0。

【**源程序**】下面给出源程序代码。

#include <stdio.h>

/\*判断x是否完数，是则返回1，否则返回0\*/

int isPerfect(int x)

{

int y,s;

for(s=0,y=1;y<x/2;y++)

{ if(!(x%y))

s+=y;

}

if(s==x)

return 1;

return 0;

}

int main()

{

int a;

printf("1000内的完数有：");

for(a=1;a<=1000;a++) /\*遍历所有数\*/

if(isPerfect(a))

printf("%8d",a);

return 0;

}

【**实验步骤**】整个实验步骤包括建立项目、编辑、编译、运行和调试程序。

1. 根据程序说明，阅读程序，理解编程思路和方法。
2. 创建一个项目，编辑源文件和保存。
3. 编译并运行程序。

纠正所有的error、warning，然后运行，此时会发现输出结果为：

1000内的完数有：24

**分析**：依据程序说明，6是完数没有输出，24的因子为1、2、3、4、6、8、12，而1+2+3+4+6+8+12不等于24，24不是完数却输出了，显然结果不对。

1. 在第20行设置断点。
2. 启动调试器。

运行到第20行时程序暂停。打开观察窗口跟踪观察变量a的值，查看for循环语句的执行情况。反复运行，直到a的值为6，因为1-5都不是完数，没有输出是正常的，但6是完数，在步骤（3）中没有输出6，说明函数isPerfect有问题，需要单步执行的方式进入函数isPerfect查找原因。

1. 跟踪进函数isPerfect。

单步执行后，程序运行到第6行，从观察窗口看到，函数参数已经被赋值为6，局部变量y和s还未被赋值，是系统赋予的随机值，因为第6行代码尚未执行。

继续单步执行，观察for的执行流程以及变量y和s的值，可以看到当y为3，执行到第6行时，再单步执行，程序会运行到第10行，说明y的值为3时退出循环，3没有加到s中，但是3是6的因子，按理应该继续循环，可能循环条件表达式描述有问题，漏检了边界值。请思考如何修改。

1. 跳出isPerfect函数。

可以执行跳出函数功能，则跳出isPerfect函数体，回到函数调用处（第20行）。

1. 停止调试。

修改源程序，即第6行中的条件判断表达式，重新编译运行，直到结果正确。

4．请单步运行下面程序，体会循环的执行过程，在单步执行期间，注意观察各个变量值的变化过程，找出程序中有错误的代码行，分析错误原因并改正错误，使程序最终输出正确结果。

【**程序说明**】输入一个正整数，统计该数各位数字中零的个数，以及找出最大的数字。例如，输入：

12007500↙️

则输出应为：

该数有4个零，最大数字是7

【**源程序**】下面给出源程序代码。

#include <stdio.h>

int main()

{

int count,max,x,t;

printf("请输入一个正整数：");

scanf("%d",&x);

count=max=0;

while(x)

{

t=x%10;

if(t=0)

count++;

if(t>max)

t=max;

x=x/10;

}

printf("该数有%d个零，最大数字是%d\n",count,max);

return 0;

}