暨南大学本科实验报告专用纸

课程名称 高级语言程序设计 成绩评定

实验项目名称 字符串编译 指导教师 张鑫源

实验项目编号 ② 实验项目类型 实验地点 家中

学生姓名 位雨昕 学号 2019051112

学院 智能科学与工程学院 系 专业 信息安全

实验时间 2020 年 3 月 15 日 上午～3 月 15 日 下午 温度 ℃湿度

（一）实验目的

1．进一步了解 Visual Studio 的使用以及 C 语言程序的结构；

2．接触 C 语言中的部分常用函数、初步掌握其使用方法；

3．锻炼个人的编程操作能力。

（二）实验内容和要求

内容：

a) 将用户输入的字符串编成密码串，编码的规则如下：对于第 n 个字符，如果该字符是英文字母，那么用英文字母表后面第 n 个字母代替该字母。例如，用户输入“ab, cd”，那么程序应该将其译为“bd, hj”。

b)可以将用户输入的密码串译回原串，例如用户输入密码串“bd, hj”，那么程序要将其译回“ab, cd”。

要求：

1. 本实验约定字符串首字符为第 1 个字符；
2. 英文字母表形成一个环，例如 z 是第 26 个字母，那么在此环中 z 的下一个字母为 a, y 后面的第 3 个字母为 b, 依次类推；
3. 允许用户输入任意字符。

（三）主要仪器设备

仪器：计算机

实验环境：Visual Studio Community 2019

（四）源程序

```
#include<stdio.h>
#include<string.h>

void main()
{

    int choice;
    printf("加密请输入0，解密请输入1：\n");
    scanf_s("%d", &choice);

    if (choice == 0)                              //加密的场合
    {
        int i, n = 1, length;                     //n代表该字符在字符串中是第n个
        char s1[30];
        char c1[30];
        printf("请输入需要加密的字符串：\n");
        getchar();
        gets_s(s1, 30);
        length = sizeof(s1);                      //用sizeof函数提取字符串长度
        for (i = 0; i < length; i++, n++)
        {
            n %= 26;                              //字母周期为26
            if (s1[i] >= 'a' && s1[i] <= 'z')     //加密内容为小写字母的场合
            {
                if (s1[i] + n > 'z')              //需要（向后）用到字母表形成的环的场合
                {
                    c1[i] = 'a' + (n - ('z' - s1[i] + 1));
                }
                else
                    c1[i] = s1[i] + n;
            }
            else if (s1[i] >= 'A' && s1[i] <= 'Z')//加密内容为大写字母的场合
            {
                if (s1[i] + n > 'Z')              //需要用到字母表形成的环的场合
                {
                    c1[i] = 'A' + (n - ('Z' - s1[i] + 1));
                }
                else
                    c1[i] = s1[i] + n;
            }
            else c1[i] = s1[i];                   //加密内容不是字母的场合
        }
```

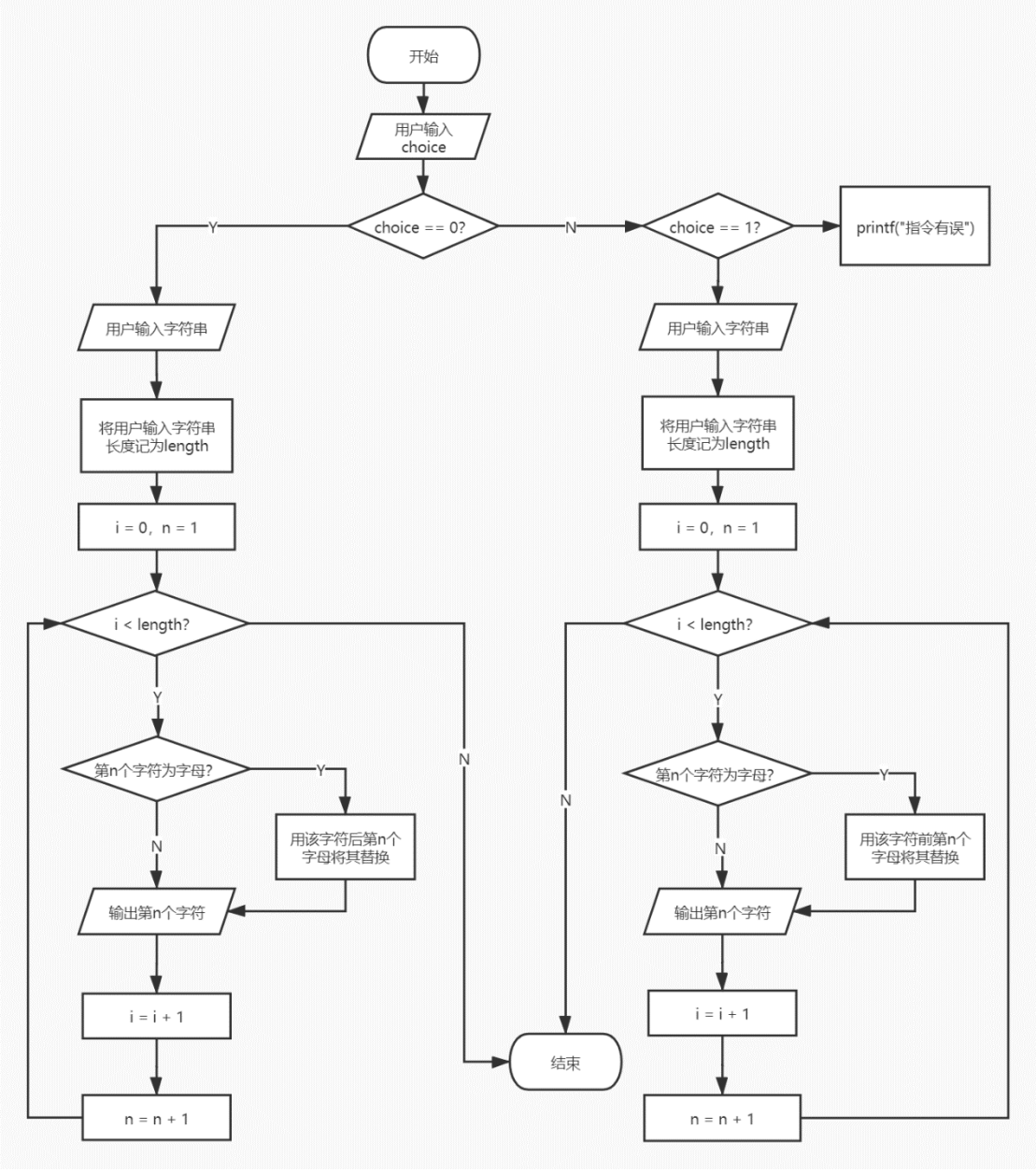
```
        printf("加密结果为：\n");
        printf("%s", &c1);
    }
    else if (choice == 1)                          //解密的场合
    {
        int i, n = 1, length;
        char s2[30], c2[30];                       //s2为密码串，c2为完成解密后的原串
        printf("请输入需要解密的字符串：\n");
        getchar();
        gets_s(s2, 30);
        length = sizeof(s2);
        for (i = 0; i < length; i++, n++)
        {
            n %= 26;
            if (s2[i] >= 'a' && s2[i] <= 'z')  //内容为小写字母
            {
                if (s2[i] - n < 'a')           //需要（往前）用到字母表形成的环的场合
                {
                    c2[i] = 'z' - (n - (s2[i] - 'a' + 1));
                }
                else
                    c2[i] = s2[i] - n;
            }
            else if (s2[i] >= 'A' && s2[i] <= 'Z') //内容为大写字母
            {
                if (s2[i] - n < 'A')           //需要用到字母表形成的环
                {
                    c2[i] = 'Z' - (n - (s2[i] - 'A' + 1));
                }
                else
                    c2[i] = s2[i] - n;
            }
            else c2[i] = s2[i];                //解密内容不是字母的场合
        }
        printf("解密结果为：\n");
        printf("%s", &c2);
    }
    else
        printf("指令有误,请重启程序并重新输入");

}
```

（五）实验步骤与调试

实验步骤：

1. 对实验进行基本构思，绘制流程图；
2. 启动 Visual Studio，创建新项目。将源程序写在新项目中；
3. 利用“本地 Windows 调试器”进行调试；
4. 进行多次调试并修正，直至得出理想结果。

实验思路：

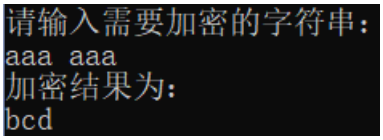


调试：

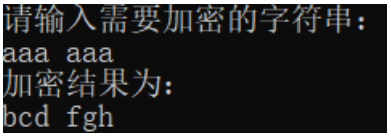
1. 最初没有对 s1、c1 等字符变量添加字符数量，导致报错：从下标要求数组或指针类型，在不是数组的变量上使用了下标；
2. 最初使用 scanf_s 函数读取用户输入的字符串，调试过程中发展此函数会将空格后输入的字符全部阻断。查阅资料后发现 scanf()遇到空格就默认结束扫描，gets()或 fgets()函数更适于“允许用户输入任意字符”的实验要求；

【运行情况】

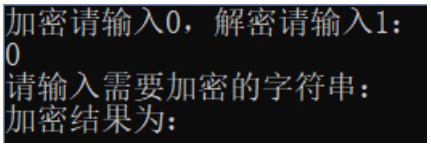
a) 使用 scanf()函数：



b) 修正：使用 gets()函数



3. 用 gets()函数替代 scanf()函数后，不了解该函数的用法，被告知形参和实参的参数的类型不同以及用于函数调用的参数太少或函数声明不正确。查阅资料后得知 gets()函数调用格式为 *gets(s)*。重新输入；
4. 将函数改为 gets()函数后，调试后发现直接跳过了输入字符串。



查阅相关资料后得知：

- *C 语言里的 gets()函数功能是从输入缓存中读取多个字符，遇到回车符时，结束输入。*
- *当使用 gets()函数之前有过数据输入，并且，操作者输入了回车确认，这个回车符没有被清理，被保存在输入缓存中时，gets()会读到这个字符，结束读字符操作。因此，从用户表面上看，gets()没有起作用，而是直接跳过了。*

而在源代码中，gets()函数的前一行是：

```
printf("请输入需要加密的字符串：\n");
```

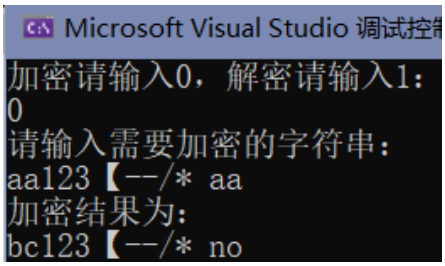
回车符被保存在输入缓存中，gets()函数读取到了这个字符，立刻结束了读取字符的操作。查阅后得知解决方法：

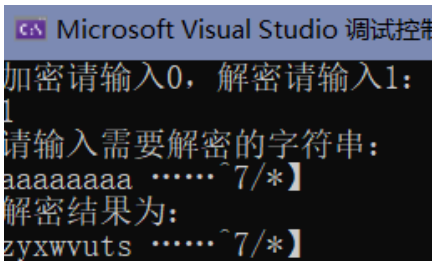
- *方法一：在 gets()前加 fflush(stdin);//强行清除缓存中的数据（windows 下可行）*
- *方法二：根据程序代码，确定前面是否有输入语句，如果有，则增加一个 getchar()命令，然后再调用 gets()命令。*
- *方法三：检查输入结果，如果得到的字符串是空串，则继续读入，如：*

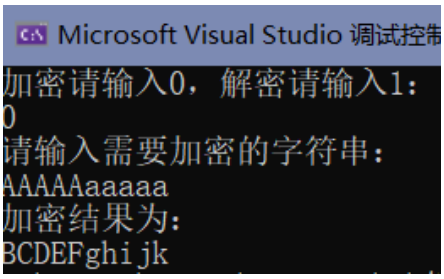
```
char str[100]={0};
do
{
gets(str);
}

while (!str[0]);
```

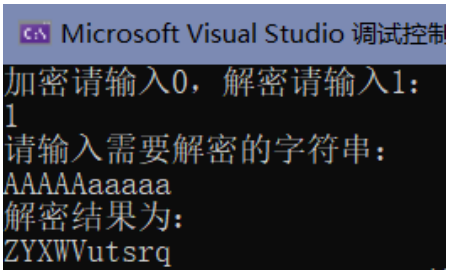
利用方法二解决了用 gets()函数运行时直接跳过输入字符串的问题。

（六）实验结果与分析









1.程序可按实验要求运行；

2.程序情况基本符合理想效果；

3.调试过程中出现的问题均已解决。