1. 实验目的与要求

（1） 掌握汇编语言程序与 C 语言程序混合编程的方法；

（2） 熟悉 C 编译器的基本优化方法;

（3） 了解 C 语言编译器的命名方法，主、子程序之间参数传递的机制。

2. 实验内容

任务 1：在 C 语言序中嵌入汇编语言指令语句序列

对于实验二的程序进行改造，主控程序、以及输入输出等功能用 C 语言实现，学生姓名搜索和成绩计算用 C 程序中嵌入汇编指令语句序列的方式实现。

任务 2：在 C 语言程序中调用汇编语言实现的函数

对于实验二的程序进行改造，主控程序、以及输入输出等功能用 C 语言实现，学生姓名搜索和成绩计算用独立的汇编语言子程序的方式实现；在 C 语言程序中调用汇编语言子程序。

要求：

（1）在不同的 C 语言开发环境中实现与汇编语言程序的混合编程，其操作方法有可能是不同的。请大家选择白己熟悉的 C 语言开发环境并查找相关的资料完成本实验。

（2）在实验报告中，详细地描述采用的开发环境及其实现方法。

（3）观察 C 语言编译器中对各种符号的命名规则（指编译器内部可以识别的命名规则，比如，符号名前面是否加下划线“_”等），主、子程序之间参数传递的机制，通过堆栈传递参数后堆栈空间回收的方法。

（4）对混合编程形成的执行程序，用调试工具观察由 C 语言形成的程序代码与由汇编语言形成的程序代码之间的相互关系，包括段、偏移的值，汇编指令访问 C 的变量时是如何翻译的等。

（5）尝试在 C 语言程序中不合理地入汇编语言的指令语句，达到破坏 C 语言程序的正确性的目的。比如，在连续的几条 C 语言语句中间加入一条修改 AX 寄存器的汇编指令语句，而 AX 的内容在此处本不该被修改，这样就可观察到破坏 C 语言程序正确性的效果（该项实验表明：在 C 语言程序中，若不考虑上下语句翻译成怎样的机器码而随意嵌入汇编指令语句时，有可能存在出错的风险）。

（6）观察 C 编译器的优化策略对代码的影响。

（7）通过调试混合编程的程序，体会与纯粹汇编语言编写的程序的调试过程的差异。

（8）通过本次实验，希望大家明白：不同的编程语言是可以协同解决一个问题的，而且可以利用不同语言的特点来更好地解决问题；利用汇编语言的知识，能够更好地理解高级语言的内部处理原理与策略，为编写更好的 C 语言程序、用好 C 编译器提供支持。

3. 实验过程

一、 提纲：

任务一：在实验二的基础上，进行程序改造，主控程序和输入输出等功能用c语言实现，学生姓名的搜索与成绩的计算用嵌入汇编语句实现。

任务二：对于实验二的程序进行改造，主控程序、以及输入输出等功能用 C 语言实现，学生姓名搜索和成绩计算用独立的汇编语言子程序的方式实现；在 C 语言程序中调用汇编语言子程序。

二、设计思想

任务一：

1.对数组中的学生姓名和数组初始化。

2.提示用户输入名字或“q”退出，“a”计算平均成绩。此过程通过c语言实现。

3.对输入的字符串进行匹配，若是“q”则退出，若是回车，回到循环开始处重新输入，若是“a”则开始计算平均成绩，若是名字，则开始判断系统中是否有该学生。此过程通过汇编实现。

4.若有该学生，计算平均成绩并存入成绩数组，通过汇编实现。

5.比较该学生的成绩区间，输出该学生的等级。通过c语言实现。

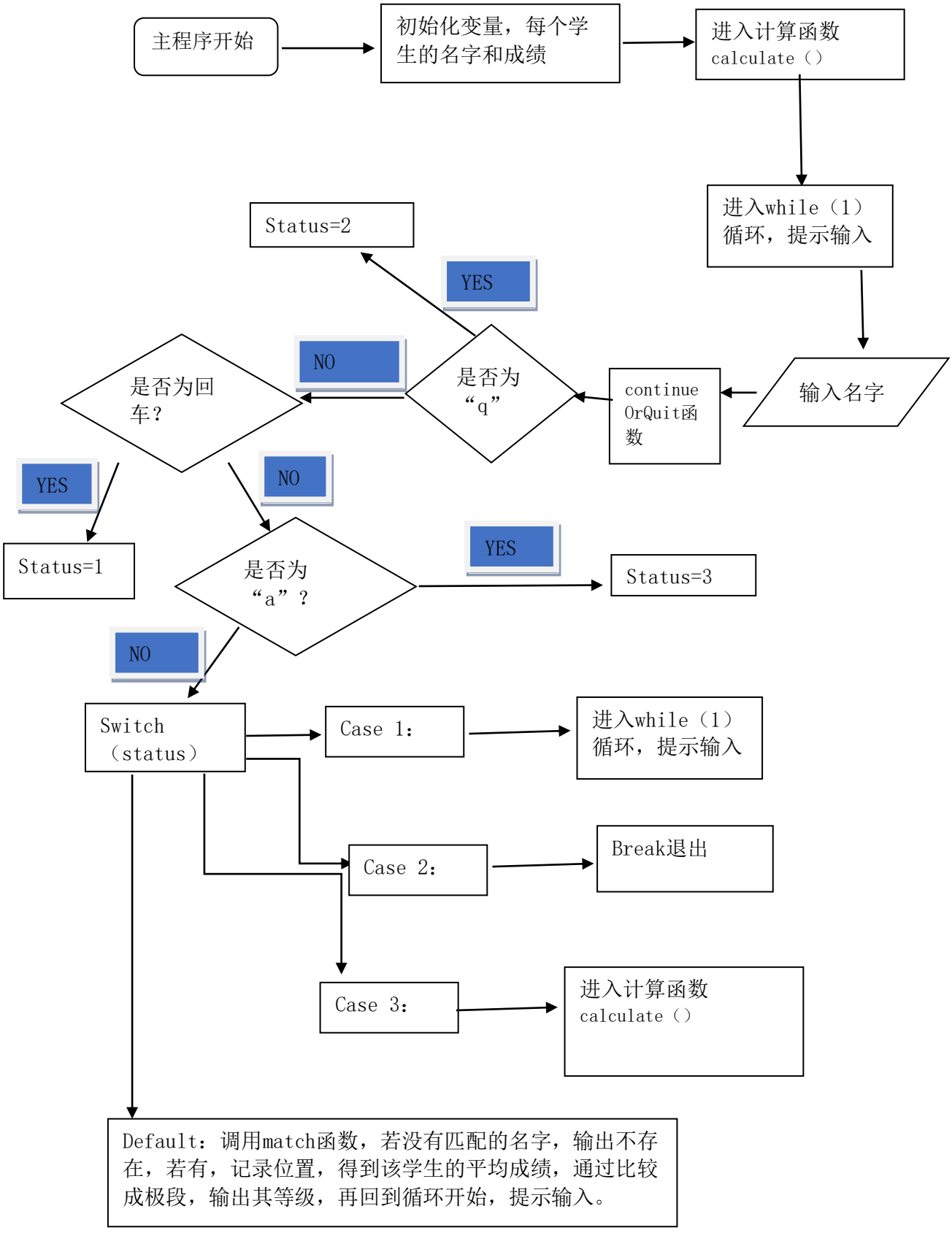
6.回到第一步，重新输入。

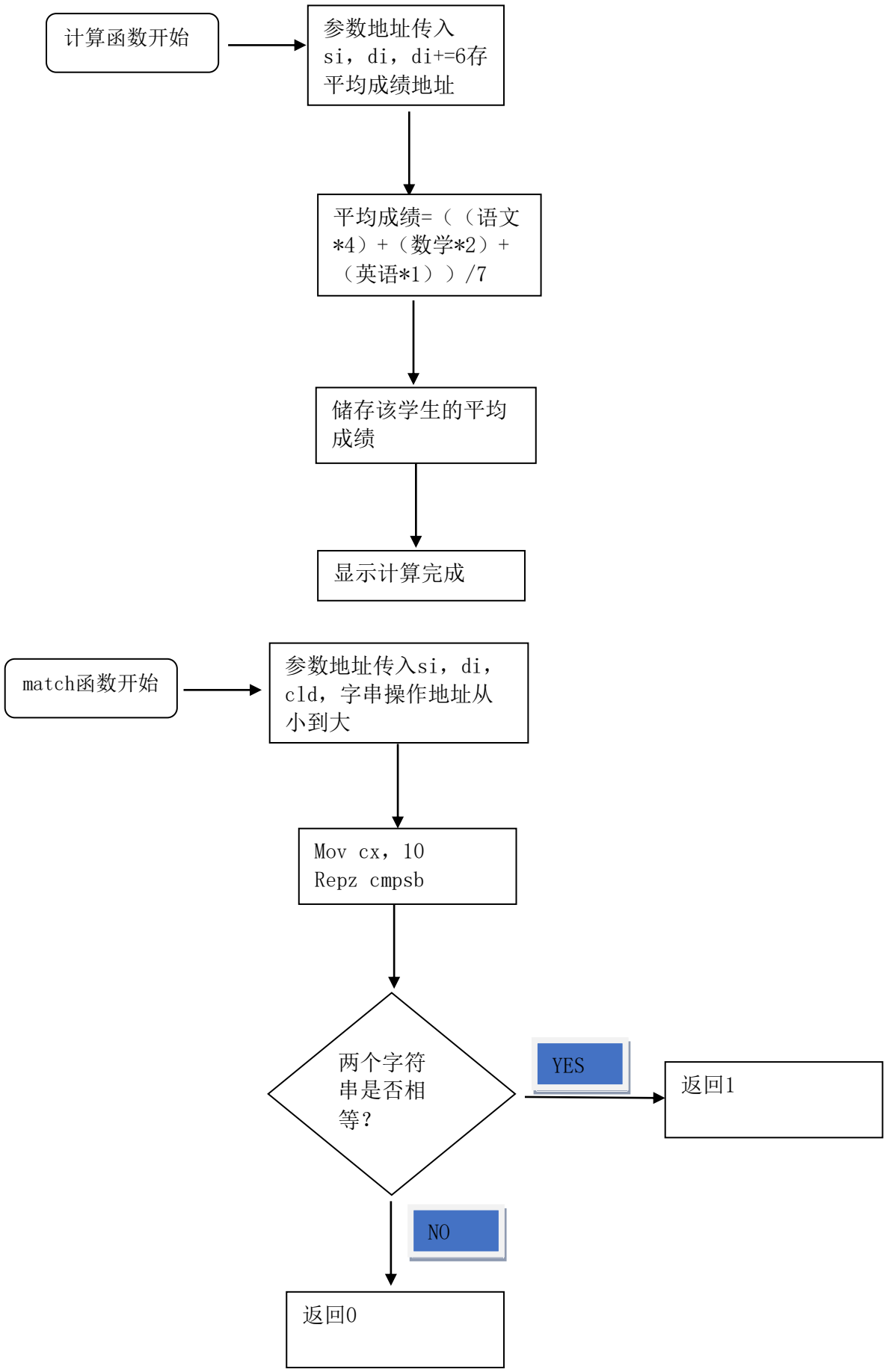
任务二：

1. 主控程序、以及输入输出等功能用 C 语言实现，学生姓名搜索和成绩计算用独立的汇编语言子程序的方式实现；在 C 语言程序中调用汇编语言子程序。新建三个.asm文件分别写三个汇编语言子程序，在main.cpp中调用。
2. 对数组中的学生姓名和分数初始化。
3. 提示用户输入名字或“q”退出，“a”计算平均成绩。此过程通过c语言实现。
4. 对输入的字符串进行匹配，若是“q”则退出，若是回车，回到循环开始处重新输入，若是“a”则开始计算平均成绩，若是名字，则开始判断系统中是否有该学生。此过程通过汇编子程序实现。
5. 若有该学生，计算平均成绩并存入成绩数组，通过汇编子程序实现。
6. 比较该学生的成绩区间，输出该学生的等级。通过c语言实现。
7. 回到第一步，重新输入。

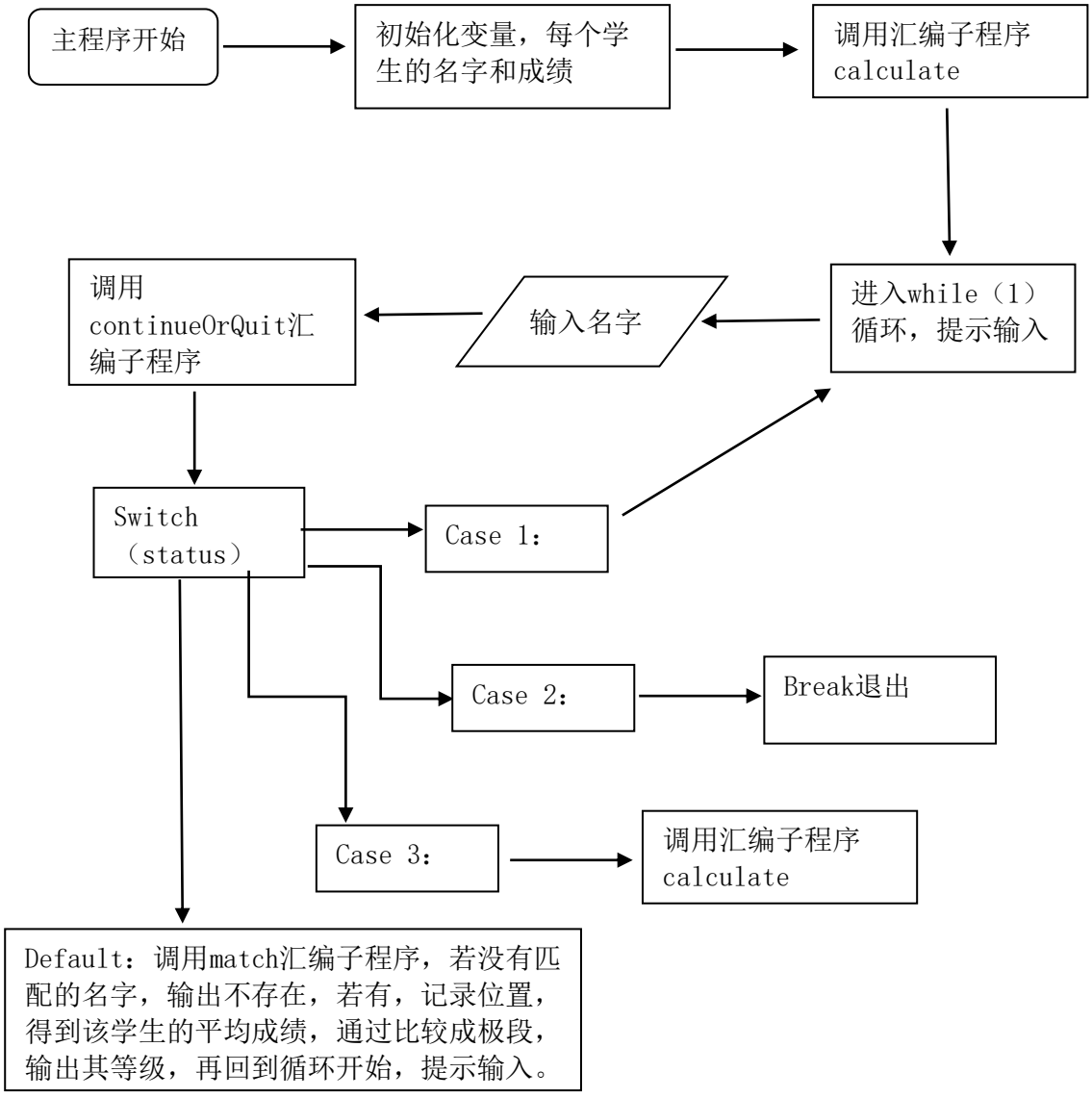
三、流程图

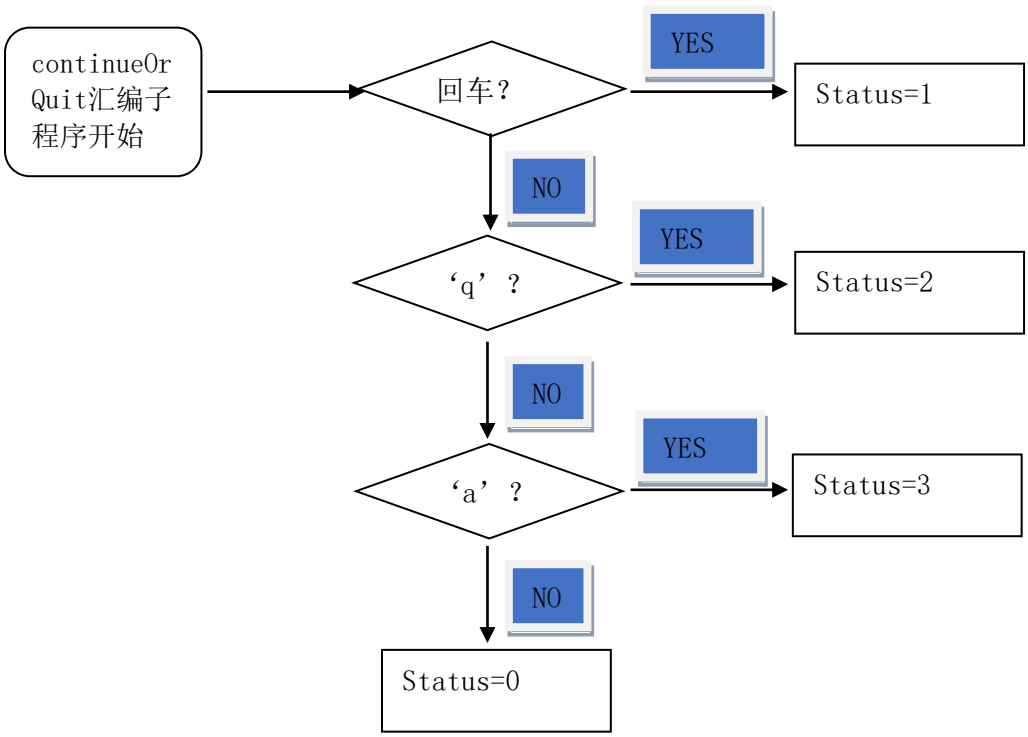
任务一：

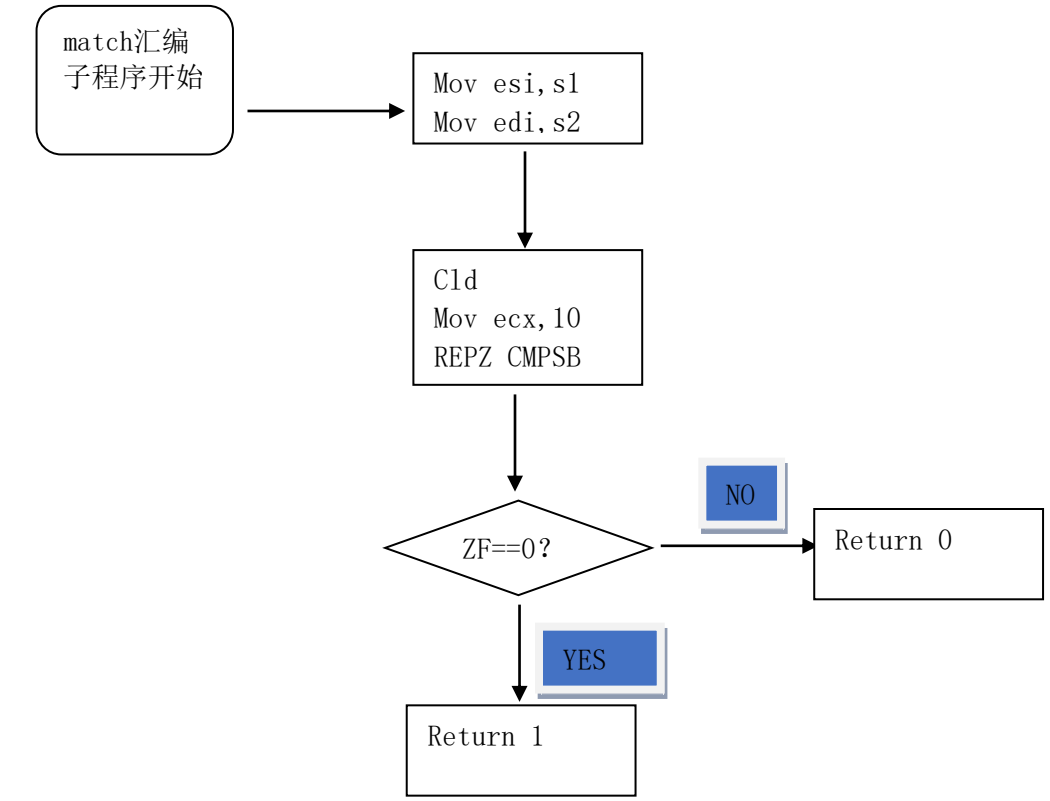

主程序开始
初始化变量，每个学生的名字和成绩
进入计算函数 calculate（）
进入while（1）循环，提示输入
输入名字
continue OrQuit函数
是否为“q”
YES
Status=2
NO
是否为回车？
YES
Status=1
NO
是否为“a”？
YES
Status=3
NO
Switch（status）
Case 1：
进入while（1）循环，提示输入
Case 2：
Break退出
Case 3：
进入计算函数 calculate（）
Default：调用match函数，若没有匹配的名字，输出不存在，若有，记录位置，得到该学生的平均成绩，通过比较成极段，输出其等级，再回到循环开始，提示输入。

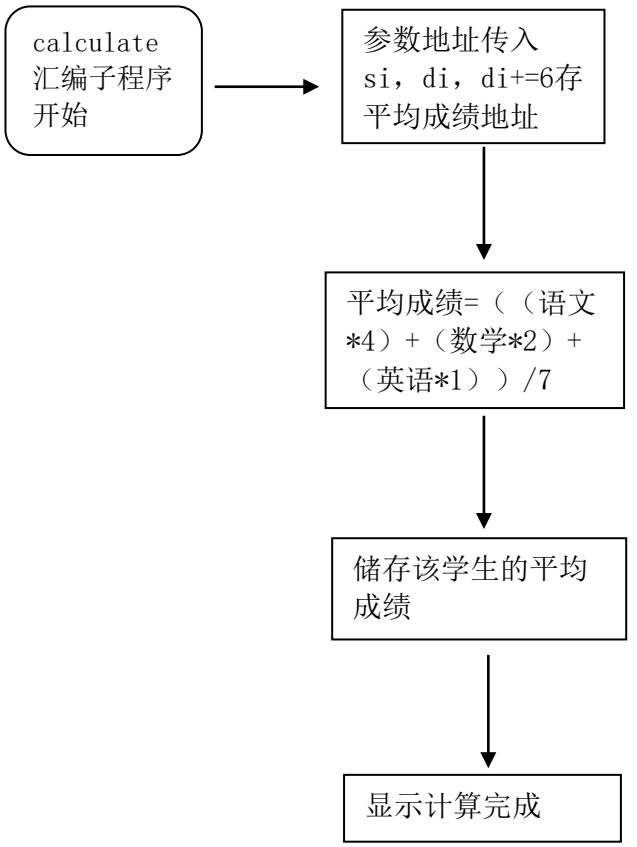

计算函数开始
参数地址传入
si，di，di+=6存
平均成绩地址
平均成绩=（（语文
*4）+（数学*2）+
（英语*1））/7
储存该学生的平均
成绩
显示计算完成
match函数开始
参数地址传入si，di，
cld，字串操作地址从
小到大
Mov cx，10
Repz cmpsb
两个字符
串是否相
等?
YES
返回1
NO
返回0

任务二：


主程序开始
初始化变量，每个学生的名字和成绩
调用汇编子程序calculate
进入while（1）循环，提示输入
输入名字
调用continueOrQuit汇编子程序
Switch（status）
Case 1：
Case 2：
Break退出
Case 3：
调用汇编子程序calculate
Default：调用match汇编子程序，若没有匹配的名字，输出不存在，若有，记录位置，得到该学生的平均成绩，通过比较成极段，输出其等级，再回到循环开始，提示输入。


continueOrQuit汇编子程序开始
回车？
YES
Status=1
NO
‘q’ ？
YES
Status=2
NO
‘a’ ？
YES
Status=3
NO
Status=0


match汇编
子程序开始
Mov esi,s1
Mov edi,s2
Cld
Mov ecx,10
REPZ CMPSB
ZF==0?
NO
Return 0
YES
Return 1


calculate
汇编子程序
开始
参数地址传入
si，di，di+=6存
平均成绩地址
平均成绩=（（语文
*4）+（数学*2）+
（英语*1））/7
储存该学生的平均
成绩
显示计算完成

四、源程序

任务一：

```
//exp2->task1 2020.10.20

#include <stdio.h>
#define NUM 1000
#define N 10

char name[NUM][10], in_name[N];
short score[NUM][4];//一个short 两个字节

int cpy(char* s1, const char s2[]) {
    int j = 0, i = 0;;
    for (j = 0; j < N; j++) {//对s1 进行初始化
        *(s1 + j) = 0;
    }
    while (s2[i]) {
        *(s1 + i) = s2[i];//将s2 字符串复制到s1
        i++;
    }
    return 0;
}

int calculate(short* s) {//汇编计算函数
    _asm
    {
        mov esi, s      //成绩首地址存入esi
        mov edi, s      //成绩首地址存入edi
        add di, 6      //跳过语文数学英语共六个字节的成绩进入平均成绩地址
        cld            //DF 标志位清零 意味着地址指针+2
        lodsw          //AX-<DS:[SI],EI+=2
        add ax, ax      //AX 此时为语文成绩 先乘2
        add ax, ax      //再乘2 为除以7 做准备
        mov bx, ax      //把计算结果存入BX
        lodsw           //AX-<DS:[SI],EI+=2
        add ax, ax      //此时AX 是数学成绩 乘2 为除以7 做准备
        add bx, ax      //语文成绩*4+数学成绩*2
        lodsw           //AX-<DS:[SI],EI+=2
        add ax, bx      //语文成绩*4+数学成绩*2+英语成绩*1
        mov dl, 7
        div dl          //AH=AX/7
        mov ah, 0       //余数清零
        stosw           //ES:[DI]<-AX,DI+=2 DI 是平均成绩位置 算完了存到里面
    }
    return 0;
}


// 回车1，退出2,计算平均3，其他0
int continueOrQuit(char* s1) {//传入输入的名字
```

```
    _asm {
        mov esi, s1 //名字首个字符地址传入SI
        cld //DF 标志位清零
        lodsw //AX-<DS:[SI],EI+=2
        cmp ax, 0000h//AX 和回车比较
        je Continue//如果是回车 跳到continue
        cmp ax, 0071h//AX 和'q'比较
        je Quit//如果是'q'跳到Quit
        cmp ax,0061h//AX 和'a'比较
        je CalAve
    }
    return 0;
Continue:_asm {
}
return 1;
Quit:_asm {
}
return 2;
CalAve:_asm {
}
return 3;
}


// 匹配1，不匹配0
int match(char* s1, char* s2) {//传入两个字符串首地址
    _asm {
        mov esi, s1//s1 地址传入si
        mov edi, s2//s2 地址传入di
        cld//DF 标志位清零
        mov cx, N//CX 存入最多的字符数
        repz cmpsb//repz 每执行一次串指令,DEC CX 只要CX=0 或ZF=0,重复执行结束.cmpsb DS:[SI]-ES:[DI] INC SI, INC DI
        jne False//到最后一个字符都相等 就字符串相等 跳转到True 段
    }
    return 1;
    False:_asm {
    }
    return 0;//字符串不匹配 返回0
}

int main() {
    int flag = 0, num = 0, c = 0, i = 0, j = 0;
    cpy(name[0], "zhangsan");//初始化0-999 号学生
    score[0][0] = 100;
    score[0][1] = 85;
    score[0][2] = 80;
    cpy(name[1], "lisi");
    score[1][0] = 80;
    score[1][1] = 100;
    score[1][2] = 70;
    cpy(name[2], "xzc");
    score[2][0] = 80;
```

```
    score[2][1] = 85;
    score[2][2] = 100;
    for (i = 3; i < NUM; i++) {
        cpy(name[i], "TempValue");
        score[i][0] = 80;
        score[i][1] = 90;
        score[i][2] = 95;
    }
    for (i = 0; i < NUM; i++) {//计算1000 个人的平均成绩
        calculate(score[i]);//score[i]是第 i 个学生的语文成绩地址
    }
    while (1) {
        flag = 0, num = 0;
        for (j = 0; j < N; j++) {
            in_name[j] = 0;//初始化 in_name 字符串
        }
        printf("Please input the student's name:\n");
        printf("(Enter q to quit and enter a to calculate the average grad
e)\n");
        i = 0;
        c = getchar();
        while (c != '\n' && i < N - 1) {
            in_name[i] = c;
            i++;
            c = getchar();//通过getchar 获得名字
        }
        int status = continueOrQuit(in_name);
        if (status == 1) {
            continue;//跳出本次循环 重新提示输入
        }
        if (status == 2) {
            printf("Exit successfully!");
            break;//跳出循环
        }
        if (status == 3) {
            for (i = 0; i < NUM; i++) {
                calculate(score[i]);
            }
            printf("Calculate all the average grades successfully!\n");
            flag = 2;
        }
        for (j = 0; j < 1000; j++) {
            if (match(in_name, name[j]) == 1) {
                num = j;
                flag = 1;
                break;//获得名字的位置 跳出for 循环
            }
        }
        if (flag == 0) {
            printf("%s does not exist!\n", in_name);
        }
        else if(flag==1){
            int t = score[num][3];
```

```
            printf("The grade of %s is %c !\n", in_name, t > 90 ? 'A' : t
> 80 ? 'B' : t > 70 ? 'C' : t > 60 ? 'D' : 'F');
        }
    }
    return 0;
}
```

任务二：

```
;calculate()函数
.386
.model flat,c
public calculate
.code

calculate proc uses esi edi,s:ptr

        mov esi, s
        mov edi, s
        add edi, 6
        cld
        lodsw
        add ax, ax
        add ax, ax
        mov bx, ax
        lodsw
        add ax, ax
        add bx, ax
        lodsw
        add ax, bx
        mov dl, 7
        div dl
        mov ah, 0
        stosw
        ret
calculate endp
end
```

```
;match 函数
.386
.model flat,c
public match
.code
match proc uses esi edi,s1:ptr,s2:ptr
       mov esi,s1
       mov edi,s2
       cld
       mov ecx,10
       repz cmpsb
```

```
       jne False
       mov eax,1
       ret
False: mov eax,0
       ret
match endp
end
```

```
;continueOrQuit 函数
.386
.model flat,c
public continueOrQuit
.code
continueOrQuit proc uses esi,s1:ptr
    mov esi,s1
    cld
    lodsw
    cmp ax,0000h
    je Continue
    cmp ax,0071h
    je Quit
    cmp ax,061h
    je CalAve
    mov eax,0
    ret
Continue: mov eax,1
        ret
Quit:mov eax,2
        ret
CalAve:mov eax,3
        ret
continueOrQuit endp
end
```

```
//main.cpp
#include<stdio.h>
#include<stdlib.h>
#define NUM 1000
#define N 10

char name[NUM][10], in_name[N];
short score[NUM][4];//一个short 两个字节

extern "C" int calculate(short* s);
extern "C" int continueOrQuit(char* s1);
extern "C" int match(char* s1, char* s2);

int cpy(char* s1, const char s2[]) {
    int j = 0, i = 0;;
    for (j = 0; j < N; j++) {//对s1 进行初始化
```

```
        *(s1 + j) = 0;
    }
    while (s2[i]) {
        *(s1 + i) = s2[i];//将s2 字符串复制到s1
        i++;
    }
    return 0;
}

int main() {
    int flag = 0;
    int  num = 0, c = 0, i = 0, j = 0;

    cpy(name[0], "zhangsan");//初始化0-999 号学生
    score[0][0] = 100;
    score[0][1] = 85;
    score[0][2] = 80;
    cpy(name[1], "lisi");
    score[1][0] = 80;
    score[1][1] = 100;
    score[1][2] = 70;
    cpy(name[2], "xzc");
    score[2][0] = 80;
    score[2][1] = 85;
    score[2][2] = 100;

    for (i = 3; i < NUM; i++) {
        cpy(name[i], "TempValue");
        score[i][0] = 80;
        score[i][1] = 90;
        score[i][2] = 95;
    }

    for (i = 0; i < NUM; i++) {//计算1000 个人的平均成绩
        calculate(score[i]);//score[i]是第i 个学生的语文成绩地址
    }

    while (1) {
        flag = 0, num = 0;
        for (j = 0; j < N; j++) {
            in_name[j] = 0;//初始化in_name 字符串
        }
        printf("Please input the student's name:\n");
        printf("(Enter q to quit and enter a to calculate the average grad
e)\n");
        i = 0;
        c = getchar();
        while (c != '\n' && i < N - 1) {
            in_name[i] = c;
            i++;
            c = getchar();//通过getchar 获得名字
        }
        int status = continueOrQuit(in_name);
```

```
        if (status == 1) {
            continue;//跳出本次循环 重新提示输入
        }
        if (status == 2) {
            printf("Exit successfully!");
            break;//跳出循环

        }
        if (status == 3) {
            for (i = 0; i < NUM; i++) {
                calculate(score[i]);
            }
            printf("Calculate all the average grades successfully!\n");
            flag = 2;
        }
            for (j = 0; j < 1000; j++) {
                if (match(in_name, name[j]) == 1) {
                    num = j;
                    flag = 1;
                    break;//获得名字的位置 跳出for 循环
                }
            }

        if (flag == 0) {
            printf("%s does not exist!\n", in_name);
        }
        else if (flag == 1) {
            int t = score[num][3];
            printf("The grade of %s is %c !\n", in_name, t > 90 ? 'A' : t
> 80 ? 'B' : t > 70 ? 'C' : t > 60 ? 'D' : 'F');
        }
    }
    return 0;
}
```

五、实验步骤

任务一：

1.找到自己熟悉的编译环境Visual Studio 2019，查阅资料了解混合编程的规定。

2.在Visual Studio 2019中输入源代码中任务一的代码，编译通过，运行。

3.分别输入“zhangsan”、“q”、“a”和回车验证功能是否成功实现。

4. 分别输入“zhangsan”、“lisi”、“xzc”、“wangwu”验证功能是否成功实现。

5. 通过调试混合编程的程序，体会与纯粹汇编语言编写的程序的调试过程的差异。

6.观察实验结果，撰写任务一部分报告。

任务二：

1. Visual Studio 2019创建一个新项目，新建一个main.cpp文件，三个.asm文件，分别命名为test，test2，test3.
2. 配置test.asm，test2.asm和test3.asm文件。
3. 在Visual Studio 2019中输入源代码中任务二的代码，编译通过，运行。
4. 分别输入“zhangsan”、“q”、“a”和回车验证功能是否成功实现。
5. 分别输入“zhangsan”、“lisi”、“xzc”、“wangwu”验证功能是否成功实现。
6. 通过调试混合编程的程序，体会与纯粹汇编语言编写的程序的调试过程的差异。
7. 观察实验结果，撰写任务二部分报告。

六、实验结果

（一）任务一：

本次任务采用visual studio 2019环境，在c语言函数中，使用_asm{汇编代码段}的方式实现嵌入式，段跳转可以通过_asm{jmp X:} X:_asm{}实现，非跳转可以顺序执行。

1. 编译后运行的结果：

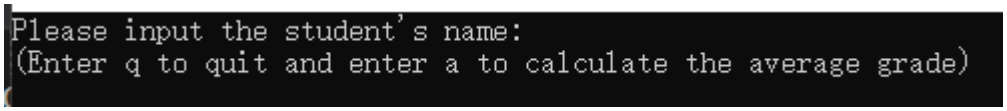


图 1-1

2. 输入“zhangsan”后的结果：

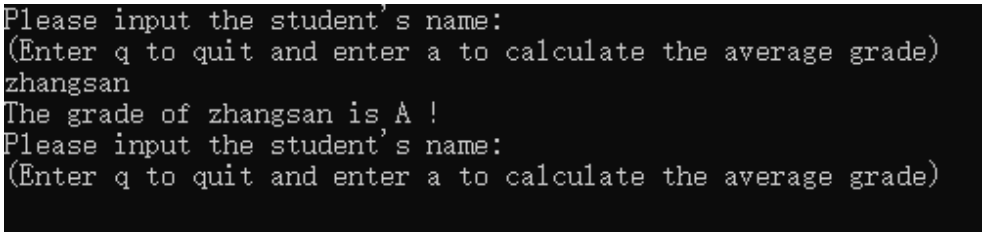


图 2-1

3.输入“q”后的结果：

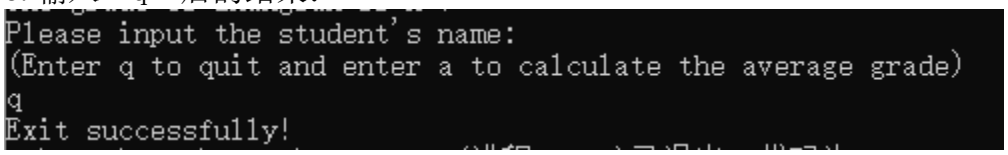


图 3-1

4.输入“lisi”后的结果：

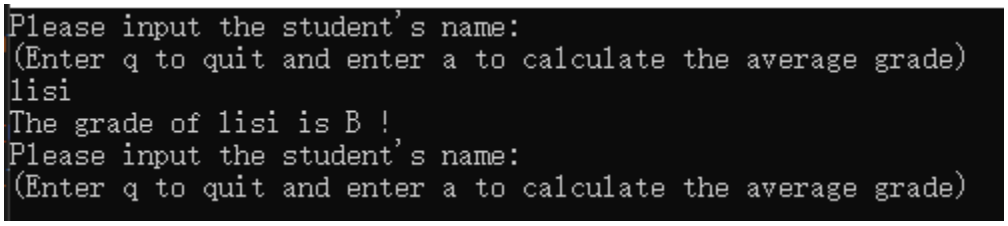


图 4-1

5.输入“xzc”后的结果：

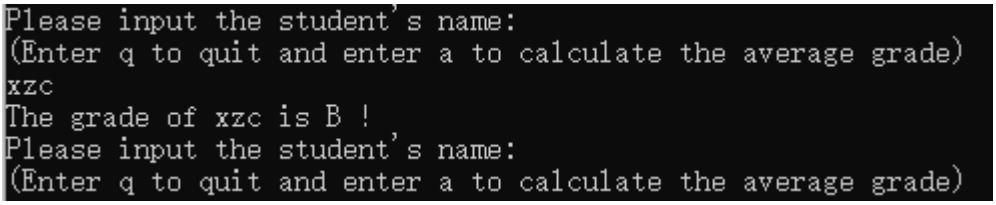


图 5-1

6.输入“wangwu”后的结果：

```
Please input the student's name:
(Enter q to quit and enter a to calculate the average grade)
wangwu
wangwu does not exist!
Please input the student's name:
(Enter q to quit and enter a to calculate the average grade)
```

图 6-1

7. 输入“a”后的结果：

```
Please input the student's name:
(Enter q to quit and enter a to calculate the average grade)
a
Calculate all the average grades successfully!
Please input the student's name:
(Enter q to quit and enter a to calculate the average grade)
```

图 7-1

8. 调试，反汇编后的部分结果：

```
        printf("Please input the student's name:\n");
005C5AB1  push        offset string "Please input the student's name@"... (05C7B54h)
005C5AB6  call        _main (05C104Bh)
005C5ABB  add         esp,4
        printf("(Enter q to quit and enter a to calculate the average grade)\n");
005C5ABE  push        offset string "(Enter q to quit and enter a to@"... (05C7B7Ch)
005C5AC3  call        _main (05C104Bh)
005C5AC8  add         esp,4
        i = 0;
005C5ACB  mov         dword ptr [i],0
        c = getchar();
005C5AD2  mov         esi,esp
        c = getchar();
005C5AD4  call        dword ptr [__imp__getchar (05D0174h)]
005C5ADA  cmp         esi,esp
005C5ADC  call        __RTC_CheckEsp (05C1235h)
005C5AE1  mov         dword ptr [c],eax
        while (c != '\n' && i < N - 1) {
005C5AE4  cmp         dword ptr [c],0Ah
005C5AE8  je          main+309h (05C5B19h)
005C5AEA  cmp         dword ptr [i],9
005C5AEE  jge         main+309h (05C5B19h)
            in_name[i] = c;
005C5AF0  mov         eax,dword ptr [i]
005C5AF3  mov         cl,byte ptr [c]
005C5AF6  mov         byte ptr in_name (05CC848h)[eax],cl
```

图 8-1-1

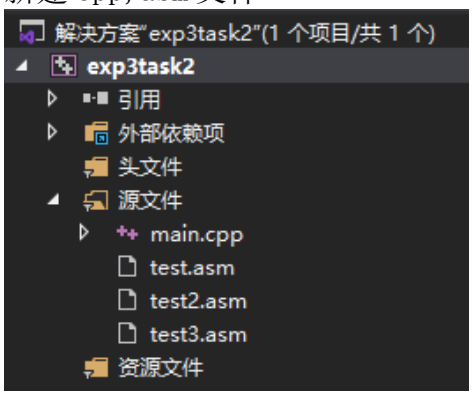
```
      if (status == 3) {
005C5B4C  cmp         dword ptr [ebp-44h],3
005C5B50  jne         main+386h (05C5B96h)
          for (i = 0; i < NUM; i++) {
005C5B52  mov         dword ptr [i],0
005C5B59  jmp         main+354h (05C5B64h)
005C5B5B  mov         eax,dword ptr [i]
005C5B5E  add         eax,1
005C5B61  mov         dword ptr [i],eax
005C5B64  cmp         dword ptr [i],3E8h
005C5B6B  jge         main+372h (05C5B82h)
              calculate(score[i]);
005C5B6D  mov         eax,dword ptr [i]
005C5B70  lea         ecx,score (05CC858h)[eax*8]
005C5B77  push        ecx
005C5B78  call        calculate (05C1348h)
005C5B7D  add         esp,4
          }
005C5B80  jmp         main+34Bh (05C5B5Bh)
          printf("Calculate all the average grades successfully!\n");
005C5B82  push        offset string "Calculate all the average grade@"... (05C7BE0h)
005C5B87  call        _main (05C104Bh)
005C5B8C  add         esp,4
          flag = 2;
005C5B8F  mov         dword ptr [flag],2
      }
```

图 8-1-2

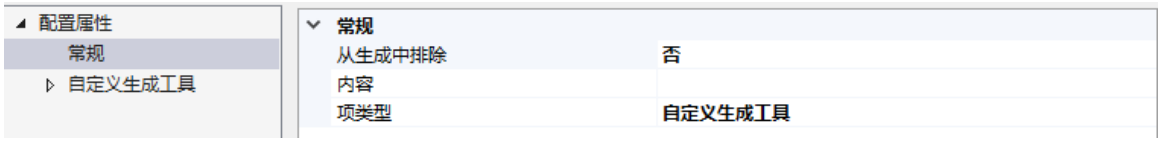
（二）任务二：

本次任务采用 visual studio 2019 环境，配置环境过程如图：

新建 cpp,asm 文件



分别如图配置三个 asm 文件



分别在 cpp 文件和 asm 文件中进行声明

```
extern "C" int calculate(short* s);
extern "C" int continueOrQuit(char* s1);
extern "C" int match(char* s1, char* s2);
```

```
;calculate()函数
.386
.model flat,c
public calculate
calculate proc uses esi edi,s:ptr
```

```
;match函数
.386
.model flat,c
public match
match proc uses esi edi,s1:ptr,s2:ptr
```

```
;continueOrQuit函数
.386
.model flat,c
public continueOrQuit
continueOrQuit proc uses esi,s1:ptr
```

1. 编译后的运行结果：

```
Please input the student's name:
(Enter q to quit and enter a to calculate the average grade)
```

图 1-2

2. 输入“zhangsan”后的结果：

```
Please input the student's name:
(Enter q to quit and enter a to calculate the average grade)
zhangsan
The grade of zhangsan is A !
```

图 2-2

3. 输入“q”后的结果：

```
Please input the student's name:
(Enter q to quit and enter a to calculate the average grade)
q
Exit successfully!
```

图 3-2

4. 输入“lisi”后的结果：

```
Please input the student's name:
(Enter q to quit and enter a to calculate the average grade)
lisi
The grade of lisi is B !
```

图 4-2

5. 输入“xzc”后的结果：

```
Please input the student's name:
(Enter q to quit and enter a to calculate the average grade)
xzc
The grade of xzc is B !
```

图 5-2

6. 输入“wangwu”后的结果：

```
Please input the student's name:
(Enter q to quit and enter a to calculate the average grade)
wangwu
wangwu does not exist!
```

图 6-2

7. 输入“a”后的结果：

```
Please input the student's name:
(Enter q to quit and enter a to calculate the average grade)
a
Calculate all the average grades successfully!
```

图7-2

8. 调试，反汇编后的部分结果：

```
       printf("Please input the student's name:\n");
00BD5A21  push        offset string "Please input the student's name@"... (0BD7B54h)
00BD5A26  call        _main (0BD1046h)
00BD5A2B  add         esp,4
       printf("(Enter q to quit and enter a to calculate the average grade)\n");
00BD5A2E  push        offset string "(Enter q to quit and enter a to@"... (0BD7B7Ch)
00BD5A33  call        _main (0BD1046h)
00BD5A38  add         esp,4
       i = 0;
00BD5A3B  mov         dword ptr [i],0
       c = getchar();
00BD5A42  mov         esi,esp
       c = getchar();
00BD5A44  call        dword ptr [__imp__getchar (0BE0174h)]
00BD5A4A  cmp         esi,esp
00BD5A4C  call        __RTC_CheckEsp (0BD1235h)
00BD5A51  mov         dword ptr [c],eax
       while (c != '\n' && i < N - 1) {
00BD5A54  cmp         dword ptr [c],0Ah
00BD5A58  je          main+309h (0BD5A89h)
00BD5A5A  cmp         dword ptr [i],9
00BD5A5E  jge         main+309h (0BD5A89h)
         in_name[i] = c;
00BD5A60  mov         eax,dword ptr [i]
00BD5A63  mov         cl,byte ptr [c]
00BD5A66  mov         byte ptr in_name (0BDC848h)[eax],cl
         i++;
00BD5A6C  mov         eax,dword ptr [i]
00BD5A6F  add         eax,1
```

图 8-2-1

4. 总结与体会

本次实验分了两个任务，分别是C语言中嵌入汇编指令语句序列的方式实现和C语言程序中调用汇编语言子程序。感觉前者更直接后者更模块化。我了解了visual studio 2019中C语言嵌入汇编指令的方式，主程序输入输出用C语言实现，其余功能用汇编实现，感觉有些地方确实用汇编比较合适，比如在match（）函数。而在输入输出环节，C语言确实比汇编要简单的多。

在任务二中，我了解了在visual studio 2019中配置asm文件，以及在asm中声明的方式。将各个功能化为汇编子程序进行调用易于修改，形式上也更简洁，易读。两次实验都用了visual studio内置的反汇编进行调试，只能看懂一部分的反汇编，一些段、偏移的值等等。

通过调试混合编程的程序，我大致体会了纯粹汇编语言编写的程序与混合编程程序的差异，了解到不同的编程语言是可以协同解决一个问题的，而且可以利用不同语言的特点来更好地解决问题，只是汇编语言的掌握和反汇编调试还是不够熟练，希望以后继续深入汇编，能够利用汇编语言的知识，更好地理解高级语言的内部处理原理与策略，为编写更好的 C 语言序、用好 C 编译器提供支持。