



§. 基础知识题 – 循环结构

要求:

- 1、完成本文档中所有的题目并写出分析、运行结果
- 2、无特殊说明，均使用VS2022编译即可
- 3、直接在本文件上作答，**写出答案/截图（不允许手写、手写拍照截图）**即可；填写答案时，为适应所填内容或贴图，**允许调整**页面的字体大小、颜色、文本框的位置等
 - ★ 贴图要有效部分即可，不需要全部内容
 - ★ 在保证一页一题的前提下，具体页面布局可以自行发挥，简单易读即可
 - ★ **不允许**手写在纸上，再拍照贴图
 - ★ **允许**在各种软件工具上完成（不含手写），再截图贴图
 - ★ 如果某题要求VS+Dev的，则如果两个编译器运行结果一致，贴VS的一张图即可，如果不一致，则两个图都要贴
- 4、转换为pdf后提交
- 5、**4月4日前**网上提交本次作业（在“文档作业”中提交）



§. 基础知识题 – 循环结构

贴图要求：只需要截取输出窗口中的有效部分即可，如果全部截取/截取过大，则视为无效贴图

例：无效贴图

```
Microsoft Visual Studio 调试控制台
Hello, world!
D:\Workspace\VS2019-Demo\Debug\cpp-demo.exe (进程 7484)已退出, 代码为 0。
按任意键关闭此窗口...
```

例：有效贴图

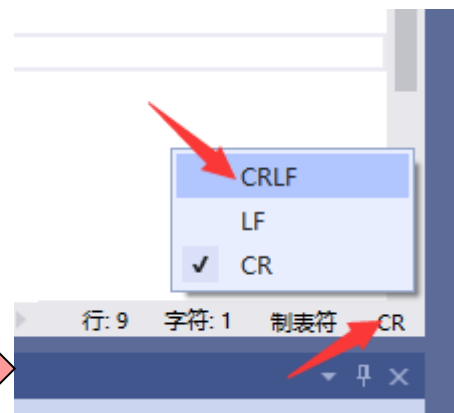
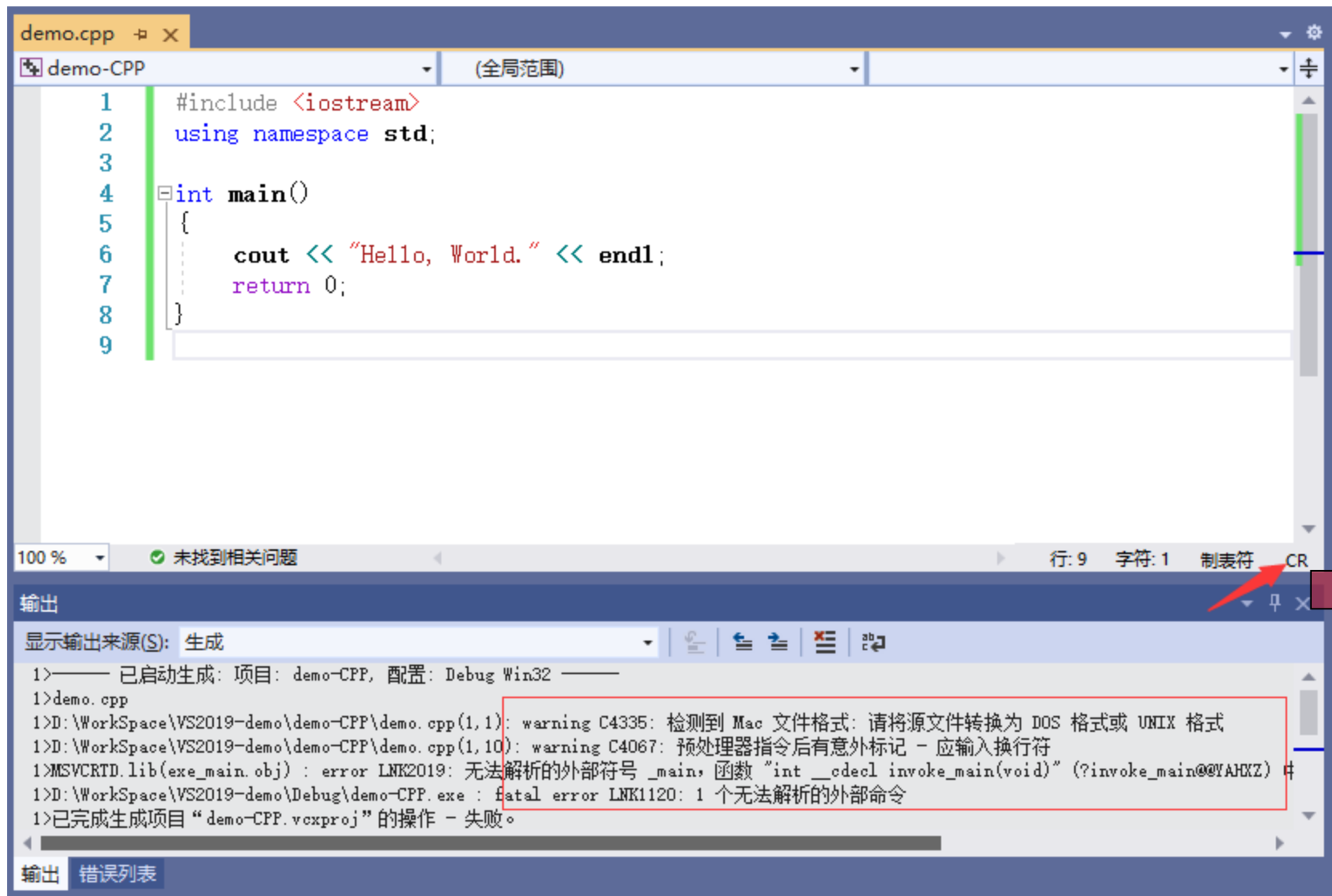
```
Microsoft Visual Studio 调试控制台
Hello, world!
```



§. 基础知识题 – 循环结构

附：用WPS等其他第三方软件打开PPT，将代码复制到VS2022中后，如果出现类似下面的**编译报错**，则观察源程序编辑窗

的右下角是否为CR，如果是，单击CR，在弹出中选择CRLF，再次CTRL+F5运行即可



§. 基础知识题 - 关系运算、逻辑运算与选择结构



特别提示:

- ★ 本次作业的答案，除特别提示外，上课全讲过，课件上都有!!!
- ★ 作业本质就是对上课内容及课件的review(因为读懂程序的逻辑很重要)
- ★ 对上课接受程度较好的同学，可能有点重复/多余，但还得做



§. 基础知识题 – 循环结构

1、循环的嵌套

A. 观察下列程序的运行结果，回答问题并将程序的运行结果截图贴上(如果有错则贴错误信息截图)

```
#include <iostream>
using namespace std;

int main()
{
    int i, j, k;
    int count1 = 0, count2 = 0, count3 = 0;

    for(i=1; i<=100; i++) {
        ++count1;
        for(j=1; j<=100; j++) {
            ++count2;
            for(k=1; k<=100; k++)
                ++count3;
        }
    }

    cout << "count1=" << count1 << endl;
    cout << "count2=" << count2 << endl;
    cout << "count3=" << count3 << endl;
    return 0;
}
```

1、贴运行结果

```
Microsoft Visual Studio 调试控制台
count1=100
count2=10000
count3=1000000
```

2、当循环嵌套时，内层循环的执行次数和外层循环是什么关系？

外层循环一次，内层for循环全部执行完成再进行外层的一次循环。内层循环的执行次数等于外层循环的执行次数乘内层for循环一次全部执行的次数。



§. 基础知识题 – 循环结构

1、循环的嵌套

B. 观察下列程序的运行结果，回答问题并将程序的运行结果截图贴上(如果有错则贴错误信息截图)

```
#include <iostream>
using namespace std;

int main()
{
    int i, j, k;
    int count1 = 0, count2 = 0, count3 = 0;

    for(i=1; i<=100; i++) {
        ++count1;
        for(j=i; j<=100; j++) {
            ++count2;
            for(k=j; k<=100; k++)
                ++count3;
        }
    }

    cout << "count1=" << count1 << endl;
    cout << "count2=" << count2 << endl;
    cout << "count3=" << count3 << endl;
    return 0;
}
```

1、贴运行结果

```
Microsoft Visual Studio 调试控制台
count1=100
count2=5050
count3=171700
```

2、当循环嵌套时，内层循环的执行次数和外层循环是什么关系？
外层循环一次，内层for循环全部执行完成再进行外层的一次循环。

在左侧程序中

第二层循环：当 $i = 1$ 时，循环从1迭代到100，总共执行了100次。
当 $i = 2$ 时，循环从2迭代到100，总共执行了99次。

...

当 $i = 100$ 时，循环从100迭代到100，总共执行了1次。
共计 $100+99+\dots+1=5050$ 次。

第三层循环：当 $i = 1$ 时，第二层循环从1迭代到100，第三层循环总共执行了 $100+99+\dots+1$ 次。
当 $i = 2$ 时，第二层循环从2迭代到100，第三层循环总共执行了 $99+\dots+1$ 次。

...

当 $i = 100$ 时，第二层循环从100迭代到100，第三层循环总共执行了1次。
共计 $5050+4950+\dots+1=171700$ 次。



§. 基础知识题 – 循环结构

1、循环的嵌套

C. 观察下列程序的运行结果，回答问题并将程序的运行结果截图贴上(如果有错则贴错误信息截图)

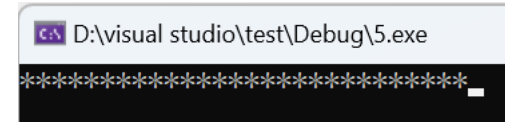
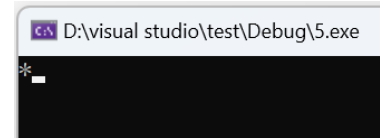
```
#include <iostream>
#include <conio.h>
using namespace std;

int main()
{
    int i, j, count = 0;
    for(i=1; i<=100; i++) {
        for(j=1; i<=100; j++) {
            ++count;
            if (count % 1000 == 0) {
                cout << "*";
                _getch();
            }
        }
    }

    cout << "count = " << count << endl;
    return 0;
}
```

//注意：这个程序无法通过按CTRL+C终止，要关窗口

1、贴运行结果（能表现出要表达的意思即可）



2、按内外for循环的执行步骤依次分析，为什么会得到这个结果？

例：第1步 – 外循环表达式1 – i=1

...

第x步 – 内循环表达式3 – j=4

注：具体内容瞎写的，不要信；步骤写到能得到结论即可

第1步 – 外循环表达式1 – i=1

第2步 – 外循环表达式2 – i<=100

第3步 – 内循环表达式1 – j=1

第4步 – 内循环表达式2 – i<=100

第5步 – 内循环表达式3 – j++

第6步 – 内循环表达式2 – i<=100

第7步 – 内循环内表达式 – if表达式

原因：内层循环的条件始终为真，因为i的值在循环体内不会改变。所以程序陷入无限循环，无论输入多少个值都不会走出循环，并且永远不会执行到打印count值的语句或返回0。

§. 基础知识题 – 循环结构



此页不要删除，也没有意义，仅仅为了分隔题目



§. 基础知识题 – 循环结构

2、break与continue

A. 已知代码如下，回答问题

```
while(1) {  
    ①  
    ②  
    if (X)  
        continue;  
    ③  
    ④  
}
```

当X为真时，重复执行__①②__ (①②③④)
当X为假时，重复执行__①②③④__ (①②③④)

```
for(1; 1; ④) {  
    ①  
    ②  
    if (X)  
        continue;  
    ③  
}
```

当X为真时，重复执行__①②④__ (①②③④)
当X为假时，重复执行__①②③④__ (①②③④)



§. 基础知识题 – 循环结构

2、break与continue

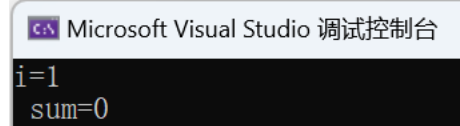
B. 观察下列程序的运行结果，回答问题并将程序的运行结果截图贴上(如果有错则贴错误信息截图)

```
#include <iostream>
using namespace std;
```

```
int main()
{
```

```
    int i=0, sum=0;
```

```
    while(i<1000) {
        i++;
        break;
        sum=sum+i;
    }
```



Microsoft Visual Studio 调试控制台
i=1
sum=0

```
    cout << "i=" << i << endl;
    cout << " sum=" << sum << endl;
```

```
    return 0;
```

```
}
```

//问题1: 循环执行了多少次? **1次**

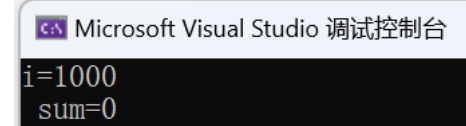
//问题2: sum=sum+i执行了多少次? **0次**

```
#include <iostream>
using namespace std;
```

```
int main()
{
```

```
    int i=0, sum=0;
```

```
    while(i<1000) {
        i++;
        continue;
        sum=sum+i;
    }
```



Microsoft Visual Studio 调试控制台
i=1000
sum=0

```
    cout << "i=" << i << endl;
    cout << " sum=" << sum << endl;
```

```
    return 0;
```

```
}
```

//问题1: 循环执行了多少次? **1000次**

//问题2: sum=sum+i执行了多少次? **0次**

§. 基础知识题 – 循环结构



此页不要删除，也没有意义，仅仅为了分隔题目



§. 基础知识题 – 循环结构

3、观察程序运行结果

A. 观察下列程序的运行结果，回答问题并将程序的运行结果截图贴上(如果有错则贴错误信息截图)

```
#include <iostream>
#include <iomanip>    //格式输出
#include <cmath>      //fabs
#include <windows.h>  //取系统时间
using namespace std;
```

```
int main()
{
```

```
    int s=1;
    double n=1, t=1, pi=0;
```

```
    LARGE_INTEGER tick, begin, end;
    QueryPerformanceFrequency(&tick);    //取计数器频率
    QueryPerformanceCounter(&begin);      //取初始硬件定时器计数
```

```
    while(fabs(t)>1e-6) {
        pi=pi+t;
        n=n+2;
        s=-s;
        t=s/n;
    }
```

```
    QueryPerformanceCounter(&end); //获得终止硬件定时器计数
```

```
    pi=pi*4;
    cout << "n=" << setprecision(10) << n << endl;
    cout << "pi=" << setiosflags(ios::fixed) << setprecision(9) << pi << endl;
```

```
    cout << "计数器频率: " << tick.QuadPart << "Hz" << endl;
    cout << "时钟计数 : " << end.QuadPart - begin.QuadPart << endl;
    cout << setprecision(6) << (end.QuadPart - begin.QuadPart)/double(tick.QuadPart) << "秒" << endl;
```

```
    return 0;
}
```

用下面的迭代公式求Pi的值

$$1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \dots$$

(1) n, t, pi为double型

精度为1e-6: n=_1000001_ pi=_3.141590654_ 时间=_0.001576_(秒)
1e-7: n=_10000001_ pi=_3.141592454_ 时间=_0.016009_(秒)
1e-8: n=_100000001_ pi=_3.141592634_ 时间=_0.160679_(秒)
1e-9: n=_1000000001_ pi=_3.141592652_ 时间=_1.621069_(秒)
(因为机器配置不同, 时间值可能不同)

(2) n, t, pi为float型

精度为1e-6: n=_1000001_ pi=_3.141593933_ 时间=_0.017361_(秒)
1e-7: n=_10000001_ pi=_3.141596556_ 时间=_0.155156_(秒)
1e-8: 无结果

问: 1、7项中哪个没结果? 为什么?

n, t, pi为float型, 精度为1e-8时无结果。因为由于float类型的精度限制, 当t变得非常小时, 它可能无法精确地表示更小的数值。导致fabs(t)始终大于1e-8, 从而使while循环陷入死循环。

2、float和double同进度下那个时间快? (观察现象即可, 不需要解释原因)
double速度更快

本页结果不要截图, 手填即可



§. 基础知识题 – 循环结构

3、观察程序运行结果

B. 观察下列程序的运行结果，回答问题并将程序的运行结果截图贴上(如果有错则贴错误信息截图)

<pre>#include <iostream> #include <iomanip> using namespace std; int main() { int n = 0, i, m, k; bool prime; for (m = 103; m <= 200; m += 2) { prime = true; k = int(sqrt(m)); for (i = 2; i <= k; i++) { if (m % i == 0) { prime = false; break; } } if (prime) { cout << setw(5) << m; n++; if (n % 10 == 0) cout << endl; } } return 0; }</pre>	<h3>打印100-200之间的素数</h3>	(1) 目前输出结果：一共21个，每10个一行
	<div data-bbox="1057 396 2484 628"><p>Microsoft Visual Studio 调试控制台</p><pre>101 103 107 109 113 127 131 137 139 149 151 157 163 167 173 179 181 191 193 197 199</pre></div> <div data-bbox="1057 628 2484 1028"><p>(2) 将m的初值从101改为103，应该是20个，共2行 实际呢？为什么？</p><div data-bbox="1116 799 1967 1025"><p>Microsoft Visual Studio 调试控制台</p><pre>103 107 109 113 127 131 137 139 149 151 157 163 167 173 179 181 191 193 197 199</pre></div><p>实际是20个数，数占两行，两行数中间有若干换行。 因为当m 为153，155的时候，n仍为10，执行了cout << endl;语句，出现两个空行。</p></div> <td data-bbox="2262 394 2491 1358"><p>(3) 将左侧程序改正确 (正确程序贴图在左侧，覆盖现有内容即可)</p></td>	<p>(3) 将左侧程序改正确 (正确程序贴图在左侧，覆盖现有内容即可)</p>

§. 基础知识题 – 循环结构



此页不要删除，也没有意义，仅仅为了分隔题目