

## 程序调试技术第一次作业

PB21051111 吴欣怡

原代码:

```
#include<stdio.h>
#define MAXSIZE 10000
int main()
{
    int j = 0;
    for (float i=0;i < MAXSIZE;i += 0.1)
    {
        j++;
    }
    printf("%d\n", j);
    return 0;
}
```

1. 运行、编译环境: Visual Studio

2. 代码分析:

浮点数的误差问题是二进制的存储问题造成的。

大部分十进制小数都无法用二进制小数完全表示, 一般都是用近似值存储在计算机中。

比如: 0.1 的二进制近似值表示是:

0.10000000000000000055511151231257827021181583404541015625

并不是数学意义上完全的二进制表示。0.1 转换成二进制是一个无限小数。

IEEE 浮点数在运算中采用了四种舍入模式。

一般情况下, 计算机显示的是一个舍入值, 1/10 打印结果: 0.1。

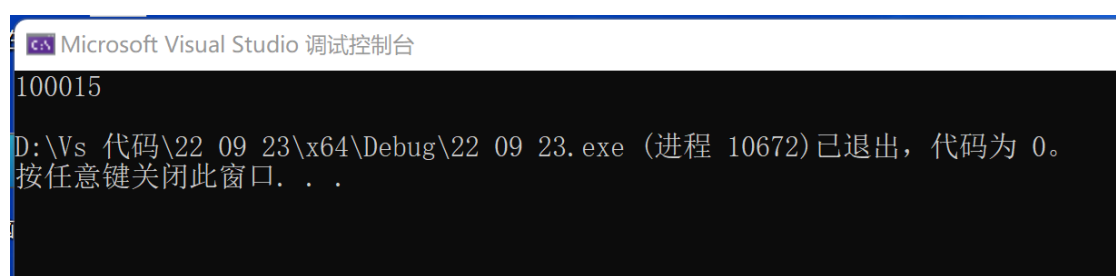
即看上去精确的结果, 其实在计算机中存储的是一个近似值。

同理, 当我们在程序中输入 0.1 时, 被计算机理解得到的也是 0.1 的近似值, 因而在多次 (如 100000 次) 的循环累加后, 就会造成误差。(如下图所示)

名称	值	类型
i	0.100000001	float
j	1	int

值	值
1.700000029	2.399999986
17	24

3. 以下为使用原代码, MAXSIZE 分别取 10000 和 100000 后的输出结果。



Microsoft Visual Studio 调试控制台

990564

D:\Vs 代码\22\_09\_23\x64\Debug\22\_09\_23.exe (进程 11028) 已退出，代码为 0。  
按任意键关闭此窗口. . .

更改 MAXSIZE 的数据，发现大约在第 6000-7000 次循环间，浮点数造成的误差开始使 j 的输出值与逻辑上不符：

```
#include<stdio.h>
#define MAXSIZE 600
int main()
```

Microsoft Visual Studio 调试控制台

6000

```
#include<stdio.h>
#define MAXSIZE 700
int main()
```

Microsoft Visual Studio 调试控制台

7001

若将"float i=0"改为"double i=0"，由于数据更加精确，可以在 MAXSIZE 在 $10^5$ 的数量级上仍保持最终结果的正确性。(但在 $10^6$ 量级上则会输出 1000001，仍无法完全解决问题)

```
#include<stdio.h>
#define MAXSIZE 100000
int main()
{
    int j = 0;
    for (double i=0; i < MAXSIZE; i += 0.1)
    {
```

Microsoft Visual Studio 调试控制台

1000000