# 实验一

**一、实验内容**

1.3 编程观察无穷级数

的求和运算.

1. 采用IEEE单精度浮点数,观察当n为何值时求和结果不再变化,将它与理论分析的结论进行比较（注：在MATLAB中可用single命令将变量转成单精度浮点数）。
2. 用IEEE双精度浮点数计算（1）中前n项的和，评估IEEE单精度浮点数计算结果的误差。
3. 如果采用IEEE双精度浮点数，估计当n为何值时求和结果不再变化，这在当前做实验的计算机上大概需要多长的计算时间？

**二、解题思路**

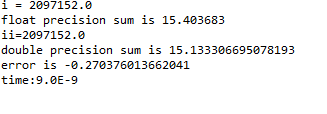
（注：程序均采用Java实现）

1. 对n执行for循环，用两个float变量记录相邻两次的求和结果，当两个变量值相等时，停止循环，输出当前结果，记录当前循环次数。
2. 对n执行（1）中所得循环次数次for循环，用double变量记录最终结果，计算此结果与（1）中求和结果的误差。
3. 调用系统函数clock()，用double变量记录求和结果，记录执行1000000次循环所用时间，然后可以计算出平均一次循环需要的时间。根据（1）中理论停止循环次数，可以得到大概所需计算时间。

**三、算法设计**

|  |
| --- |
| **public** **class** Exp1  {  **public** **static** **void** main(String[] args)  {  **float** i = 0.0f;  **float** s = 0.0f;  **while** (**true**)  {  **float** s1 = s;  i = i + 1.0f;  s += 1.0f / i;  **if**(s == s1)  **break**;  }  System.*out*.println("i = " + i);  System.*out*.println("float precision sum is " + s);  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/    **double** ii = 0.0d;  **double** ss = 0.0d;  **while** (**true**)  {  **if**(ii >= i)  **break**;  ii = ii + 1.0d;  ss += 1.0d/ii;  }  System.*out*.println("ii=" + ii);  System.*out*.println("double precision sum is " + ss);  System.*out*.println("error is " + (ss - s));    **long** start = System.*currentTimeMillis*();  **for** (i=1;i<=1000000;i++)  {  s += 1.0/i;  }  **long** finish = System.*currentTimeMillis*();  **double** time = ((**double**)finish - (**double**)start)/(1000000\*1000);  System.*out*.println("time:"+time);    }  } |

**四、实验结果和结论**



1、单精度浮点数最多有7位十进制有效数字，超出部分会四舍五入。

因此，当。这与程序所得结果2097152基本符合，相对误差为0.0486。

2、单精度计算前2097152项和结果与双精度计算结果相对误差为0.270376。

3、双精度浮点数最多有16位十进制有效数字，因此当。故在实验的计算机上大概需要的计算时间为。