第一次上机作业

问题描述：

分别以单精度和双精度数据类型

- 计算 n=10000 时 $f(n)=\sum_{i=1}^{n}\frac{1}{i^4}$ 的值，分别按从 i=1 到 10000 和 i=10 000 到 1 的顺序计算，对比误差并分析结果。
- 已知 x=100000，分别采用以下公式计算 y，哪个结果更加准确？为什么。

$$y=\ln\left(x-\sqrt{x^2-1}\right) \qquad y=-\ln\left(x+\sqrt{x^2-1}\right)$$

问题分析：

（问题 1）：通过循环算法使得级数做和即可。误差分析方法，将有效位数为 20 位的更精确结果作为真值，通过公式百分误差 = |计算值-真值|/真值 * 100%，可得误差，然后将上述获得的误差进行对比分析。

（问题 2）：直接利用 log 函数和 sqrt 函数计算即可。

Matlab 程序：

问题一：

单精度：

```
format long
f1 = single(0);
f2 = single(0);
for i = single(1):single(1):single(10000)
    f1 = f1 + i^(-4);
end
disp(f1);

for i = single(10000):single(-1):single(1)
    f2 = f2 + i^(-4);
end
disp(f2);
```

结果为：

```
>> computer1
  1.0823221


  1.0823232
```

双精度:
程序和单精度基本相同，将 single 类型换为 double 类型。
结果为:

```
>> computer2
   1.082323233710861


   1.082323233710805
```

问题二:
单精度:

```
-     format long
-     x = single(100000);
-     y1 = single(log(x - sqrt(x^2-1)));
-     y2 = single(-log(x + sqrt(x^2-1)));
-     disp(y1);
-     disp(y2);
```

结果为:

```
>> computer3
  -Inf


 -12.2060728
```

双精度:
程序和单精度基本相同，将 single 类型换为 double 类型。
结果为:

```
>> computer4
 -12.206073762186564


 -12.206072645505174
```

结果分析：

问题一：

利用 vpa（y，20）函数得到有效位数为 20 位更精确结果作为真值
ans=1.0823232337108039669

精度	序号	当前计算值	相对误差
单精度	**从 1 到 10000**	1.0823221	1.047478949532704e-04
	从 10000 到 1	1.0823232	3.114670634070485e-06
双精度	**从 1 到 10000**	1.082323233710861	5.272497317652603e-12
	从 10000 到 1	1.082323233710805	1.025777688259261e-13

从 i=1 加到 i=10000 时，是大数加小数，会损耗掉比较多的有效位数。而从 i=10000 加到 i=1，是小数加大数，误差较前一种方法更小。而双精度比单精度计算有效位数更多，误差更小。

问题二：

第二种方法更准确。因为 100000 相对于 1 来说非常大，第一种计算方法出现了两个相近数字的相减，会发生减性抵消，取入准确位数之后误差较大的部分位数，第二种方法中未出现减性抵消，所以比起第一种方法结果更加精确，
利用计算机直接计算得到-12.2060726455051737。和第二种方法比较接近。

心得：

这次的作业分别从大数加小数和减性抵消两个问题展示了舍入误差在利用 MATLAB 计算时会造成的影响。使我对计算机的有效位数和相对误差有了更深入的了解与直观的认识。在选择算法时要充分规避或减小舍入误差造成的影响，选择更好的算法。