

# Etude 6 - Numbers

Pair Members: Anthony Dickson, Marcus Lee

## 1 Harmonic Numbers

**Definition** The harmonic numbers are defined as the initial partial sums of the series:

$$\frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \dots$$

The above can also be represented as follows:

$$H(n) = \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n-1} + \frac{1}{n}$$

**Hypothesis** We expect that it makes no difference in what order we add terms together.

**Method** Using the above formula, we created a java application which calculates  $H(n)$  using both single and double precision floating point numbers. Our java application computes  $H(n)$  in both order of largest term first ( $\frac{1}{1} + \frac{1}{2} + \dots + \frac{1}{n}$ ) and reverse order ( $\frac{1}{n} + \frac{1}{n-1} + \dots + \frac{1}{1}$ ). The results for calculating  $H(10000)$  are shown below in figure 1.1.

**Figure 1.1: Results of calculating  $H(10000)$ .**

**Largest Term First**

Single Precision: 9.787613

Double Precision: 9.787606036044348

**Reverse order**

Single Precision: 9.787604

Double Precision: 9.787604

**Observations** As we can see in figure 1.1, the results for  $H(10000)$  computed from largest term first are different when using single and double precision numbers. Most notably, the single precision number has about half the number of digits of the double precision number. However, the single and double precision results are the same when  $H(10000)$  was calculated in reverse order. Furthermore, the single precision results and the double precision results are different based on the order that the terms were added together. These observations can be seen in  $H(n)$  for values on either side of 10000 as shown in below in figure 1.2 and figure 1.3.

However, the single and double precision results for reverse order in figure 1.3 show a small difference (0.000001). So perhaps  $H(n)$  calculated in reverse order is infact different for single and double precision numbers (albeit very slightly and perhaps not significantly).

**Figure 1.2: Results of calculating  $H(1000)$ .**

**Largest Term First**

Single Precision: 7.48547843

Double Precision: 7.485470860550343

**Reverse order**

Single Precision: 7.4854717

Double Precision: 7.4854717

**Figure 1.3: Results of calculating  $H(100000)$ .**

**Largest Term First**

Single Precision: 12.090851

Double Precision: 12.090146129863335

**Reverse order**

Single Precision: 12.090153

Double Precision: 12.090152

**Conclusion** Computing harmonic numbers in both single and double precision did not produce answers that agree. The harmonic numbers computed in single precision tended to be slightly bigger than the harmonic numbers computed in double precision, but only had about half the number of digits.

Computing harmonic numbers in reverse order produced results that mostly agree, and also differed to the results that were produced when computing from the largest term first. However, the results that differ only differ

by a very small amount and the difference could be considered to be insignificant.

## 2 Standard Deviation

As per given in the assignment, Standard deviation of a series containing numbers from 1, 2, 3, ,4 ,5 ,6 ,7 ,8 ,9 .....  $(n - 1)$ ,  $n$  is calculated for single precision and double precision data types using Java Code(SDeviation class). It has implemented two methods to find Standard Deviation using two different way given in Assignment. Output of the code is as per following for  $n = 10000$ .

Output:

Method-1 Output

Single Precision: 2886.752

Double Precision: 2886.751331514372

Method-1 Output with added Fixed Value [ 2.0 ]

Single Precision: 2886.7524

Double Precision: 2886.7520243346153

Method-2 Output

Single Precision: 2886.7517

Double Precision: 2886.751331514372

Method-2 Output with added Fixed Value [ 2.0 ]

Single Precision: 2886.7515

Double Precision: 2886.751331514372

Based on following output, for standard deviation precision matters. For small number in the array standard deviation doesn't varie much. But it varies with difference of 1.00 or more for single and double precision data types. This code also includes scenario where a Fixed value is added to every member of the series and calulated standard deviation. It doesnt vary in case of addition of same fixed value to every member in the series respective to precisions. Method-2 would be more precise out of given two in case of standard deviation calculation.

### 3 An Identity

For identity task, Java code (Identity class) has considered given equation and calculated output for single precision and double precision data types. Output for same is as per below.

Output:

Single Precision Input  $x = 2.5$   $y = 3.5$

Single Precision Left Side: 2.5 Right Side: 2.5 Result: *true*

Double Precision Input  $x = 2.5848$   $y = 3.5234$

Double Precision Left Side: 2.5848 Right Side: 2.584800000000005 Result: *false*

Single Precision Input  $x = 2.51$   $y = 3.5$

Single Precision Left Side: 2.51 Right Side: 2.5100002 Result: *false* Double Precision Input  $x = 2.5848$   $y = 3.5234$

Double Precision Left Side: 2.5848 Right Side: 2.584800000000005 Result: *false*

As per the above output for different inputs, Single precision till single decimal points satisfy the equation. if there are more than one decimals after points then even for single precision point equations fails even though it is mathematically correct. In case of Double precision it fails every time irrespective of given input.