CS 3320 – Numerical Software Development

Program 1

Calculate ulps of Two Numbers

Write a function, **ulps(x,y)**, that takes two floating point parameters, **x** and **y**, and returns the number of *ulps* (floating-point intervals) between **x** and **y**. You will of course need to take into account that **x** and **y** may have different exponents of the floating-point base in their representation. Also, do not assume that **x <= y** (handle either case). Your function may assume that its parameter will be the floating-point type. Do *not* use logarithms to find the exponent of your floating-point numbers – use repeated multiplication/division following the pattern discussed in class. Your function should handle negative numbers and numbers less than 1.

Your function will return infinity if the input parameters have the following properties:

- Opposite in signs, or
- Either one of them is zero, or
- Either one of them is either positive infinity or negative infinity.

If the input parameters are both negative, convert them to be positive numbers by taking the absolute value. Your algorithm only needs to work with two positive floating-point numbers.

The following code segment shows how to use `math` and `sys` modules to get the base, infinity (`inf`), machine epsilon (`eps`), and mantissa digits (`prec`) in Python.

```
import sys
import math
base = sys.float_info.radix
eps = sys.float_info.epsilon
prec = sys.float_info.mant_dig
inf = math.inf
```

Algorithm analysis:

1. Check the input parameters for special conditions.
2. Find the exponents for both input parameters in the machine base (`base`).
   For example: Find `exp` such that $(base)^{exp} \leq a < (base)^{exp+1}$.
3. Examine the `exp` for both parameter:
   a. If they are the same: count the intervals between them. (Remember that the spacing is $\varepsilon \cdot B^{e}$ for each interval $[B^{e}, B^{e+1}]$)
   b. If they differ by one: add the intervals from the smaller number to B^{exp+1} to the intervals from B^{exp+1} to the larger number.
   c. If they differ by more than one: In addition to the number of intervals in part b, add the numbers of intervals in the exponent ranges in between the exponents of these two numbers.

Verify your algorithm with the following inputs with my answers which are listed after each call:

```
    print(ulps(-1.0, -1.0000000000000003)) //1
    print(ulps(1.0, 1.0000000000000003))   //1
    print(ulps(1.0, 1.0000000000000004))   //2
    print(ulps(1.0, 1.0000000000000005))   //2
    print(ulps(1.0, 1.0000000000000006))   //3
    print(ulps(0.9999999999999999, 1.0))   //1
    print(ulps(0.4999999999999995, 2.0)) //9007199254741001
    print(ulps(0.5000000000000005, 2.0)) //9007199254740987
    print(ulps(0.5, 2.0)) //9007199254740992
    print(ulps(1.0, 2.0)) //4503599627370496
    print(2.0**52)        // 4503599627370496.0
    print(ulps(-1.0, 1.0) //inf
    print(ulps(-1.0, 0.0) //inf
    print(ulps(0.0, 1.0)  //inf
    print(ulps(5.0, math.inf) //inf
    print(ulps(15.0, 100.0))  // 12103423998558208
```

Submission:
Submit your source code and a screenshot of your program outputs.

Python is the preferred language of this class. If you choose a different language, please work with the course IA to ensure that your submission can be executed with his environment and verify your results.