CSC 406: Computer System I, 2019 Winter,

Assignment #2

Purpose:

To go over:

- Integer and bitwise operations
- The IEEE floating point format
- A *little* more on pointers

Assignment

Define the following constants:

SIGN_SHIFT	how many bits to shift the sign field from the least signficant position to where the exponent bit field belongs.
EXPONENT_SHIFT	how many bits to shift the exponent bit field from the least signficant position to where the exponent bit field belongs.
MANTISSA_MASK	the mask to only keep the mantissa bit field.
MANTISSA_HIDDEN_BIT	the hidden bit in its proper position
MANTISSA_SHIFT	How many bits to shift the mantissa bit field from the least signficant position to where the mantissa bit field belongs.

Use those constants, and the others given, to finish the following functions:

`int isPositive(float number)`	Returns '1' if 'number' is positive, or '0' otherwise.
`int obtainExponent(float number)`	Returns the power-of-2 exponent of 'number'.
`int obtainMantissa(float number)`	Returns the mantissa of 'number'.

`int isZero(float number)`	Returns return '1' if 'number' is +0.0 or -0.0, or false otherwise.

Copy and paste the following:

```
/*-------------------------------------------------------------------
------*
 *---
       ---*
 *---          floatCompare.c
       ---*
 *---
       ---*
 *---       This file defines a program that supports the function
       ---*
 *---   isLessThan(float lhs,float rhs).  It compares 'lhs' and 'rhs'
       ---*
 *---   and implements the floating point '<' operator, but does so
       ---*
 *---   with only integer and bitwise operations.
       ---*
 *---
       ---*
 *---   ----    ----    ----    ----    ----    ----    ----    ----
       ---*
 *---
       ---*
 *---   Version 2a                                   Joseph Phillips
       ---*
 *---
       ---*
 *-------------------------------------------------------------------
------*/

//      Compile with:
//      $ gcc floatCompare.c -o floatCompare -std=c99 -lm

#include       <stdlib.h>
```

```
#include       <stdio.h>
#include       <float.h>
#include       <math.h>


//  PURPOSE:  To define a nickname for type 'unsigned int'.
typedef unsigned int                  uInt;


//--                  Sign related constants                --//

//  PURPOSE:  To tell how many bits to shift the sign field from the
//      least signficant position to where the exponent bit field
belongs.
const   uInt    SIGN_SHIFT            = 0;    // CHANGE THAT 0!


//  PURPOSE:  To be the mask to only keep the sign bit field.
#define        SIGN_MASK              (uInt)(0x1 << SIGN_SHIFT)


//--                  Exponent related constants
       --//

//  PURPOSE:  To tell how many bits to shift the exponent bit field
from the
//      least signficant position to where the exponent bit field
belongs.
const   uInt    EXPONENT_SHIFT        = 0;    // CHANGE THAT 0!


//  PURPOSE:  To be the mask to only keep the exponent bit field.
#define        EXPONENT_MASK          (uInt)(0xFF << EXPONENT_SHIFT)


//  PURPOSE:  To tell the exponent bit pattern for denormalized
numbers.
const   uInt    EXPONENT_DENORMALIZED_BIT_PATTERN
```

```
                                        = 0x00;

//  PURPOSE:  To tell the exponent value (*not* the bit pattern
value) that
//      signifies a denormalized number.
const   int     DENORMALIZE_EXPONENT    = -127;


//  PURPOSE:  To tell the 'bias' of the exponents:
//      exponent = (bit field number) - 'bias'.
const   uInt    EXPONENT_BIAS           = 0x7F;


//--                    Mantissa related constants
        --//

//  PURPOSE:  To tell the mask to only keep the mantissa bit field.
const   uInt    MANTISSA_MASK           = 0;    // CHANGE THAT 0!


//  PURPOSE:  To tell give the hidden bit in its proper position.
const   uInt    MANTISSA_HIDDEN_BIT     = 0;    // CHANGE THAT 0!


//  PURPOSE:  To tell how many bits to shift the mantissa bit field
from the
//      least signficant position to where the mantissa bit field
belongs.
const   uInt    MANTISSA_SHIFT          = 0;    // PERHAPS CHANGE THAT
0?

//  PURPOSE:  To tell how many mantissa bits there are (including
hidden bit)
const   uInt    NUM_MANTISSA_BITS       = 24;


//--                    Miscellaneous related constants
        --//
```

```
//  PURPOSE:  To give the maximum length of C-strings.
const   uInt    TEXT_LEN                = 64;


//--                          Functions:
       --//

//  PURPOSE:  To return '1' if 'number' is positive, or '0'
otherwise.
int             isPositive      (float          number
                                )
{
  //  Make an integer with the same bit pattern as number
  //  If that integer has SIGN_MASK on then return '0', otherwise '1'
}


//  PURPOSE:  To return the power-of-2 exponent of 'number'.
int             obtainExponent  (float          number
                                )
{
  //  Make an integer with the same bit pattern as number
  //  Use EXPONENT_MASK to only keep only the exponent bits.
  //  Shift the result over by EXPONENT_SHIFT
  //  Subtract EXPONENT_BIAS from the shifted result, return() that
value
}


//  PURPOSE:  To return the mantissa of 'number'.
int             obtainMantissa  (float          number
                                )
{
  //  Make an integer with the same bit pattern as number
  //  Use MANTISSA_MASK to only keep only the exponent bits.
  //  Shift the result over by MANTISSA_SHIFT
```

```
  //  Only if (obtainExponent(number) != DENORMALIZE_EXPONENT) then
turn the MANTISSA_HIDDEN_BIT on
}


//  PURPOSE:  To return '1' if 'number' is +0.0 or -0.0, or false
otherwise.
int             isZero          (float          number
                                )
{
  //  If (obtainExponent(number) == DENORMALIZE_EXPONENT) and
  //     (obtainMantissa(number) == 0x00) then return 1, otherwise 0.
}


//  PURPOSE:  To return '1' if 'lhs' is computed as being less than
'rhs' or
//      '0' otherwise.
int             isLessThan      (float          lhs,
                                 float          rhs
                                )
{
  //  I.  Application validity check:

  //  II.  Compare numbers:
  //  II.A.  Handle zero as a special case:
  if  ( isZero(lhs) && isZero(rhs) )
    return(0);

  //  II.B.  Compare signs:
  int   isLhsPos        = isPositive(lhs);
  int   isRhsPos        = isPositive(rhs);

  if  (isLhsPos && !isRhsPos)
    return(0);

  if  (!isLhsPos && isRhsPos)
    return(1);
```

```
  //  II.C.  Compare exponents:
  int   lhsExp         = obtainExponent(lhs);
  int   rhsExp         = obtainExponent(rhs);

  if  (lhsExp > rhsExp)
    return( isLhsPos ? 0 : 1 );

  if  (lhsExp < rhsExp)
    return( isLhsPos ? 1 : 0 );

  //  II.D.  Compare mantissas:
  int   lhsMant        = obtainMantissa(lhs);
  int   rhsMant        = obtainMantissa(rhs);

  if  (lhsMant > rhsMant)
    return( isLhsPos ? 0 : 1 );

  if  (lhsMant < rhsMant)
    return( isLhsPos ? 1 : 0 );

  //  III.  Finished, if get here they are equal:
  return(0);
}


//  PURPOSE:  To compare several floating point numbers with
'isLessThan()',
//      and to tell how accurate the function is.  Ignores arguments.
Returns
//      'EXIT_SUCCESS' to OS.
int             main            ()
{
  //  I.  Application validity check:

  //  II.  Do comparisons:
  //  II.A.  Define numbers to compare:
  const float  NUMBER_ARRAY[] = {-INFINITY,
```

```
                                    -FLT_MAX,
                                    -1e+10,
                                    -1.0,
                                    -1e-10,
                                    -1.17549e-38,
                                    -5.87747e-39,
                                    -2.93874e-39,
                                    -1.4013e-45,
                                    -0.0,
                                    +0.0,
                                    +1.4013e-45,
                                    +2.93874e-39,
                                    +5.87747e-39,
                                    +1.17549e-38,
                                    +1e-10,
                                    +1.0,
                                    +1e+10,
                                    +FLT_MAX,
                                    +INFINITY
                                   };
  const int    NUM_NUMS       = sizeof(NUMBER_ARRAY) /
sizeof(float);

  //  II.B.  Decompose all numbers into sign, exponent and mantissa:
  int          index;

  for  (index = 0;  index < NUM_NUMS;  index++)
  {
    float      number  = NUMBER_ARRAY[index];
    int        isPos   = isPositive(number);
    int        exponent= obtainExponent(number);
    int        mantissa= obtainMantissa(number);

    float      reconstitute
                       = (exponent == DENORMALIZE_EXPONENT)
                         ? pow(2.0,exponent+1) *
                           ((double)(mantissa))/MANTISSA_HIDDEN_BIT
                         : pow(2.0,exponent) *
```

```
                        ((double)(mantissa))/MANTISSA_HIDDEN_BIT;

    if  (!isPos)
      reconstitute     = -reconstitute;

    printf("%12g: %c\t%4d\t0x%06X\t%g\n",
           number,
           isPositive(number) ? '+' : '-',
           exponent,
           mantissa,
           reconstitute
          );
  }

  //  II.C.  Do comparisons:
  int           indexOut;
  int           indexIn;

  for  (indexOut = 0;  indexOut < NUM_NUMS;  indexOut++)
  {
    float       lhs     = NUMBER_ARRAY[indexOut];

    for  (indexIn = 0;  indexIn < NUM_NUMS;  indexIn++)
    {
      float     rhs     = NUMBER_ARRAY[indexIn];
      int       result = isLessThan(lhs,rhs);

      printf("isLessThan(%12g,%12g) == %5s (%s)\n",lhs,rhs,
             (result ? "true" : "false"),
             ( (result == (lhs < rhs)) ? "correct" : "WRONG")
            );
    }
  }

  //  III.  Finished:
  return(EXIT_SUCCESS);
}
```

Proper output:

```
$ ./floatCompare
isLessThan(        -inf,        -inf) == false (correct)
isLessThan(        -inf,-3.40282e+38) ==  true (correct)
isLessThan(        -inf,      -1e+10) ==  true (correct)
isLessThan(        -inf,      -1e-10) ==  true (correct)
isLessThan(        -inf,-1.17549e-38) ==  true (correct)
isLessThan(        -inf,-5.87747e-39) ==  true (correct)
isLessThan(        -inf,-2.93874e-39) ==  true (correct)
isLessThan(        -inf, -1.4013e-45) ==  true (correct)
isLessThan(        -inf,          -0) ==  true (correct)
isLessThan(        -inf,           0) ==  true (correct)
isLessThan(        -inf,  1.4013e-45) ==  true (correct)
isLessThan(        -inf, 2.93874e-39) ==  true (correct)
isLessThan(        -inf, 5.87747e-39) ==  true (correct)
isLessThan(        -inf, 1.17549e-38) ==  true (correct)
isLessThan(        -inf,       1e-10) ==  true (correct)
isLessThan(        -inf,           1) ==  true (correct)
isLessThan(        -inf,       1e+10) ==  true (correct)
isLessThan(        -inf, 3.40282e+38) ==  true (correct)
isLessThan(        -inf,         inf) ==  true (correct)
isLessThan(-3.40282e+38,        -inf) == false (correct)
isLessThan(-3.40282e+38,-3.40282e+38) == false (correct)
isLessThan(-3.40282e+38,      -1e+10) ==  true (correct)
isLessThan(-3.40282e+38,      -1e-10) ==  true (correct)
isLessThan(-3.40282e+38,-1.17549e-38) ==  true (correct)
isLessThan(-3.40282e+38,-5.87747e-39) ==  true (correct)
isLessThan(-3.40282e+38,-2.93874e-39) ==  true (correct)
isLessThan(-3.40282e+38, -1.4013e-45) ==  true (correct)
isLessThan(-3.40282e+38,          -0) ==  true (correct)
isLessThan(-3.40282e+38,           0) ==  true (correct)
isLessThan(-3.40282e+38,  1.4013e-45) ==  true (correct)
isLessThan(-3.40282e+38, 2.93874e-39) ==  true (correct)
isLessThan(-3.40282e+38, 5.87747e-39) ==  true (correct)
isLessThan(-3.40282e+38, 1.17549e-38) ==  true (correct)
isLessThan(-3.40282e+38,       1e-10) ==  true (correct)
isLessThan(-3.40282e+38,           1) ==  true (correct)
isLessThan(-3.40282e+38,       1e+10) ==  true (correct)
```

```
isLessThan(-3.40282e+38, 3.40282e+38) ==  true (correct)
isLessThan(-3.40282e+38,         inf) ==  true (correct)
isLessThan(      -1e+10,        -inf) == false (correct)
isLessThan(      -1e+10,-3.40282e+38) == false (correct)
isLessThan(      -1e+10,      -1e+10) == false (correct)
isLessThan(      -1e+10,      -1e-10) ==  true (correct)
isLessThan(      -1e+10,-1.17549e-38) ==  true (correct)
isLessThan(      -1e+10,-5.87747e-39) ==  true (correct)
isLessThan(      -1e+10,-2.93874e-39) ==  true (correct)
isLessThan(      -1e+10, -1.4013e-45) ==  true (correct)
isLessThan(      -1e+10,          -0) ==  true (correct)
isLessThan(      -1e+10,           0) ==  true (correct)
isLessThan(      -1e+10,  1.4013e-45) ==  true (correct)
isLessThan(      -1e+10, 2.93874e-39) ==  true (correct)
isLessThan(      -1e+10, 5.87747e-39) ==  true (correct)
isLessThan(      -1e+10, 1.17549e-38) ==  true (correct)
isLessThan(      -1e+10,       1e-10) ==  true (correct)
isLessThan(      -1e+10,           1) ==  true (correct)
isLessThan(      -1e+10,       1e+10) ==  true (correct)
isLessThan(      -1e+10, 3.40282e+38) ==  true (correct)
isLessThan(      -1e+10,         inf) ==  true (correct)
isLessThan(      -1e-10,        -inf) == false (correct)
isLessThan(      -1e-10,-3.40282e+38) == false (correct)
isLessThan(      -1e-10,      -1e+10) == false (correct)
isLessThan(      -1e-10,      -1e-10) == false (correct)
isLessThan(      -1e-10,-1.17549e-38) ==  true (correct)
isLessThan(      -1e-10,-5.87747e-39) ==  true (correct)
isLessThan(      -1e-10,-2.93874e-39) ==  true (correct)
isLessThan(      -1e-10, -1.4013e-45) ==  true (correct)
isLessThan(      -1e-10,          -0) ==  true (correct)
isLessThan(      -1e-10,           0) ==  true (correct)
isLessThan(      -1e-10,  1.4013e-45) ==  true (correct)
isLessThan(      -1e-10, 2.93874e-39) ==  true (correct)
isLessThan(      -1e-10, 5.87747e-39) ==  true (correct)
isLessThan(      -1e-10, 1.17549e-38) ==  true (correct)
isLessThan(      -1e-10,       1e-10) ==  true (correct)
isLessThan(      -1e-10,           1) ==  true (correct)
isLessThan(      -1e-10,       1e+10) ==  true (correct)
```

```
isLessThan(      -1e-10, 3.40282e+38) ==  true (correct)
isLessThan(      -1e-10,         inf) ==  true (correct)
isLessThan(-1.17549e-38,        -inf) == false (correct)
isLessThan(-1.17549e-38,-3.40282e+38) == false (correct)
isLessThan(-1.17549e-38,      -1e+10) == false (correct)
isLessThan(-1.17549e-38,      -1e-10) == false (correct)
isLessThan(-1.17549e-38,-1.17549e-38) == false (correct)
isLessThan(-1.17549e-38,-5.87747e-39) ==  true (correct)
isLessThan(-1.17549e-38,-2.93874e-39) ==  true (correct)
isLessThan(-1.17549e-38, -1.4013e-45) ==  true (correct)
isLessThan(-1.17549e-38,          -0) ==  true (correct)
isLessThan(-1.17549e-38,           0) ==  true (correct)
isLessThan(-1.17549e-38,  1.4013e-45) ==  true (correct)
isLessThan(-1.17549e-38, 2.93874e-39) ==  true (correct)
isLessThan(-1.17549e-38, 5.87747e-39) ==  true (correct)
isLessThan(-1.17549e-38, 1.17549e-38) ==  true (correct)
isLessThan(-1.17549e-38,       1e-10) ==  true (correct)
isLessThan(-1.17549e-38,           1) ==  true (correct)
isLessThan(-1.17549e-38,       1e+10) ==  true (correct)
isLessThan(-1.17549e-38, 3.40282e+38) ==  true (correct)
isLessThan(-1.17549e-38,         inf) ==  true (correct)
isLessThan(-5.87747e-39,        -inf) == false (correct)
isLessThan(-5.87747e-39,-3.40282e+38) == false (correct)
isLessThan(-5.87747e-39,      -1e+10) == false (correct)
isLessThan(-5.87747e-39,      -1e-10) == false (correct)
isLessThan(-5.87747e-39,-1.17549e-38) == false (correct)
isLessThan(-5.87747e-39,-5.87747e-39) == false (correct)
isLessThan(-5.87747e-39,-2.93874e-39) ==  true (correct)
isLessThan(-5.87747e-39, -1.4013e-45) ==  true (correct)
isLessThan(-5.87747e-39,          -0) ==  true (correct)
isLessThan(-5.87747e-39,           0) ==  true (correct)
isLessThan(-5.87747e-39,  1.4013e-45) ==  true (correct)
isLessThan(-5.87747e-39, 2.93874e-39) ==  true (correct)
isLessThan(-5.87747e-39, 5.87747e-39) ==  true (correct)
isLessThan(-5.87747e-39, 1.17549e-38) ==  true (correct)
isLessThan(-5.87747e-39,       1e-10) ==  true (correct)
isLessThan(-5.87747e-39,           1) ==  true (correct)
isLessThan(-5.87747e-39,       1e+10) ==  true (correct)
```

```
isLessThan(-5.87747e-39, 3.40282e+38) ==  true (correct)
isLessThan(-5.87747e-39,         inf) ==  true (correct)
isLessThan(-2.93874e-39,        -inf) == false (correct)
isLessThan(-2.93874e-39,-3.40282e+38) == false (correct)
isLessThan(-2.93874e-39,      -1e+10) == false (correct)
isLessThan(-2.93874e-39,      -1e-10) == false (correct)
isLessThan(-2.93874e-39,-1.17549e-38) == false (correct)
isLessThan(-2.93874e-39,-5.87747e-39) == false (correct)
isLessThan(-2.93874e-39,-2.93874e-39) == false (correct)
isLessThan(-2.93874e-39, -1.4013e-45) ==  true (correct)
isLessThan(-2.93874e-39,          -0) ==  true (correct)
isLessThan(-2.93874e-39,           0) ==  true (correct)
isLessThan(-2.93874e-39,  1.4013e-45) ==  true (correct)
isLessThan(-2.93874e-39, 2.93874e-39) ==  true (correct)
isLessThan(-2.93874e-39, 5.87747e-39) ==  true (correct)
isLessThan(-2.93874e-39, 1.17549e-38) ==  true (correct)
isLessThan(-2.93874e-39,       1e-10) ==  true (correct)
isLessThan(-2.93874e-39,           1) ==  true (correct)
isLessThan(-2.93874e-39,       1e+10) ==  true (correct)
isLessThan(-2.93874e-39, 3.40282e+38) ==  true (correct)
isLessThan(-2.93874e-39,         inf) ==  true (correct)
isLessThan( -1.4013e-45,        -inf) == false (correct)
isLessThan( -1.4013e-45,-3.40282e+38) == false (correct)
isLessThan( -1.4013e-45,      -1e+10) == false (correct)
isLessThan( -1.4013e-45,      -1e-10) == false (correct)
isLessThan( -1.4013e-45,-1.17549e-38) == false (correct)
isLessThan( -1.4013e-45,-5.87747e-39) == false (correct)
isLessThan( -1.4013e-45,-2.93874e-39) == false (correct)
isLessThan( -1.4013e-45, -1.4013e-45) == false (correct)
isLessThan( -1.4013e-45,          -0) ==  true (correct)
isLessThan( -1.4013e-45,           0) ==  true (correct)
isLessThan( -1.4013e-45,  1.4013e-45) ==  true (correct)
isLessThan( -1.4013e-45, 2.93874e-39) ==  true (correct)
isLessThan( -1.4013e-45, 5.87747e-39) ==  true (correct)
isLessThan( -1.4013e-45, 1.17549e-38) ==  true (correct)
isLessThan( -1.4013e-45,       1e-10) ==  true (correct)
isLessThan( -1.4013e-45,           1) ==  true (correct)
isLessThan( -1.4013e-45,       1e+10) ==  true (correct)
```

```
isLessThan( -1.4013e-45, 3.40282e+38) ==  true (correct)
isLessThan( -1.4013e-45,         inf) ==  true (correct)
isLessThan(          -0,        -inf) == false (correct)
isLessThan(          -0,-3.40282e+38) == false (correct)
isLessThan(          -0,      -1e+10) == false (correct)
isLessThan(          -0,      -1e-10) == false (correct)
isLessThan(          -0,-1.17549e-38) == false (correct)
isLessThan(          -0,-5.87747e-39) == false (correct)
isLessThan(          -0,-2.93874e-39) == false (correct)
isLessThan(          -0, -1.4013e-45) == false (correct)
isLessThan(          -0,          -0) == false (correct)
isLessThan(          -0,           0) == false (correct)
isLessThan(          -0,  1.4013e-45) ==  true (correct)
isLessThan(          -0, 2.93874e-39) ==  true (correct)
isLessThan(          -0, 5.87747e-39) ==  true (correct)
isLessThan(          -0, 1.17549e-38) ==  true (correct)
isLessThan(          -0,       1e-10) ==  true (correct)
isLessThan(          -0,           1) ==  true (correct)
isLessThan(          -0,       1e+10) ==  true (correct)
isLessThan(          -0, 3.40282e+38) ==  true (correct)
isLessThan(          -0,         inf) ==  true (correct)
isLessThan(           0,        -inf) == false (correct)
isLessThan(           0,-3.40282e+38) == false (correct)
isLessThan(           0,      -1e+10) == false (correct)
isLessThan(           0,      -1e-10) == false (correct)
isLessThan(           0,-1.17549e-38) == false (correct)
isLessThan(           0,-5.87747e-39) == false (correct)
isLessThan(           0,-2.93874e-39) == false (correct)
isLessThan(           0, -1.4013e-45) == false (correct)
isLessThan(           0,          -0) == false (correct)
isLessThan(           0,           0) == false (correct)
isLessThan(           0,  1.4013e-45) ==  true (correct)
isLessThan(           0, 2.93874e-39) ==  true (correct)
isLessThan(           0, 5.87747e-39) ==  true (correct)
isLessThan(           0, 1.17549e-38) ==  true (correct)
isLessThan(           0,       1e-10) ==  true (correct)
isLessThan(           0,           1) ==  true (correct)
isLessThan(           0,       1e+10) ==  true (correct)
```

```
isLessThan(           0, 3.40282e+38) ==  true (correct)
isLessThan(           0,         inf) ==  true (correct)
isLessThan(  1.4013e-45,        -inf) == false (correct)
isLessThan(  1.4013e-45,-3.40282e+38) == false (correct)
isLessThan(  1.4013e-45,      -1e+10) == false (correct)
isLessThan(  1.4013e-45,      -1e-10) == false (correct)
isLessThan(  1.4013e-45,-1.17549e-38) == false (correct)
isLessThan(  1.4013e-45,-5.87747e-39) == false (correct)
isLessThan(  1.4013e-45,-2.93874e-39) == false (correct)
isLessThan(  1.4013e-45, -1.4013e-45) == false (correct)
isLessThan(  1.4013e-45,          -0) == false (correct)
isLessThan(  1.4013e-45,           0) == false (correct)
isLessThan(  1.4013e-45,  1.4013e-45) == false (correct)
isLessThan(  1.4013e-45, 2.93874e-39) ==  true (correct)
isLessThan(  1.4013e-45, 5.87747e-39) ==  true (correct)
isLessThan(  1.4013e-45, 1.17549e-38) ==  true (correct)
isLessThan(  1.4013e-45,       1e-10) ==  true (correct)
isLessThan(  1.4013e-45,           1) ==  true (correct)
isLessThan(  1.4013e-45,       1e+10) ==  true (correct)
isLessThan(  1.4013e-45, 3.40282e+38) ==  true (correct)
isLessThan(  1.4013e-45,         inf) ==  true (correct)
isLessThan( 2.93874e-39,        -inf) == false (correct)
isLessThan( 2.93874e-39,-3.40282e+38) == false (correct)
isLessThan( 2.93874e-39,      -1e+10) == false (correct)
isLessThan( 2.93874e-39,      -1e-10) == false (correct)
isLessThan( 2.93874e-39,-1.17549e-38) == false (correct)
isLessThan( 2.93874e-39,-5.87747e-39) == false (correct)
isLessThan( 2.93874e-39,-2.93874e-39) == false (correct)
isLessThan( 2.93874e-39, -1.4013e-45) == false (correct)
isLessThan( 2.93874e-39,          -0) == false (correct)
isLessThan( 2.93874e-39,           0) == false (correct)
isLessThan( 2.93874e-39,  1.4013e-45) == false (correct)
isLessThan( 2.93874e-39, 2.93874e-39) == false (correct)
isLessThan( 2.93874e-39, 5.87747e-39) ==  true (correct)
isLessThan( 2.93874e-39, 1.17549e-38) ==  true (correct)
isLessThan( 2.93874e-39,       1e-10) ==  true (correct)
isLessThan( 2.93874e-39,           1) ==  true (correct)
isLessThan( 2.93874e-39,       1e+10) ==  true (correct)
```

```
isLessThan( 2.93874e-39, 3.40282e+38) ==  true (correct)
isLessThan( 2.93874e-39,         inf) ==  true (correct)
isLessThan( 5.87747e-39,        -inf) == false (correct)
isLessThan( 5.87747e-39,-3.40282e+38) == false (correct)
isLessThan( 5.87747e-39,      -1e+10) == false (correct)
isLessThan( 5.87747e-39,      -1e-10) == false (correct)
isLessThan( 5.87747e-39,-1.17549e-38) == false (correct)
isLessThan( 5.87747e-39,-5.87747e-39) == false (correct)
isLessThan( 5.87747e-39,-2.93874e-39) == false (correct)
isLessThan( 5.87747e-39, -1.4013e-45) == false (correct)
isLessThan( 5.87747e-39,          -0) == false (correct)
isLessThan( 5.87747e-39,           0) == false (correct)
isLessThan( 5.87747e-39,  1.4013e-45) == false (correct)
isLessThan( 5.87747e-39, 2.93874e-39) == false (correct)
isLessThan( 5.87747e-39, 5.87747e-39) == false (correct)
isLessThan( 5.87747e-39, 1.17549e-38) ==  true (correct)
isLessThan( 5.87747e-39,       1e-10) ==  true (correct)
isLessThan( 5.87747e-39,           1) ==  true (correct)
isLessThan( 5.87747e-39,       1e+10) ==  true (correct)
isLessThan( 5.87747e-39, 3.40282e+38) ==  true (correct)
isLessThan( 5.87747e-39,         inf) ==  true (correct)
isLessThan( 1.17549e-38,        -inf) == false (correct)
isLessThan( 1.17549e-38,-3.40282e+38) == false (correct)
isLessThan( 1.17549e-38,      -1e+10) == false (correct)
isLessThan( 1.17549e-38,      -1e-10) == false (correct)
isLessThan( 1.17549e-38,-1.17549e-38) == false (correct)
isLessThan( 1.17549e-38,-5.87747e-39) == false (correct)
isLessThan( 1.17549e-38,-2.93874e-39) == false (correct)
isLessThan( 1.17549e-38, -1.4013e-45) == false (correct)
isLessThan( 1.17549e-38,          -0) == false (correct)
isLessThan( 1.17549e-38,           0) == false (correct)
isLessThan( 1.17549e-38,  1.4013e-45) == false (correct)
isLessThan( 1.17549e-38, 2.93874e-39) == false (correct)
isLessThan( 1.17549e-38, 5.87747e-39) == false (correct)
isLessThan( 1.17549e-38, 1.17549e-38) == false (correct)
isLessThan( 1.17549e-38,       1e-10) ==  true (correct)
isLessThan( 1.17549e-38,           1) ==  true (correct)
isLessThan( 1.17549e-38,       1e+10) ==  true (correct)
```

```
isLessThan( 1.17549e-38, 3.40282e+38) ==  true (correct)
isLessThan( 1.17549e-38,         inf) ==  true (correct)
isLessThan(       1e-10,        -inf) == false (correct)
isLessThan(       1e-10,-3.40282e+38) == false (correct)
isLessThan(       1e-10,      -1e+10) == false (correct)
isLessThan(       1e-10,      -1e-10) == false (correct)
isLessThan(       1e-10,-1.17549e-38) == false (correct)
isLessThan(       1e-10,-5.87747e-39) == false (correct)
isLessThan(       1e-10,-2.93874e-39) == false (correct)
isLessThan(       1e-10, -1.4013e-45) == false (correct)
isLessThan(       1e-10,          -0) == false (correct)
isLessThan(       1e-10,           0) == false (correct)
isLessThan(       1e-10,  1.4013e-45) == false (correct)
isLessThan(       1e-10, 2.93874e-39) == false (correct)
isLessThan(       1e-10, 5.87747e-39) == false (correct)
isLessThan(       1e-10, 1.17549e-38) == false (correct)
isLessThan(       1e-10,       1e-10) == false (correct)
isLessThan(       1e-10,           1) ==  true (correct)
isLessThan(       1e-10,       1e+10) ==  true (correct)
isLessThan(       1e-10, 3.40282e+38) ==  true (correct)
isLessThan(       1e-10,         inf) ==  true (correct)
isLessThan(           1,        -inf) == false (correct)
isLessThan(           1,-3.40282e+38) == false (correct)
isLessThan(           1,      -1e+10) == false (correct)
isLessThan(           1,      -1e-10) == false (correct)
isLessThan(           1,-1.17549e-38) == false (correct)
isLessThan(           1,-5.87747e-39) == false (correct)
isLessThan(           1,-2.93874e-39) == false (correct)
isLessThan(           1, -1.4013e-45) == false (correct)
isLessThan(           1,          -0) == false (correct)
isLessThan(           1,           0) == false (correct)
isLessThan(           1,  1.4013e-45) == false (correct)
isLessThan(           1, 2.93874e-39) == false (correct)
isLessThan(           1, 5.87747e-39) == false (correct)
isLessThan(           1, 1.17549e-38) == false (correct)
isLessThan(           1,       1e-10) == false (correct)
isLessThan(           1,           1) == false (correct)
isLessThan(           1,       1e+10) ==  true (correct)
```

```
isLessThan(           1, 3.40282e+38) ==  true (correct)
isLessThan(           1,         inf) ==  true (correct)
isLessThan(       1e+10,        -inf) == false (correct)
isLessThan(       1e+10,-3.40282e+38) == false (correct)
isLessThan(       1e+10,      -1e+10) == false (correct)
isLessThan(       1e+10,      -1e-10) == false (correct)
isLessThan(       1e+10,-1.17549e-38) == false (correct)
isLessThan(       1e+10,-5.87747e-39) == false (correct)
isLessThan(       1e+10,-2.93874e-39) == false (correct)
isLessThan(       1e+10, -1.4013e-45) == false (correct)
isLessThan(       1e+10,          -0) == false (correct)
isLessThan(       1e+10,           0) == false (correct)
isLessThan(       1e+10,  1.4013e-45) == false (correct)
isLessThan(       1e+10, 2.93874e-39) == false (correct)
isLessThan(       1e+10, 5.87747e-39) == false (correct)
isLessThan(       1e+10, 1.17549e-38) == false (correct)
isLessThan(       1e+10,       1e-10) == false (correct)
isLessThan(       1e+10,           1) == false (correct)
isLessThan(       1e+10,       1e+10) == false (correct)
isLessThan(       1e+10, 3.40282e+38) ==  true (correct)
isLessThan(       1e+10,         inf) ==  true (correct)
isLessThan( 3.40282e+38,        -inf) == false (correct)
isLessThan( 3.40282e+38,-3.40282e+38) == false (correct)
isLessThan( 3.40282e+38,      -1e+10) == false (correct)
isLessThan( 3.40282e+38,      -1e-10) == false (correct)
isLessThan( 3.40282e+38,-1.17549e-38) == false (correct)
isLessThan( 3.40282e+38,-5.87747e-39) == false (correct)
isLessThan( 3.40282e+38,-2.93874e-39) == false (correct)
isLessThan( 3.40282e+38, -1.4013e-45) == false (correct)
isLessThan( 3.40282e+38,          -0) == false (correct)
isLessThan( 3.40282e+38,           0) == false (correct)
isLessThan( 3.40282e+38,  1.4013e-45) == false (correct)
isLessThan( 3.40282e+38, 2.93874e-39) == false (correct)
isLessThan( 3.40282e+38, 5.87747e-39) == false (correct)
isLessThan( 3.40282e+38, 1.17549e-38) == false (correct)
isLessThan( 3.40282e+38,       1e-10) == false (correct)
isLessThan( 3.40282e+38,           1) == false (correct)
isLessThan( 3.40282e+38,       1e+10) == false (correct)
```

```
isLessThan( 3.40282e+38, 3.40282e+38) == false (correct)
isLessThan( 3.40282e+38,         inf) ==  true (correct)
isLessThan(         inf,        -inf) == false (correct)
isLessThan(         inf,-3.40282e+38) == false (correct)
isLessThan(         inf,      -1e+10) == false (correct)
isLessThan(         inf,      -1e-10) == false (correct)
isLessThan(         inf,-1.17549e-38) == false (correct)
isLessThan(         inf,-5.87747e-39) == false (correct)
isLessThan(         inf,-2.93874e-39) == false (correct)
isLessThan(         inf, -1.4013e-45) == false (correct)
isLessThan(         inf,          -0) == false (correct)
isLessThan(         inf,           0) == false (correct)
isLessThan(         inf,  1.4013e-45) == false (correct)
isLessThan(         inf, 2.93874e-39) == false (correct)
isLessThan(         inf, 5.87747e-39) == false (correct)
isLessThan(         inf, 1.17549e-38) == false (correct)
isLessThan(         inf,       1e-10) == false (correct)
isLessThan(         inf,           1) == false (correct)
isLessThan(         inf,       1e+10) == false (correct)
isLessThan(         inf, 3.40282e+38) == false (correct)
isLessThan(         inf,         inf) == false (correct)
```