README

Created on: February 24, 2012
By: Komail Dharsee

This program is designed to take two arguments. The first argument is a 32 bit binary number and the second argument is the type to interpret the first argument.

Data Structures Used:

- Array: to hold each digit obtained by mod and divide

Format Analysis & Design:
After checks the input for validity and format, the program begins by going one out of two paths depending on int or float input.
If int, the number gets taken and put into an unsigned long it to hold the actual value. This conversion from string to int runs in constant time O(1) such that it always runs on a 32 bit sized input. Next, it checks for negativity, if so the number gets turned to positive through a two's complement standard and printed out with an appended "-". Otherwise it goes straight to printing. Print run-time is in worst case constant time O(1) because the actual value of the number cannot go higher than 2^8 thus limiting the number of operations to a constant worst case time.

If float, The first check is to see if the number is in normalized, denormalized or any form of special cases (infinity, negative infinity). In any case, the run time is the same due to the nature of the input (fixed at 32 bit). The program goes through and gets the value of the mantissa and exponent, then prints out in worst case constant time O(1). Printing in special case and denormalized values goes through a different algorithm of printing such that it outputs each bit into an array and flips it around to print.
In normalized form, the print just goes through the value of the mantissa and prints out each bit accordingly.