ARCHORG February 20, 2017
Case Project #1 Prof RLUy/MPeradilla

Case Project: ALU simulator (use any programming language) or x86-32 assembly langague

Deadline: Submission of source code via CANVAS on March 30, 2017 at 11:59pm
Demo: March 30, 2017 class time

Topics:

1. IEEE-754 Decimal-32 floating point converter (including all special cases)
    a. Input:
        - Decimal and base-10 (i.e., $127.0x10^5$)
        - Decimal only (i.e., 5.5)
    b. Output:
        i. Binary output with space (i.e., S CCCCC EEEEEEEE MMMMMMMMMM MMMMMMMMMM)
        ii. Hexadecimal (i.e., 43280000)
2. IEEE-754 Decimal-64 floating point converter (including all special cases)
    a. Input:
        - Decimal and base-10 (i.e., $127.0x10^5$)
        - Decimal only (i.e., 5.5)
    b. Output:
        i. Binary output with space (i.e., S CCCCC EEEEEEEE MMMMMMMMMM MMMMMMMMMM MMMMMMMMMM MMMMMMMMMM)
        ii. Hexadecimal (i.e., 4328000000000000)
3. IEEE-754 Binary-32 floating point converter (including all special cases)
    - Input:
        - binary mantissa and base-2 (i.e., $101.01x2^5$)
        - decimal mantissa and base-10 (i.e., $4.5x10^2$)
        - decimal mantissa only (i.e., 4.5)
    - Output:
        - Binary output with space (i.e., 0 10000110 01010000000000000000000)
        - Hexadecimal (43280000)
4. IEEE-754 Binary-64 floating point converter (including all special cases)
    - Input:
        - binary mantissa and base-2 (i.e., $101.01x2^5$)
        - decimal mantissa and base-10 (i.e., $4.5x10^2$)
        - decimal mantissa only (i.e., 4.5)
    - Output:

- Binary output with space (see example above)
- Hexadecimal (see example above)

5. Sequential Circuit Binary Multiplier simulator (accepts both decimal & binary input, up to 16-bit)
6. Non-Restoring Unsigned Division simulator (accepts both decimal & binary input, up to 16-bit)
7. Restoring Unsigned Division simulator (accepts both decimal & binary input, up to 16-bit)
8. Unicode generator. Input: Unicode; Output: UTF-8, UTF-16, UTF-32.
9. TIC-TAC-TOE program in [using x86-32 assembly]
10. MxM matrix multiplication in [using x86-32 assembly]
11. Optimized sorting algorithm (bubble sort): a.) input: # of integer and the integers b.) output: option to show the sort (1) step-by-step or (2) final sorted output [using x86-32 assembly language]
12. Optimized sorting algorithm (insertion sort): a.) input: # of integer and the integers b.) output: option to show the sort (1) step-by-step or (2) final sorted output [using x86-32 assembly]
13. Optimized sorting algorithm (selection sort): a.) input: # of integer and the integers b.) output: option to show the sort (1) step-by-step or (2) final sorted output [using x86-32 assembly]