CSARCH2
Simulation Project

Case Project: CSARCH2 simulator

General Directions:

- Application platform: Stand-alone or web-based. Regardless, there should be a GUI (graphics user interface) instead of "text "-based output.
- Programming languages: any programming languages (Web-based, C, JAR [stand-alone self-executable], assembly language, Python, etc.).
- Application repository (source code and analysis writeup): GitHub (make sure that I can access it).

Output (all stored in GitHub):

a.) Screenshot of the program output (all possible test cases that will cover the specifications (normal, special case, different inputs, etc.)
b.) Short video (5 minutes long, speedup allowed if needed) stored in GitHub
  - Showing program compilation is correct.
  - Showing all test cases that will cover the specifications (normal, special case, different inputs, etc.)
c.) **S**ource code /executable program (stand-alone) or link to the web-based app

* Project demo if needed. Either face-to-face or through Zoom.

Group #	Topic (see description on next page)
01	IEEE-754 Binary-32 floating-point operation
02	Decimal-32 floating point converter
03	Decimal-64 floating point converter
04	Decimal-128 floating point converter
05	Binary-16 floating point converter
06	Binary-32 floating point converter
07	Binary-64 floating point converter
08	Binary-128 floating point converter
09	Unicode converter/translator

* IEEE-754 binary-32 floating point operation
    - Input: (1) Two operands in binary and base-2 (2) Choice of rounding (G/R/S or rounding) (3) Number of digits supported
    - Process: addition of two operands
    - Output: a.) Step-by-step operation (i.e., 1. Initial normalization 2.) operation 3.) post-operation normalization 4.) final answer; (b.) Option to output in text file

* IEEE-754 Decimal-32 floating-point converter (including all special cases)
    - Input: Decimal and base-10 (i.e., $127.0x10^5$) – should be able to handle more than 7 digits properly (provide an option for the user to choose rounding method). Also, should support special cases (i.e., NaN).
    - Output: (1) binary output with space between sections (2) its hexadecimal equivalent (3) with the option to output in the text file.

* IEEE-754 Decimal-64 floating-point converter (including all special cases)
    - Input: Decimal and base-10 (i.e., $127.0x10^5$) – should handle more than 16 digits properly (provide options for the user to choose rounding method). Also, should support special cases (i.e., NaN).
    - Output: (1) binary output with space between section (2) its hexadecimal equivalent (3) with option to output in text file.

* IEEE-754 Decimal-128 floating-point converter (including all special cases)
    - Input: Decimal and base-10 (i.e., $127.0x10^5$) – should handle more than 34 digits properly (provide options for the user to choose rounding method). Also, should support special cases (i.e., NaN).
    - Output: (1) binary output with space between section (2) its hexadecimal equivalent (3) with option to output in text file.

* IEEE-754 Binary-16 floating point converter (including all special cases)
    - Input: (1) binary mantissa and base-2 (i.e., $101.01x2^5$) (2) Decimal and base-10 (i.e. $65.0x10^3$) Also should support special cases (i.e., NaN).
    - Output: (1) binary output with space between section (2) its hexadecimal equivalent (3) with option to output in text file.

* IEEE-754 Binary-32 floating point converter (including all special cases)
    - Input: (1) binary mantissa and base-2 (i.e., $101.01x2^5$) (2) Decimal and base-10 (i.e. $65.0x10^3$). Also should support special cases (i.e., NaN).
    - Output: (1) binary output with space between section (2) its hexadecimal equivalent (3) with option to output in text file.

* IEEE-754 Binary-64 floating point converter (including all special cases)
    - Input: (1) binary mantissa and base-2 (i.e., $101.01x2^5$) (2) Decimal and base-10 (i.e. $65.0x10^3$). Also should support special cases (i.e., NaN).
    - Output: (1) binary output with space between section (2) its hexadecimal equivalent (3) with option to output in text file.

* IEEE-754 Binary-128 floating point converter (including all special cases)
    - Input: (1) binary mantissa and base-2 (i.e., $101.01x2^5$) (2) Decimal and base-10 (i.e. $65.0x10^3$). Also should support special cases (i.e., NaN).
    - Output: (1) binary output with space between section (2) its hexadecimal equivalent (3) with option to output in text file.

*Unicode (converter/translator)

- Converter:
    - Input: Unicode (with invalid Unicode check)
    - Output: UTF-8; UTF-16; UTF-32 [format: xx xx xx ; where x is hex nibble]
- Translator:
    - Input: either UTF-8; UTF-16 or UTF-32
    - Output: Unicode code point
- In both cases, with the option to output in text file.