

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., 101.01×2^5) (2) Decimal and base-10 (i.e. 65.0×10^3) 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.

For the given specs we used Java as our programming language, under the public class `HalfPrecisionConverter`, we have our classes `convertInput`, `base10toDecimal`, `decimaltoBinary`, `binaryToHex`, `normalize`, and `getExpRep`.

For `convertInput`, we prepare our variables, the `XIndex`, `decimal` and `signBit` in order to concatenate after reading the input, the class inside gets the signbit of the input, checks if the input is base-10 that converts into a binary value, `normalize` to 1.f, gets the exponent and fraction.

For `base10Decimal` class, its function is to convert base-10 input to decimal value as double, with the formula `double decimal = base * Math.pow(10, exponent)`.

For the `decimalToBinary`, its function is to convert decimal value to binary value as string. If the fractional part is equal to 0 it returns to the binary integer part, if the fractional part is more than 0, the length should not be more than 32 bits, else, it appends to 1 or 0 then it returns to a concatenated binary string.

For the class `binaryToHex` it converts the binary value from the class `decimalToBinary` to hexadecimal. It checks and cuts the spaces of the given binary, then an operation modulo 4 is used until it reaches 0. Then it initializes the stringbuilder to store the hexadecimal.

For the `normalize` class, it normalizes binary to 1.f

The `normalize` method adjusts a binary string to represent a normalized floating-point number with the format `1.f * 2^e`. It extracts the exponent part from the input string and adjusts it based on the position of the radix point. Then, it constructs the normalized representation by appending the binary fraction part to '1.' and the adjusted exponent, returning it as a string.

For the `getExpRep`, it gets the binary exponent representation as string.

Code Snippets:

Half Precision Converter

Input:

IEEE-754 Binary-16 floating point:

Binary: 0 11111 00000000000

Hexadecimal: 7C00

Half Precision Converter

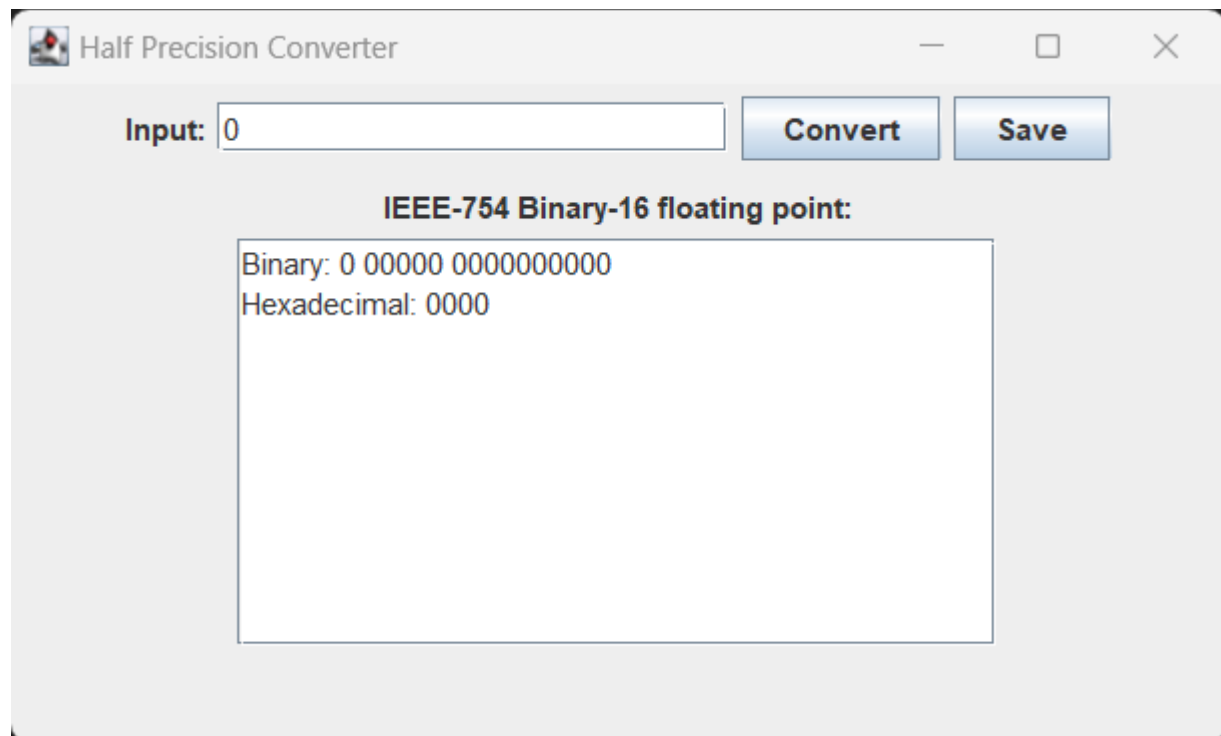
Input:

IEEE-754 Binary-16 floating point:

Binary: 0 100010 1110100001

Hexadecimal: 08BA1

Zero



Half Precision Converter

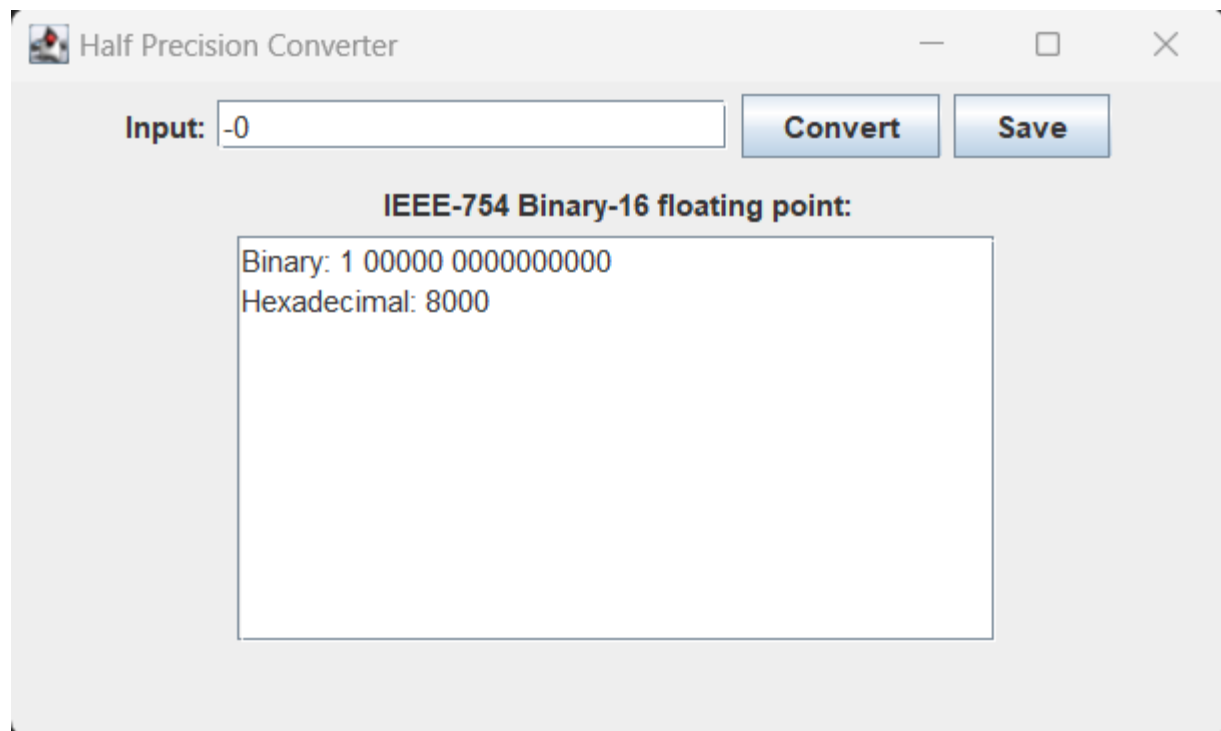
Input:

Convert **Save**

IEEE-754 Binary-16 floating point:

Binary: 0 00000 000000000000
Hexadecimal: 0000

This screenshot shows the 'Half Precision Converter' application window. The title bar includes a small icon and the text 'Half Precision Converter', along with standard window controls (minimize, maximize, close). The main interface features an 'Input' label followed by a text box containing the value '0'. To the right of the text box are two buttons labeled 'Convert' and 'Save'. Below these elements, the text 'IEEE-754 Binary-16 floating point:' is displayed. Underneath this text is a large rectangular box containing the converted values: 'Binary: 0 00000 000000000000' and 'Hexadecimal: 0000'.



Half Precision Converter

Input:

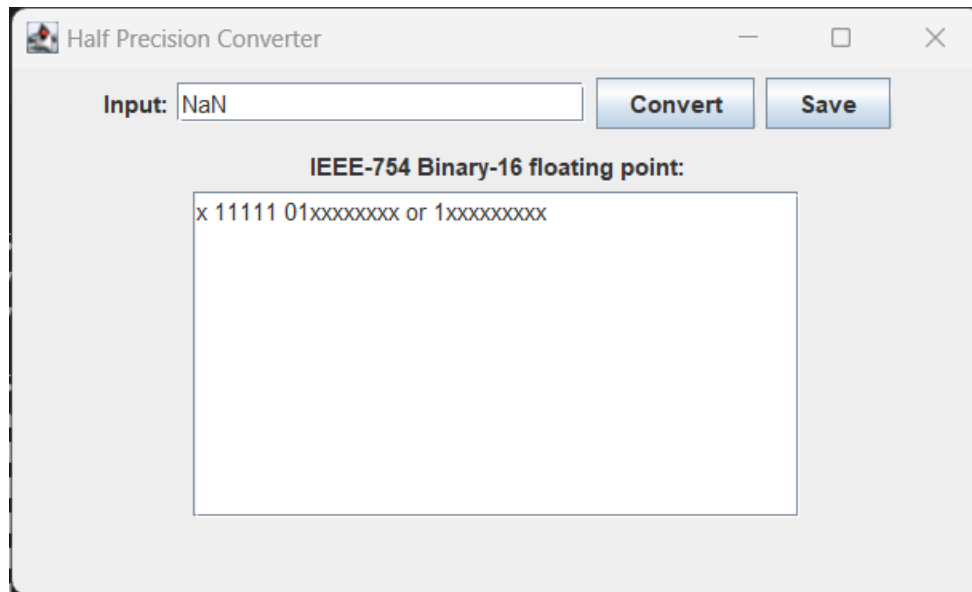
Convert **Save**

IEEE-754 Binary-16 floating point:

Binary: 1 00000 000000000000
Hexadecimal: 8000

This screenshot shows the 'Half Precision Converter' application window with the input value changed to '-0'. The title bar and window controls remain the same. The 'Input' text box now contains '-0'. The 'Convert' and 'Save' buttons are still present. The output section, under the heading 'IEEE-754 Binary-16 floating point:', shows the converted values: 'Binary: 1 00000 000000000000' and 'Hexadecimal: 8000'.

NaN



Half Precision Converter

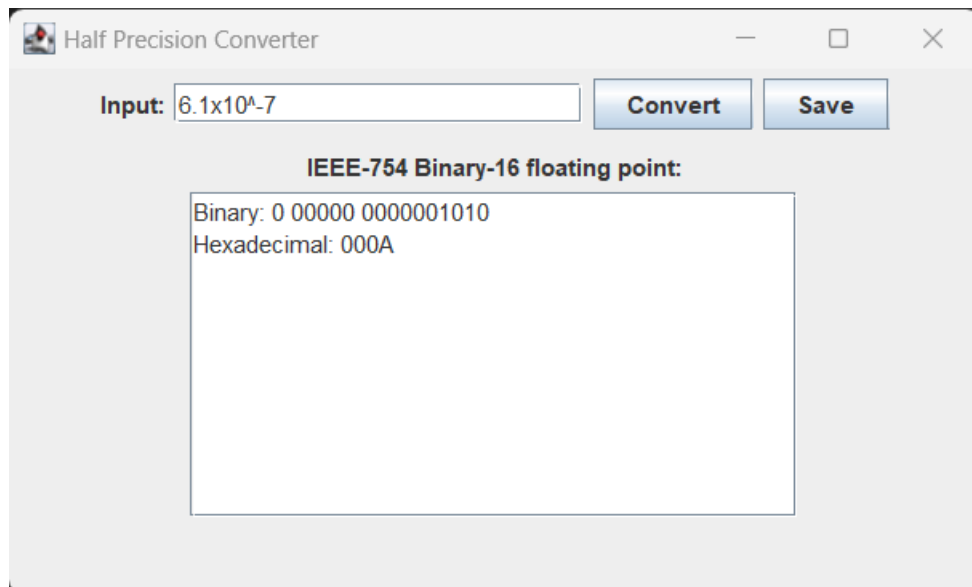
Input: NaN Convert Save

IEEE-754 Binary-16 floating point:

x 1111 01xxxxxxx or 1xxxxxxx

The image shows a software window titled "Half Precision Converter". It has a standard Windows-style title bar with a minimize button, a maximize button, and a close button. Below the title bar, there is an "Input:" label followed by a text box containing "NaN". To the right of the text box are two buttons: "Convert" and "Save". Below these elements, there is a section titled "IEEE-754 Binary-16 floating point:". Under this title is a large rectangular text area containing the text "x 1111 01xxxxxxx or 1xxxxxxx".

Denormalized




Half Precision Converter

Input: 6.1×10^{-7} Convert Save

IEEE-754 Binary-16 floating point:

Binary: 0 00000 0000001010
Hexadecimal: 000A

The image shows a software window titled "Half Precision Converter". It has a standard Windows-style title bar with a minimize button, a maximize button, and a close button. Below the title bar, there is an "Input:" label followed by a text box containing the scientific notation 6.1×10^{-7} . To the right of the text box are two buttons: "Convert" and "Save". Below these elements, there is a section titled "IEEE-754 Binary-16 floating point:". Under this title is a large rectangular text area containing two lines of text: "Binary: 0 00000 0000001010" and "Hexadecimal: 000A".

 Half Precision Converter

Input:

Convert

Save

IEEE-754 Binary-16 floating point:

Binary: 0 00000 0100000000

Hexadecimal: 0100