A1

In [4]:
```
# Standard imports
import numpy as np
np.seterr(all='ignore');  # allows floating-point exceptions
import matplotlib.pyplot as plt
```

Q1: randfp

In [1]:
```
def randfp(t, L, U):
    '''
     b = randfp(t, L, U)

     Generate a random normalized binary floating-point number with
     t digits, and an exponent in the range [L,U]. For example,

       b = randfp(5, -4, 4)

     might yield

       b = '-0.10111b-2'

     or

       b = '+0.11100b4'

     Note that the output is a string, and that the first character is
     always either a '+' or '-'. The number after the 'b' is
     the exponent for the base 2, although the exponent itself is
     represented in base-10. For example,

       b = '+0.11100b4'

     represents the number 0.11100 x 2^4.
    '''

    # ==== YOUR CODE HERE ====

    # Output should start as empty, each section will be added sequentially
    output = ''

    # Generates either 1 or 0
    sign = np.random.randint(0,2)

    if (sign == 0):
        output += '+'
    else:
        output += '-'

    output += '0.1'
```

```
    # Each digit is random except first cause its randomized
    for i in range (0, t-1):
        output += str(np.random.randint(0,2))

    output += 'b'

    output += str(np.random.randint(L, U+1))

    return output
```

In [7]:
```
b = randfp(5, -4, 4)
print(b)
```

```
-0.10000b0
```

In [6]:
```
? randfp
```

```
Signature:  randfp(t, L, U)
Docstring:
b = randfp(t, L, U)

Generate a random normalized binary floating-point number with
t digits, and an exponent in the range [L,U]. For example,

  b = randfp(5, -4, 4)

might yield

  b = '-0.10111b-2'

or

  b = '+0.11100b4'

Note that the output is a string, and that the first character is
always either a '+' or '-'. The number after the 'b' is
the exponent for the base 2, although the exponent itself is
represented in base-10. For example,

  b = '+0.11100b4'

represents the number 0.11100 x 2^4.
File:      c:\users\robbie\appdata\local\temp\ipykernel_16044\74839634.py
Type:      function
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: