Coding Task: 1
Theme: Divide and Conquer

Rules:

- Use **Python 3.7** or higher in **Google Colab**. It comes with pre-installed packages like numpy, time, etc.
- Submission should be a single **.zip** file (no other extensions like .rar, .gz) Keep this in mind.
- Submission should contain
  1. Code files in **.ipynb** format. Use separate **.ipynb** files for separate questions. **DO NOT** use .py files. We will test your code in Google Colab.

  2. README file describing how to run the code. TAs will run your code to see if they are really working.

  3. A write-up in **.pdf** format. It should contain answers to the questions with appropriate screenshots or test results from the output of your code. Treat this as a mini-report that TAs will evaluate after your code successfully runs.
- **DO NOT** use built-in functions for cases where you have been asked to implement that exact same functionality. You can use built-in functions if you want to test whether your designed function produces the correct output.

Q1. **Insertion Sort**:

a. Insertion Sort is one example of a very simple quadratic time sorting algorithm. Read: https://en.wikipedia.org/wiki/Insertion_sort .

b. Implement insertion sort. You should specifically use **recursion** to implement it.

c. Plot the running time of your implementation vs varying input sizes. Does it match with the theoretical running time complexity? You may use matplotlib to plot your graph. If your graph doesn't obey the theoretical formula, why do you think it behaves so?

Q2. **Inverse Fast Fourier Transform (IFFT)**:

a. Take two polynomials in coefficient form.

$A(x) = 1 + 2x + 3x^3$ and $B(x) = 5 + 7x^4$

b. Find out what is C(x) using the naive polynomial multiplication algorithm as discussed in the tutorial, where C(x) = A(x).B(x)

c. Now implement the Inverse Fast Fourier Transform (IFFT) Algorithm. You may use the FFT algorithm shown in tutorial if you wish. You may also use `numpy.fft.ifft` package to verify that your implementation is correct.

d. Use the FFT and IFFT to perform the Evaluation and Interpolation procedures to multiply polynomials A(x) and B(x). Check whether it matches your result from part (b). Keep in mind that a polynomial of degree ‘d’ needs atleast ‘d + 1’ points for its unique value representation. Pad zeros accordingly keeping in mind the degree of C(x).