Maymunah Hicks

Dr. Shannon Starr

Scientific Programming

4 November 2024

Project 3

In Reverse_FYK.ipynb add a portion of the program to run the forward FYK from the data you obtained to show the actual permutation and/or the sorted data.

```
import numpy as np

import random

import math

import bisect

# import matplotlib.pyplot as plt


def find_min(numlist):

    # This function finds the minimum of a list of numbers

    # and returns the position of the (first occurrence of the) minimum

    # Input :

    # numlist : list of numbers to sort

    # Output :

    # position : an index for where the minimum is in the list


    temp_pos = 0
    temp_val = numlist[temp_pos]

    for i in range(1, len(numlist)):

        if numlist[i] < temp_val:  # Check if numlist[i] < temp_val

            temp_pos = i  # Update temp_pos to i

            temp_val = numlist[temp_pos]  # Update temp_val


    return temp_pos


def FYKdata(numlist):

    # This function finds the reverse FYK permutation for a list of numbers

    # FYK stands for Fisher Yates Knuth

    # Input :

    # numlist : list of numbers to sort

    # Output :

    # FYKlist : list of integers giving the FYK data of the permutation to sort
it


    temp_numlist = numlist[:]  # Make a copy of numlist for modification

    FYKlist = []  # Initialize output data as empty

    for i in range(len(numlist)):  # Run find_min n times (n = len(numlist))

        temp_pos = find_min(temp_numlist)  # Get position from find_min

        FYKlist.append(temp_pos)  # Append that number to FYKlist

        temp_numlist.pop(temp_pos)  # Remove that term in the temp_numlist


    return FYKlist
```

```python
def forward_FYK(numlist, FYKlist):
    # This function reconstructs the sorted list from the reverse FYK data
    sorted_list = []  # Initialize sorted list
    temp_numlist = numlist[:]  # Copy of the original list
    for pos in FYKlist:
        sorted_list.append(temp_numlist.pop(pos))  # Append and remove elements in FYK order
    return sorted_list


# Generate a random list and apply reverse FYK
perm_length = 100  # Change this to change the length of your original permutation
perm_pi = np.random.permutation(perm_length)  # Create a random permutation
perm_pi_list = perm_pi.tolist()  # Change the array to a list
print("Original List:", perm_pi_list)  # Print the random list
FYK_pi_data = FYKdata(perm_pi_list)  # Apply the reverse FYK algorithm to our perm
print("Reverse FYK Data:", FYK_pi_data)  # Print the reverse FYK data


# Use the forward FYK function to reconstruct the sorted list
reconstructed_list = forward_FYK(perm_pi_list, FYK_pi_data)  # Reconstruct the sorted list
print("Reconstructed (Sorted) List:", reconstructed_list)  # Print the sorted list
```

Original List: [97, 96, 50, 69, 11, 57, 2, 18, 88, 47, 37, 15, 80, 93, 54, 34, 43, 76, 70, 77, 74, 7, 94, 42, 87, 58, 27, 83, 91, 44, 89, 13, 0, 75, 49, 84, 30, 78, 79, 12, 1, 92, 68, 22, 72, 38, 63, 28, 41, 8, 19, 29, 16, 4, 36, 59, 71, 53, 26, 52, 45, 95, 17, 51, 32, 81, 21, 35, 9, 85, 73, 31, 66, 48, 25, 55, 98, 61, 40, 6, 24, 56, 23, 60, 90, 3, 5, 64, 82, 33, 46, 99, 86, 67, 10, 65, 20, 14, 39, 62]

Reverse FYK Data: [32, 39, 6, 82, 50, 81, 75, 20, 45, 62, 84, 4, 35, 28, 83, 9, 43, 51, 5, 40, 77, 52, 34, 64, 62, 57, 44, 21, 36, 37, 28, 49, 44, 59, 11, 44, 35, 7, 31, 59, 48, 32, 16, 10, 19, 33, 48, 6, 38, 20, 2, 31, 29, 28, 7, 32, 34, 3, 12, 23, 31, 30, 37, 22, 30, 34, 27, 32, 20, 2, 6, 19, 18, 21, 7, 12, 5, 5, 11, 11, 3, 12, 15, 6, 8, 10, 13, 5, 2, 5, 8, 4, 4, 2, 2, 2, 1, 0, 0, 0]

Reconstructed (Sorted) List After Implementing Forward FYK function: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99]