Design and Analysis of Algorithms

Instructor : Dr. Sasanka Roy

Assignment submitted by : Ranjan Kumar Choubey

Roll No : CS2306

4.2 Sorting by Permutations

The task is to create a program that will sort a given array of n distinct integers by generating and evaluating all possible permutations of the given n-sized input array. The goal is to find and output the permutation that represents the earliest non-decreasing arrangement, thereby sorting the array in ascending order. You must implement a brute-force approach to solve this problem, such that the program looks through all permutations and verify if the sequence is sorted.

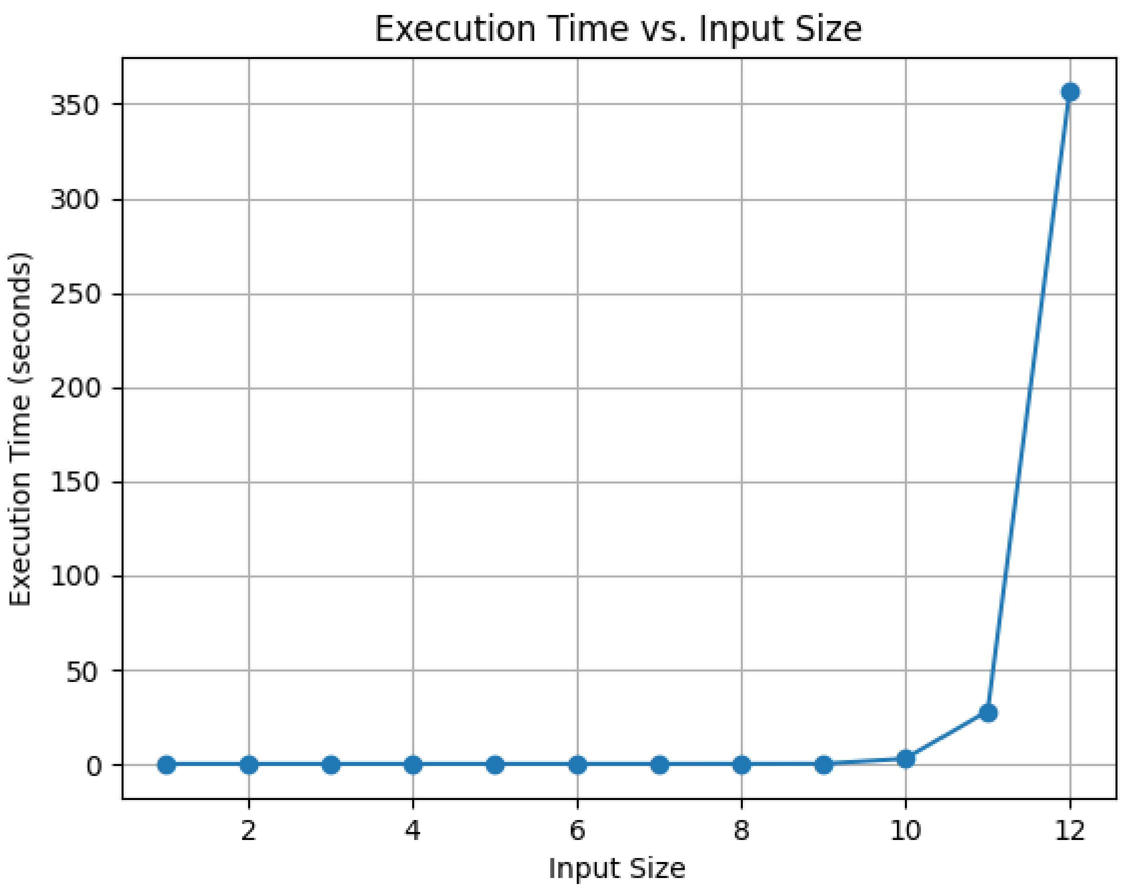
In [ ]:
```
import itertools
import random
import sys
import timeit
import matplotlib.pyplot as plt

def BruteForce_Sorting(input_size, timeout):

    input_data = [random.randrange(1, 100) for i in range(input_size)]
    print("Input: ", input_data)

    start_time = timeit.default_timer()

    for p in itertools.permutations(input_data):
        if all(p[i] <= p[i + 1] for i in range(len(p) - 1)) and (timeit.d
            print("Output: ", list(p))
            return

    # If the timeout is reached and no sorted permutation is found
    print("Time over - Execution terminated.")
    sys.exit(1)

if __name__ == "__main__":
    input_size = 10
    timeout = 1 * 60  # 1 minutes = 60 seconds
    BruteForce_Sorting(input_size, timeout)
```

```
Input:  [86, 96, 58, 4, 90, 27, 18, 63, 5, 69]
Output:  [4, 5, 18, 27, 58, 63, 69, 86, 90, 96]
```

In [ ]:
```
import itertools
import random
import sys
import timeit
import matplotlib.pyplot as plt
```

```python
def BruteForce_Sorting(input_size):
    input_data = [random.randrange(1, 100) for i in range(input_size)]
    start_time = timeit.default_timer()

    for p in itertools.permutations(input_data):
        if all(p[i] <= p[i + 1] for i in range(len(p) - 1)):
            return timeit.default_timer() - start_time

    return float('inf')

if __name__ == "__main__":
    input_sizes = range(5, 21)  # Input sizes from 5 to 20
    execution_times = []

    for input_size in input_sizes:
        print(f"Running with input size {input_size}...")
        execution_time = BruteForce_Sorting(input_size)
        execution_times.append(execution_time)

    # Plotting the results
    plt.plot(input_sizes, execution_times, marker='o')
    plt.xlabel('Input Size')
    plt.ylabel('Execution Time (seconds)')
    plt.title('Execution Time vs. Input Size')
    plt.grid(True)
    plt.show()
```

```
Running with input size 1...
Running with input size 2...
Running with input size 3...
Running with input size 4...
Running with input size 5...
Running with input size 6...
Running with input size 7...
Running with input size 8...
Running with input size 9...
Running with input size 10...
Running with input size 11...
Running with input size 12...
```


Execution Time vs. Input Size
Execution Time (seconds)
0
50
100
150
200
250
300
350
2
4
6
8
10
12
Input Size