CCCS 104 - Data Structures and Algorithms

LEARNING TASK (SORTING)

GROUP NO: _____9_____ SECTION:__BSCS2A__

GROUP LEADER: Venn P. Delos Santos

GROUP MEMBERS: Marc Christian D. Tumaneng

John Mark A. Pajenago

SORTING ALGORITHM

Computer science requires the use of sorting algorithms. This method allows data representation to be more comprehensible. In simpler terms, we can say that it basically refers to organizing data in a logical manner, either by sorting it in ascending or descending order. Some examples of sorting algorithms are **Selection Sort**, where it works by taking the smallest element in an unsorted array and bringing it to the front from left to right until it finds the smallest element. **Exchange Sort**, this sort compares the first element with each following element of the array, making any necessary swaps. **Shell Sort**, it is a sort that is considered as a generalized version of insertion sort, where it first sorts elements that are far apart from each other and successively reduces the interval between the elements to be sorted. **Radix Sort,** this sort arranges the elements by first grouping the individual digits of the same place value. Then, sort the elements according to their increasing/decreasing order.

In the particular problem that we're going to solve we will use the **Insertion sort**, it is a sorting algorithm that places or sorts each element in the suitable position after each iteration. In this sort Finding the correct location for the current number is important. You must move the current element to make space for the new number after the correct position has been identified. The time complexities of an insertion sort are; Worst Case Complexity: O(n2), Best Case Complexity O(n), Average Case Complexity O(n2), and it has a space complexity of O(1).

REAL-LIFE SORTING EXAMPLE/PROBLEM

The Final Examination for your Physical Education Class has already concluded. The following day, you are assigned by your teacher to arrange the test papers of your class in an ascending order based on the students' scores so they can be distributed to the entire class accordingly. There are 20 students in your class, including you. The scores of the students' exams are 88, 83, 92, 84, 97, 93, 95, 87, 81, 92, 88, 85, 87, 94, 82, 86, 96, 98, 85, 94.

PROPOSED SOLUTION

To help sort the test paper in an ascending order based on the student's scores, we designed a program that implements a sorting algorithm (insertion sort) to arrange the right order of the test papers. Using this solution we can arrange the test papers easily and we can avoid further confusions.

Program Code

```
# BSCS2A - Group 9
# Python program that implements Insertion Sort algorithm

# Define the function for insertion sort with the parameters needed
import time #import time for getting the execution time

def insertion_sort(array):

    for step in range(1, len(array)):
        key = array[step]
        j = step - 1

        """ Compare key with each element on the left of it
            until an element smaller than it is found"""
        # For descending order, change key<array[j] to key>array[j].
        while j >= 0 and key < array[j]:
            array[j + 1] = array[j]
            j = j - 1

        # Place key at after the element just smaller than it.
        array[j + 1] = key
```

```python
# Get the Exam Scores
exam_scores = list(map(int, input("Enter the Exam Scores: \n").split()))

#get the start time
st = time.time()

insertion_sort(exam_scores)

#Display the sorted exam scores in an ascending order
print('\nSorted Exam Scores in Ascending Order:')
print(exam_scores)

# get the end time
et = time.time()
# get the execution time
elapsed_time = et - st
print('Execution time:', elapsed_time, 'seconds')
```

RESULTS

The result of the program we made that implements Insertion sort that has a time complexity of Ω(n);

```
PS C:\Users\marcc> & C:/Users/marcc/AppData/Local/Programs/Python/Python310/python.exe "d:/CSPC-2ND YEAR.1ST SEM/CCCS 10
4 -DATA STRUCTURES/LEARNING TASK/Searching, Sorting and Hashing Techniques/CODE/Group9_Insertion_sort.py"
Enter the Exam Scores:
88 83 92 84 97 93 95 87 81 93 95 87 81 92 88 85 87 94 82 86 96 98 85 94

Sorted Exam Scores in Ascending Order:
[81, 81, 82, 83, 84, 85, 85, 86, 87, 87, 87, 88, 88, 92, 92, 93, 93, 94, 94, 95, 95, 96, 97, 98]
Execution time: 0.0009999275207519531 seconds
```

The program successfully sorted the grades into an ascending order, and it was shown that the program's execution time is 0.0009999275207519531 seconds.

The result of the program we made that implements Radix sort that has a time complexity of Ω(nk);

```
PS C:\Users\marcc> & C:/Users/marcc/AppData/Local/Programs/Python/Python310/python.exe "d:/CSPC-2ND YEAR.1ST SEM/CCCS 10
4 -DATA STRUCTURES/LEARNING TASK/Searching, Sorting and Hashing Techniques/CODE/Group9_Radix_sort.py"
Enter the Exam Scores:
88 83 92 84 97 93 95 87 81 93 95 87 81 92 88 85 87 94 82 86 96 98 85 84

Sorted Exam Scores in Ascending Order:
[81, 81, 82, 83, 84, 84, 85, 85, 86, 87, 87, 87, 88, 88, 92, 92, 93, 93, 94, 95, 95, 96, 97, 98]
Execution time: 0.0009992122650146484 seconds
PS C:\Users\marcc>
```

The program successfully sorted the grades into an ascending order, and it was shown that the program's execution time is 0.0009992122650146484 seconds.
Based on the result of each program it was clear that the Radix sort executes faster than the insertion sort.

Program Code

```
# Python program that implements Radix Sort algorithm
# Using counting sort to sort the elements in the basis of significant places
import time #import time for getting the execution time

def countingSort(array, place):
    size = len(array)
    output = [0] * size
    count = [0] * 10

    # Calculate count of elements
    for i in range(0, size):
        index = array[i] // place
        count[index % 10] += 1

    # Calculate cumulative count
    for i in range(1, 10):
        count[i] += count[i - 1]

    # Place the elements in sorted order
    i = size - 1
    while i >= 0:
        index = array[i] // place
        output[count[index % 10] - 1] = array[i]
        count[index % 10] -= 1
        i -= 1

    for i in range(0, size):
        array[i] = output[i]


# Main function to implement radix sort
def radixSort(array):
    # Get maximum element
    max_element = max(array)

    # Apply counting sort to sort elements based on place value.
    place = 1
    while max_element // place > 0:
        countingSort(array, place)
        place *= 10


# Get the Exam Scores
exam_scores = list(map(int, input("Enter the Exam Scores: \n").split()))
```

```
# get the start time
st = time.time()

radixSort(exam_scores)
#Display the sorted exam scores in an ascending order
print('\nSorted Exam Scores in Ascending Order:')
print(exam_scores)

# get the end time
et = time.time()
# get the execution time
elapsed_time = et - st
print('Execution time:', elapsed_time, 'seconds')
```

Tutorial Video

YouTube Link/s: https://youtu.be/H1AB-ISJAFg

REFERENCES

<use ACM citation, see https://www.acm.org/publications/authors/reference-formatting >

[1] Sorting Algorithms, Retrieved November 21, 2022, from

https://www.geeksforgeeks.org/sorting-algorithms-in-python/

[2] Insertion Sort , Retrieved November 21, 2022, from

https://www.programiz.com/dsa/insertion-sort

[3] Real life sorting algorithms example, Retrieved November 21, 2022, from

https://www.mygreatlearning.com/blog/insertion-sort-with-a-real-world-example/?fbclid=IwAR0OG39K

HCW1bjHRoRerFJYBhCmLlSi-Np6tyuIcZGG_VmybWePHsBvHXfw

[4] Radix Sort, Retrieved November 21, 2022, from

https://www.programiz.com/dsa/radix-sort

TAKEAWAYS

Name of Member: Venn P. Delos Santos
Contribution to the Group: Coding the program, documentation, recording
Learnings: Learned how sorting algorithms such as Insertion sort and Radix sort work

Name of Member: Marc Christian D. Tumaneng
Contribution to the Group: Coding the program, documentation, recording
Learnings: I understand the concept and implementation of sorting algorithms like insertion sort and radix sort and its implementation on solving real life problems.

Name of Member: John Mark A. Pajenago
Contribution to the Group: Coding the program, documentation, recording
Learnings:Learned how to use sorting algorithm and how to use it at real life problem