Department of Computer Technology and Information Systems

CTIS264 – Computer Algorithms

Spring 2019 - 2020

Lab Guide 4 - Week 5

Instructor : Füsun YÜRÜTEN

Assistant : Leyla SEZER

Objectives:

- **Random array generation with numpy module**
- **Selection sort algorithm**
- **Bubble Sort Algorithms in two different style.**
- **String matching**
- **Sequential Search**

Q1. Write a Python program that generates a random array using numpy module then sorts the list using selection sort algorithm. Size of the array is 1000 and generates number between 0-100.

ALGORITHM *SelectionSort*$(A[0..n-1])$
//Sorts a given array by selection sort
//Input: An array $A[0..n-1]$ of orderable elements
//Output: Array $A[0..n-1]$ sorted in ascending order
for $i \leftarrow 0$ **to** $n-2$ **do**
    $min \leftarrow i$
    for $j \leftarrow i+1$ **to** $n-1$ **do**
        if $A[j] < A[min]$ $min \leftarrow j$
    swap $A[i]$ and $A[min]$

Output:

```
[ 1 1 2 2 …… 99 99]
0.18502426147460938
```

Q2a.

Write a Python program that generates a random array then sorts the list using **bubble sort algorithm written by <u>using nested for loops</u>**. Size of the array is 1000 and generates number between 0-100.

ALGORITHM
BubbleSort(A[0..n − 1])
//Sorts a given array by bubble sort
//Input: An array A[0..n − 1] of orderable elements
//Output: Array A[0..n − 1] sorted in nondecreasing order
for i ← 0 to n − 2 do
    for j ← 0 to n − 2 − i do
        if A[j + 1]< A[j] swap A[j] and A[j + 1]

- Test the same program generates a random array using numpy module. Size of the array is 1000 and generates number between 0-100. Notice the changes in time.

Output:

```
[ 1 1 2 2 …… 99 99]
0.3469216823577881
```

Q2b.
Write a Python program that generates a random array then sorts the list using **bubble sort algorithm written by using while loop**. Size of the array is 1000 and generates number between 0-100.

```
ALGORITHM
BubbleSort(A[0..n − 1])
//Sorts a given array by bubble sort
//Input: An array A[0..n − 1] of orderable elements
//Output: Array A[0..n − 1] sorted in nondecreasing order
exchange= True
i=0
while  i < n − 2 and exchange do
        exchange= False
        for j ← 0 to n − 2 − i do
                if A[j + 1]< A[j]
                        swap A[j] and A[j + 1]
                        exchange=true
         i ← i+1
```

- Test the same program generates a random array using numpy module. Size of the array is 1000 and generates number between 0-100. Notice the changes in time.

Output:

```
[ 1 1 2 2 …… 99 99]
0.35378289222717285
```

Q3. Write a Python program that;

- Writes a function that gets a text and a pattern, search the pattern inside the text. If it is found returns the first index, otherwise; returns -1.

```
ALGORITHM BruteForceStringMatch(T[0..n − 1], P[0..m − 1])
    //Implements brute-force string matching
    //Input: An array T[0..n − 1] of n characters representing a text and
    //        an array P[0..m − 1] of m characters representing a pattern
    //Output: The index of the first character in the text that starts a
    //          matching substring or −1 if the search is unsuccessful
    for i ← 0 to n − m do
        j ← 0
        while j < m and P[j] = T[i + j] do
            j ← j + 1
        if j = m return i
    return −1
```

- Example: Program searches the pattern; 'NOT' inside the text ''NOBODY_NOTICED_HIM", then returns the first index of the pattern.

Output:

```
First index of  NOT in the text is  7
0.0
```

Q4. Write a Python program that;

- Writes a function that takes a list and a search value, then searches the value inside the list according to the given sequential search algorithm.

ALGORITHM *SequentialSearch*$(A[0..n-1], K)$

```
//Searches for a given value in a given array by sequential search
//Input: An array A[0..n − 1] and a search key K
//Output: The index of the first element of A that matches K
//        or −1 if there are no matching elements
i ← 0
while i < n and A[i] ≠ K do
    i ← i + 1
if i < n return i
else return −1
```

- Program defines the given list below, then inputs a search value from the user, displays the index of the search value if it is found, otherwise; program gives "NOT FOUND" message.

Use the given list values;

```
testlist = [1, 2, 32, 8, 17, 19, 42, 13, 0]
```

Output:

```
Enter a search value: 13
Value found in index 7
Passed time in execution: 0.009287118911743164
```