**Algorithm Task  
  
  
10 - Diagonal Difference**

**Team Number: 165**

|  |  |  |
| --- | --- | --- |
|  | **ID** | **Full Name** |
| **1** | **20210159** | **إسلام مجدي محمد الصفي علي الجزار** |
| **2** | **20210052** | **أحمد رفعت كامل أحمد عفيفي** |
| **3** | **20211104** | **يوسف محمد نادي حسن مرسي** |
| **4** | **20210081** | **أحمد علي إبراهيم سليمان حرب** |
| **5** | **20210057** | **أحمد سعدالدين محمد علي المراكبي** |
| **6** | **20210050** | **أحمد رضا شعبان عبدالوهاب** |

**1- Non-recursive:**

**1.1- Pseudocode:**

**ALGORITHM DiagonalDifference(n, arr) {**

**primaryDiagonal <- 0**

**secondaryDiagonal <- 0**

**for i <- 0 to n – 1 do**

**primaryDiagonal <- primaryDiagonal + arr[i][i]**

**secondaryDiagonal <- secondaryDiagonal + arr[i][n - i - 1]**

**return abs(primaryDiagonal - secondaryDiagonal)**

**}**

**1.2- Analysis:**

**Shape

Description automatically generated with medium confidence**

**So, Time Complexity is Θ(n)**

**Shape

Description automatically generated with low confidence**

**2- Another Non-recursive**

**2.1- Pseudocode:**

**ALGORITHM DiagonalDifference(n, arr) {**

**primaryDiagonal <- 0**

**secondaryDiagonal <- 0**

**for i <- 0 to n – 1 do**

**for j <- 0 to n – 1 do**

**if i = j**

**primaryDiagonal <- primaryDiagonal + arr[i][j]**

**if i = n - j - 1**

**secondaryDiagonal <- secondaryDiagonal + arr[i][j]**

**return abs(primaryDiagonal - secondaryDiagonal)**

**}**

**2.2- Analysis:**

**Shape

Description automatically generated with medium confidence**

**So, Time Complexity is Θ(n2)**

**Shape

Description automatically generated with medium confidence**

**3- Recursive:**

**3.1- Pseudocode:**

**ALGORITHM DiagonalDifference(n, arr, i) {**

**If (i = n)**

**return 0**

**return arr[i][i] – arr[n – i – 1] + DiagonalDifference(n, arr, i + 1)**

**}**

**3.2- Analysis:**

**Shape

Description automatically generated with low confidence**

**So, Time Complexity is Θ(n)**

**A picture containing shape

Description automatically generated**

**3- Comparison**

|  |  |  |  |
| --- | --- | --- | --- |
| **ALGORITHM** | **Time Complexity** | | |
| **Best Case** | **Average Case** | **Worst Case** |
| **Non-recursive** | **Ω(n)** | **Θ(n)** | **O(n)** |
| **Another Non-Recursive** | **Ω(n2)** | **Θ(n2)** | **O(n2)** |
| **Recursive** | **Ω(n)** | **Θ(n)** | **O(n)** |

**First Non-recursive Algorithm is better than another one, because it uses just 1 for loop, So Its Time Complexity is lower**

**And both of the first Non-recursive, and Recursive Algorithm have the same Time Complexity, but as Performance Non-recursive is better**