Algorithm regions

Input: [\*mapRegions – PtMap, \*listPatients – PtList]

BEGIN

Size <- 0;

tempReg <- null;

patient <- null;

\*mapKeyArray = mapKeys(\*mapRegions);

ptListSize <- 0;

listSizeErrorCode <- listSize(\*listPatients, &ptListSize);

regionArray[size];

FOR i <- 0 TO (i < size) DO

mapGet(\*mapRegions, mapKeyArray[i], &tempReg);

FOR j <- 0 TO (j < ptListSize) DO

listGet(\*listPatients, j, &patient);

if (strcmp(patient.region, mapKeyArray[i].region) = 0 AND strlen(patient.region) > 0) THEN

if (isIso(patient)) THEN

regionArray[i] <- tempReg;

END IF

END IF

END FOR

END FOR

FOR j <- 0 TO (j < size) DO

FOR i <- j + 1 TO (i < size) DO

if (strcmp(regionArray[j].name, regionArray[i].name) > 0) THEN

tempReg <- regionArray[j];

regionArray[j] <- regionArray[i];

regionArray[i] <- tempReg;

END IF

END FOR

END FOR

FOR i <- 0 TO (i < size-1) DO

PRINT “$regionArray[i]”

END FOR

free(mapKeyArray)

END

Regions ->

Este comando, tem o intuito de mostrar uma lista de regiões por ordem alfabética, que tem pessoas ainda doentes. Os primeiros dois ciclos for são encadeados, mas não possuem uma dependência de complexidade entre eles, sendo cada um ***O(n).*** O segundo conjunto de ciclos for encadeados(), já possuem uma dependência entre eles elevando a complexidade algorítmica deste comando para ***O(n²).*** Como escolhemos o pior cenário a complexidade algorítmica deste comando é ***O(n²).***

Neste comando, contemos uma complexidade quadrática (***O(n²)***), devido a ser constituído por um ciclo for principal, que possui outro **ciclo for** encadeado, ambos condicionados por variantes distintas lineares. Também possui um listGet (***O(1)***), mas este não influencia, pois pondera-se a pior complexidade.

Dentro deste ciclo também é executado listAdd, que é composto por uma complexidade linear. Aqui temos duas complexidades lineares que embora estejam encadeadas, não estão dependentes uma da outra (por exemplo o listAdd é efetuado com a mesma complexidade, independentemente do tamanho da lista). A complexidade deste algoritmo é linear O(n).

O(n^2)

Report ->

Este comando, tem o intuito de criar um ficheiro report.txt e guardar neste a taxa de mortalidade e taxa de incidência, total e por região. Este possui vários ciclos for encadeados, mas que não possuem uma dependência de complexidade entre eles, sendo cada um ***O(n).*** Como escolhemos o pior cenário a complexidade algorítmica deste comando é ***O(n).***

O(n^2)