**OBSERVACIONES DEL LA PRACTICA**

Lindsay Vanessa Pinto Morato Cod. 202023138

José Daniel Montero Cod. 202012732

**CONTENIDO**

[EQUIPO DE PRUEBAS 1](#_Toc71306532)

[CONTROLES EN LAS PRUEBAS Y RESULTADOS 2](#_Toc71306533)

[COMPLEJIDAD 2](#_Toc71306534)

[ANÁLISIS DE CONSUMO DE MEMORIA Y TIEMPO 5](#_Toc71306535)

[CONCLUSIONES 8](#_Toc71306536)

**LISTA DE TABLAS**

[Table 1. Especificaciones computador 1](#_Toc69325431)

[Table 2. Datos de tiempo por requerimiento discriminados según el reto 2](#_Toc69325432)

[Table 3. Datos de uso de memoria por requerimiento discriminados según el reto 2](#_Toc69325433)

**LISTA DE ILUSTRACIONES**

[Figure 2. Comparación de rendimiento de tiempo entre los requerimientos de los dos retos 3](#_Toc69325326)

[Figure 3. Comparación de rendimiento de memoria entre los requerimientos de los dos retos 4](#_Toc69325327)

# **EQUIPO DE PRUEBAS**

Table 1. Especificaciones computadores

|  |  |
| --- | --- |
|  | Máquina 1 |
| Marca | Computador Huawei Matebook D14 |
| Procesadores | AMD Ryzen 5 3500U with Radeon Vega Mobile Gfx 2.10GHZ |
| Memoria RAM (GB) | 8,00GB (6.94 utilizable) |
| Sistema Operativo | Windows 10 Home Single languaje |
| Tipo de sistema | Sistema operativo de 64bits, procesador x64 |

|  |  |
| --- | --- |
|  | Máquina 2 |
| Marca |  |
| Procesadores |  |
| Memoria RAM (GB) |  |
| Sistema Operativo |  |
| Tipo de sistema |  |

# **CONTROLES EN LAS PRUEBAS Y RESULTADOS**

## **COMPLEJIDAD**

|  |  |  |
| --- | --- | --- |
| **Requerimiento** | | **O(n)** |
| 1 | Encontrar cantidad de clusteres dentro de la red de cables submarinos |  |
| 2 | Encontrar landing points que sirven como punto de interconexión a más cables |  |
| 3 | Encontrar la ruta mínima para enviar información entre dos países |  |
| 4 | Identificar la infraestructura crítica para garantizar el mantenimiento preventivo |  |
| 5 | Conocer el impacto que tendría el fallo de un determinado landing point |  |

A continuación se muestra el análisis detallado de estas complejidades

**REQUERIMIENTO 1: Encontrar cantidad de clusteres dentro de la red de cables submarinos**

|  |  |
| --- | --- |
| def findClusters(analyzer, landingPoint1, landingPoint2):  scc = kos.KosarajuSCC(analyzer['connections'])  vertexAId = me.getValue(mp.get(analyzer['landingPointNames'], landingPoint1))  vertexBId = me.getValue(mp.get(analyzer['landingPointNames'], landingPoint2))  lpA = me.getValue(mp.get(analyzer['landingPoints'], vertexAId))  lpB = me.getValue(mp.get(analyzer['landingPoints'], vertexBId))  vertexA = lt.getElement(lpA['vertices'], 1)  vertexB = lt.getElement(lpB['vertices'], 1)  return kos.connectedComponents(scc), kos.stronglyConnected(scc, vertexA, vertexB) | O (V+E) -> lineal  O(1)  O(1)  O(1)  O(1)  O(1)  O(1) |
| Llamaremos a O(1) = C y se realizará suma de cada una de las complejidades  Resolviendo:  Por complejidad asintótica se tiene que la complejidad del algorítmo es de La cual es lineal | |

**REQUERIMIENTO 3: Encontrar la ruta mínima para enviar información entre dos países**

|  |  |
| --- | --- |
| def findShortestPath(analyzer, pais1, pais2):  country1 = me.getValue(mp.get(analyzer['countries'], pais1))  vertex1 = country1['vertex']  country2 = me.getValue(mp.get(analyzer['countries'], pais2))  vertex2 = country2['vertex']  search = djk.Dijkstra(analyzer['connections'], vertex1)  dist = djk.distTo(search, vertex2)  path = djk.pathTo(search, vertex2)  return path, dist | O(1)  O(1)  O(1)  O(1)  E log (v)  O(1) -> De acuerdo al algoritmo en la libreria  O(1) -> De acuerdo al algoritmo en la libreria |
| Llamaremos a O(1) = C y se realizará suma de cada una de las complejidades  Resolviendo:  Por complejidad asintótica se tiene que la complejidad del algorítmo es de La cual es lineal | |