

Enunciado

Sitios turísticos

El departamento del Quindío es uno de los lugares turísticos más visitados en Colombia por su interminable lista de recursos naturales y de atractivos turísticos que motiva a pasar buenos momentos en familia. Además, la región reúne una serie de ventajas para hacer más grato su recorrido; entre otras cosas, por la cercanía entre los destinos y buenas y variadas alternativas para llegar a ellos. En el eje cafetero podemos encontrar una gran variedad de sitios aledaños como lo son: Salento, Córdoba, Calarca, entre otros.





Datos:

0. Genova
1. Pijao
2. Buenavista
3. Córdoba
4. La Tebaida
5. Armenia
6. Calarca
7. Montenegro

- 8. Quimbaya
- 9. Filandia
- 10. Salento

Sin embargo, se ha visto que en ocasiones los turistas desconocen las rutas y toman otras que hacen que gasten mucho más tiempo o que sencillamente no lleguen al destino que quieren, lo cual ha generado preocupación en la gobernación por el temor a que esto provoque que la cantidad de visitantes empiece a disminuir y se vea afectado en su economía.

Debido a esto, el gobernador del Quindío lo ha contratado a usted para desarrollar un programa que permita crear rutas para los viajeros con diferentes criterios, por ejemplo, la ruta más corta de un lugar a otro o rutas para llegar a un destino pasando por un lugar específico. Además, el programa debe permitir al usuario saber qué destinos se encuentran cerca a un lugar específico. Su programa debe estar en la condición de solucionar el problema con dos versiones de algoritmos de grafos vistos en clase.

TAD GRAFO

Grafo $G=(V,E)$ donde V es un conjunto no vacío de vértices y E es un conjunto de pares no ordenados de elementos distintos de V llamados aristas

{Inv: 2 vértices solo tienen una arista conectandolos}

- | | | |
|------------------|---|---------|
| • AñadirVertice | → | Boolean |
| • AñadirArista | → | Boolean |
| • BuscarRuta | → | List |
| • BuscarRuta2 | → | List |
| • BuscarCercanos | → | List |

AñadirVertice()

Retorna un booleano si el vértice se pudo agregar

{pre: El vértice debe ser diferente de null}

{post: True ∨ False}

AñadirArista(vertice1, vertice2)

Retorna un booleano si la arista se pudo agregar

{pre: La arista debe ser diferente de null y los vértices deben existir}

{post: True ∨ False}

BuscarRuta(vertice1, vertice2)

Busca la ruta más rápida y agrega los vértices por donde pasa a una lista

{pre: Los vértices deben existir}

{post: Se devuelve una lista con los lugares por donde pasa}

BuscarRuta2(vertice1, vertice2, verticeIntermedio)

Busca la ruta más rápida que pasa por un lugar específico y agrega los vértices por donde pasa a una lista

{pre: Los vértices deben existir}

{post: Se devuelve una lista con los lugares por donde pasa}

BuscarCercanos(vertexe)

Busca los vértices cercanos a un vértice específico

{pre: El vértice debe existir}

{post: Se devuelve una lista con los vértices cercanos}