DESCRIPCIÓN Y ESTIMACIÓN DE COMPLEJIDAD DE REQUERIMIENTOS  
  
  
  
  
  
  
  
  
  
  
  
Camilo Andres Salinas Martinez.

201714930.  
  
Laura Isabella Forero Camacho.  
  
201716278

UNIVERSIDAD DE LOS ANDES  
FACULTAD DE INGENIERÍA  
CARRERA DE INGENIERÍA DE SISTEMAS  
FEBRERO, BOGOTÁ D.C.  
2018

**Análisis de Requerimientos Funcionales**

|  |  |
| --- | --- |
| NOMBRE | 1A- Obtener cola de servicios en un intervalo de tiempo. |
| RESUMEN | Generar una cola con los servicios de taxi, ordenados cronológicamente, por un intervalo de tiempo dato. |
| ENTRADAS | |
| Fecha/hora inicial de consulta  Fecha/hora final de consulta | |
| RESULTADOS | |
| Cola de tipo Servicio | |

|  |  |
| --- | --- |
| NOMBRE | 2A- Buscar el taxi de una compañía con más servicios inició en un periodo de tiempo. |
| RESUMEN | Buscar el taxi de una compañía dada que más servicios inició en un periodo de tiempo dado |
| ENTRADAS | |
| Compañia.  Fecha/hora inicial de consulta  Fecha/hora final de consulta | |
| RESULTADOS | |
| Taxi con mayor número de servicios. | |

|  |  |
| --- | --- |
| NOMBRE | **3A-** Buscar la información completa de un taxi. |
| RESUMEN | Buscar la información completa de un taxi, a partir de su identificador, en un periodo de tiempo. |
| ENTRADAS | |
| Identificador de taxi | |
| RESULTADOS | |
| Nombre de su compañía.  Total de dinero ganado.  Total de de servicios prestados.  Total de de distancia recorrida.  Total de tiempo total de servicios. | |

|  |  |
| --- | --- |
| NOMBRE | **4A-** Dar lista de rangos de distancia recorrida. |
| RESUMEN | Retornar una lista de rangos de distancia recorrida, con todos los servicios de taxis servidos por las compañías, en un tiempo dado. |
| ENTRADAS | |
| Fecha/hora inicial de consulta  Fecha/hora final de consulta | |
| RESULTADOS | |
| Lista de rangos de distancia recorrida. | |

|  |  |
| --- | --- |
| NOMBRE | **1B**- Mostrar la información de las compañías de taxi. |
| RESUMEN | Mostrar el total de compañías que tienen al menos un taxi inscrito y el total de taxis que prestan servicio para al menos una compañía. generar la lista alfabética de compañías a las cuales aparecen inscritos los servicios de taxi de la fuente de datos de consulta. |
| ENTRADAS | |
| Ninguna | |
| RESULTADOS | |
| Nombre de la compañía  Número de taxis que tiene registrados la compañia. | |

|  |  |
| --- | --- |
| NOMBRE | **2B-** Taxi de compañía con mayor facturación. |
| RESUMEN | Buscar el taxi de una compañía dada que mayor facturación ha generado en un periodo de tiempo dado. |
| ENTRADAS | |
| Nombre compañia.  Fecha/hora inicial de consulta  Fecha/hora final de consulta | |
| RESULTADOS | |
| Taxi | |

|  |  |
| --- | --- |
| NOMBRE | **3B-I**nformación de una zona de la ciudad. |
| RESUMEN | Buscar la información completa de una zona de la ciudad en un periodo de tiempo dado. |
| ENTRADAS | |
| Zona de la ciudad. | |
| RESULTADOS | |
| El número total de servicios que se recogieron en la zona de consulta y terminaron en otra zona.  El total de servicios que iniciaron y terminaron en la misma zona de consulta.  El valor total pagado por los usuarios. | |

|  |  |
| --- | --- |
| NOMBRE | **4B-** Lista de zonas, con sus correspondientes servicios. |
| RESUMEN | Retornar una lista con todas las zonas de la ciudad (ordenadas por su identificador).Cada zona debe tener el total de servicios iniciados en dicha zona en un rango de fechas. |
| ENTRADAS | |
| Ninguna. | |
| RESULTADOS | |
| Lista con todas las zonas de la ciudad, en cada posición se incluyen los servicios de cada zona. | |

|  |  |
| --- | --- |
| NOMBRE | **1C-** Cargar. |
| RESUMEN | Cargar toda la información de una fuente de datos seleccionada por el usuario. (Se generan las estructuras necesarias). |
| ENTRADAS | |
| Fuente de datos. | |
| RESULTADOS | |
| Se generar los VO y las estructuras de datos. | |

|  |  |
| --- | --- |
| NOMBRE | **2C-** Top *X* de compañías que más servicios iniciaron en un periodo de tiempo dado. |
| RESUMEN | Identificar el top *X* de compañías que más servicios iniciaron en un periodo de tiempo dado por una fecha/hora inicial y una fecha/hora final. |
| ENTRADAS | |
| Número de elementos en top (valor X).  Fecha/hora inicial de consulta  Fecha/hora final de consulta | |
| RESULTADOS | |
| Lista de Compañías. | |

|  |  |
| --- | --- |
| NOMBRE | **3C-** Taxi más rentable de cada compañía. |
| RESUMEN | Buscar el taxi más rentable de cada compañía. El taxi más rentable de una compañía es aquel cuya relación de plata ganada y distancia recorrida en los servicios prestados es mayor. |
| ENTRADAS | |
| Nombre de la compañía. | |
| RESULTADOS | |
| Taxi. | |

|  |  |
| --- | --- |
| NOMBRE | **4C-** Pilas con información compacta. |
| RESUMEN | Para ello usted debe guardar en una pila todos los servicios generados por el taxi en orden cronológico, entre una hora inicial y una hora final, en una fecha determinada. Para comprimir la información, usted debe ir guardando los servicios reportados por el taxi, siempre y cuando la distancia acumulada de dichos servicios sea inferior a un valor dado. |
| ENTRADAS | |
| Id Taxi  Distancia máxima que contendrá cada pila.  Fecha/hora inicial de consulta  Fecha/hora final de consulta | |
| RESULTADOS | |
| Pilas con la información compactada. | |

**Análisis de Estructuras de Datos**

Nombre: Linked List

Descripción: Estructura de datos lineal compuesta de nodos relacionados ya sea por un enlace doble o simple.

Ventajas:

* Eliminación de datos sencilla.
* Inserción de nodos sencilla.
* Estructura de fácil escalamiento, ya que la memoria usada para su implementación puede variarse mientras el programa corre.
* No es necesario definir un tamaño inicial.
* Puede ser usada en pilas y colas.

Desventajas:

* Dificil acceso aleatorio
* Ciertos algoritmos de ordenamiento y busqueda se hacen complicados o imposibles de ejecutar (búsqueda binaria, Timsort).
* Ocupa más espacio en memoria que un arreglo, ya que tiene que guardar los apuntadores entre nodos (enlaces)

Casos de uso:

* Requerimientos:
  + 3A: Creación de lista de servicios.
  + 4A: Lista de rangos, y rangos en si.
  + 1B: Lista de compañías con taxis inscritos, Lista ordenada alfabéticamente.
  + 3B: Lista de servicios por zonas.
  + 4B: Lista de zonas.
  + 2C: Lista Top X.

Complejidad:

Hemos elegido para busqueda sobre esta estructura una búsqueda lineal.

* La cual debido a la estructura de datos el peor caso daría una complejidad O(N), ya que sería el caso en el que sea el último elemento en la lista, o un elemento no existente.

Para el ordenamiento elegimos MergeSort:

* Debido a que en su peor caso nos da una complejidad O(NLogN), debido al funcionamiento de este algoritmo, no existen casos de mejor o peor rendimiento.

Justificación:

Se siguió los lineamientos dados por los requerimientos entregados en el proyecto, esta fue la razón de uso de esta estructura lineal (Aun se esta en elección si se usará una lista sencilla o doble, Sin embargo se implementaron ambos casos).

Nombre: Cola

Descripción: Estructura de datos secuencial, conformada por nodos, en la cual los datos entran por un extremo y salen por el otro, permitiendo saber la duración relativa entre elementos (Ej: el ultimo elemento lleva menos tiempo dentro de la estructura que el primero)

Ventajas:

* Inserción de nodos sencilla.
* Estructura de fácil escalamiento, ya que la memoria usada para su implementación puede variarse mientras el programa corre.
* No es necesario definir un tamaño inicial.

Desventajas:

* Difícil acceso a elementos intermedios, imposible hacerse sin modificar la estructura.
* Es necesario rehacer la estructura para ordenarla, o ingresar los datos ordenados inicialmente.
* Ocupa más espacio en memoria que un arreglo, ya que tiene que guardar los apuntadores entre nodos (enlaces)

Casos de uso:

* Requerimientos:
  + 1C: Cola de Servicios.

Complejidad:

* Debido a las dificultades de acceso y distintas características de la cola, usaremos una lista previa para el ordenamiento, y para la búsqueda se usara una búsqueda lineal que descarte elementos no necesitados.
  + Por lo tanto la busqueda sera de O(N).

Justificación:

Se siguió los lineamientos dados por los requerimientos entregados en el proyecto, esta fue la razón de uso de esta estructura lineal.

Nombre: Pila

Descripción: Estructura de datos secuencial, conformada por nodos, en la cual los datos entran y salen por un extremo, es decir, el último en entrar es el primero en salir. Sus métodos básicos son push (insertar) y pop(obtener el último insertado).

Ventajas:

* Ingreso de datos sencillo.
* estructura de almacenamiento eficiente.

Desventajas:

* Acceso aleatorio imposible sin modificar la estructura al azar.
* Ordenamientos y búsquedas con complejidad máxima.

Casos de uso:

* Requerimientos:
  + 4C: Almacenamiento de servicios.

Complejidad:

* Debido a las dificultades de acceso y distintas características de la cola, usaremos una lista previa para el ordenamiento, y para la búsqueda se usara una búsqueda lineal que descarte elementos no necesitados.
  + Por lo tanto la busqueda sera de O(N).

Justificación:

Se siguió los lineamientos dados por los requerimientos entregados en el proyecto, esta fue la razón de uso de esta estructura lineal.

Analisis de Algoritmos:

MergeSort:

Es un algoritmo de ordenamiento externo que se basa en la combinación de dos matrices ordenadas y luego fusiona sus resultados.

Complejidad : O( NLogN).