Duración: 30 minutos

Este test está diseñado para evaluar tu fluidez en la codificación y habilidades para resolver problemas. Las soluciones deben centrarse en resolver el problema dado, producir código limpio y funcional, Puedes usar cualquier lenguaje de programación y no serás penalizado por errores menores de sintaxis.

**Problema 1:**

Dada una cadena de texto, escribe una función que devuelva un valor booleano que indique si es un palíndromo o no. Se dice que una cadena es un palíndromo si el reverso de la cadena es el mismo que la cadena. Por ejemplo, "malayalam" es un palíndromo, "geek" no lo es.

Lenguaje que estás usando: ……………….

Escribe tu código a continuación:

**Problema 2:**

Escribe una función que tome una Lista/Arreglo 2D de transacciones y devuelva una lista de IDs de

transacciones que son fraudulentas. Cualquier transacción mayor o igual a 10000 se considera fraudulenta. Cualquier transacción de la misma tarjeta de crédito en una ciudad diferente dentro de los 30 minutos también se considera fraudulenta.

Entrada: Una Lista/Arreglo 2D de transacciones, donde cada registro de transacción tiene un ID de transacción (entero), ID de tarjeta de crédito (entero), cantidad de transacción (doble), ciudad (cadena) y tiempo en minutos (entero). Puedes asumir que todas las transacciones ocurren el mismo día.

Ejemplo:

Entrada:

[

[1, 1000, 500.00, "Vadodara", 0],

[2, 1000, 500.00, "Mumbai", 5],

[3, 1001, 500.00, "Mumbai", 10],

[4, 1001, 10000.00, "Mumbai", 10]

]

Salida:

[2, 4]

Las transacciones 2 y 4 deben considerarse fraudulentas. La transacción 2 ocurrió dentro de los 30 minutos de la transacción 1 con el mismo ID de tarjeta de crédito (1000) y en una ciudad diferente. La transacción 4 tiene un monto mayor o igual a 10000.

Lenguaje que estás usando: …………….

Escribe tu código a continuación:

**Problema 3:**

Un sistema ferroviario está haciendo un seguimiento de los tiempos de viaje de los clientes entre diferentes estaciones. Están usando estos datos para calcular el tiempo promedio que toma viajar de una estación a otra.

Implementa una clase llamada `UndergroundSystem` con las siguientes funciones:

* `void checkIn(int id, string stationName, int t)`: Un cliente con un ID de tarjeta igual a id se registra en la estación stationName a la hora t. Un cliente solo puede registrarse en un lugar a la vez.
* `void checkOut(int id, string stationName, int t)`: Un cliente con un ID de tarjeta igual a id se registra en la estación stationName a la hora t.
* `double getAverageTime(string startStation, string endStation)`: Devuelve el tiempo promedio que toma viajar de startStation a endStation.

El tiempo promedio se calcula a partir de todos los tiempos de viaje anteriores de startStation a endStation que ocurrieron directamente, es decir, un check-in en startStation seguido de un check-out en endStation. El tiempo que toma viajar de startStation a endStation puede ser diferente del tiempo que toma viajar de endStation a startStation. Habrá al menos un cliente que haya viajado de startStation a endStation antes de que se llame a getAverageTime. Puedes asumir que todas las llamadas a los métodos checkIn y checkOut son consistentes. Si un cliente se registra a la hora t1 y se da de baja a la hora t2, entonces t1 < t2. Todos los eventos ocurren en orden cronológico.

Lenguaje que estás usando:

Escribe tu código a continuació: