Corto #5

Santiago Pereira - 22318
Gabriela Mazariegos - 22513

Diseño de idea:

La simulación va a mostrar/simular una cafetería en una hora ocupada, cuando llegan muchos clientes, hacen pedidos y son atendidos en diferentes estaciones. La cafetería va a tener tres estaciones que van a trabajar al mismo tiempo:

Estación de Pedidos	En esta estación se van tomando todos los pedidos de los clientes
Estación de Preparación	Prepara los pedidos/órdenes de los clientes
Estación de Cobro	Como su nombre lo dice, realizará el cobro y entrega de pedidos

Aplicación de Conceptos OpenMP

- parallel for:
  - Nos va a servir para simular que varios clientes llegan al mismo tiempo.
  - Cada vuelta del ciclo va a ser un cliente diferente con su ID y gustos propios.
  - Va a permitir atender a varios clientes a la vez.
- sections:
  - Se usa para que las tres estaciones trabajen de forma independiente:
    - Toma de pedidos
    - Preparación de bebidas
    - Cobro y entrega
  - Cada estación es un proceso separado que puede correr en paralelo.
- shared:
  - *inventario*: Array con cantidades de ingredientes (café, leche, azúcar, etc.)
  - *caja_registradora*: Total de dinero recaudado
  - *cola_pedidos*: Queue de pedidos pendientes
  - *cola_bebidas*: Queue de bebidas listas para entrega
  - *num_clientes_atendidos*: Contador global de clientes procesados
- firstprivate:
  - *id_cliente*: Identificador único de cada cliente
  - *tipo_bebida_preferida*: Preferencia inicial de cada cliente
  - *dinero_cliente*: Cantidad de dinero que trae cada cliente
  - *tiempo_llegada*: Momento en que llega cada cliente
- reduction:
  - *suma (+)*: Para calcular ventas totales, tiempo total de servicio
  - *max*: Para encontrar el tiempo máximo de espera
  - *min*: Para encontrar el tiempo mínimo de servicio
  - *count*: Para contar diferentes tipos de bebidas vendidas

Variables Compartidas y Privadas

- Compartidas:
  - *inventario*[MAX_INGREDIENTES]: Inventario global de ingredientes
  - *caja_registradora*: Dinero total recaudado
  - *cola_pedidos*: Cola global de pedidos
  - *estadisticas_ventas*[MAX_BEBIDAS]: Contadores de cada tipo de bebida
- Privadas:
  - *id_cliente*: Único para cada thread osea el cliente
  - *tipo_bebida*: Elección específica del cliente
  - *tiempo_servicio*: Tiempo que toma atender a este cliente específico
  - *dinero_pagado*: Cantidad que paga cada cliente individual

Pseudocódigo:

```
// Declaración de variables globales
SHARED:
   int inventario[5] = {100, 100, 100, 100, 100}  // café, leche, azúcar, hielo, fruta
   float caja_registradora = 0.0
   int cola_pedidos[MAX_COLA]
   int cola_bebidas_listas[MAX_COLA]
   int estadisticas_ventas[4] = {0}  // café, té, smoothie, frappé

REDUCTION VARIABLES:
   float ventas_totales = 0.0
   int tiempo_total_servicio = 0
   int max_tiempo_espera = 0
   int total_clientes_satisfechos = 0

MAIN FUNCTION:
   inicializar_cafeteria()

   // Simulación paralela de clientes llegando
   #pragma omp parallel for firstprivate(id_cliente, dinero_cliente)
                        shared(inventario, caja_registradora)
                        reduction(+:ventas_totales, total_clientes_satisfechos)
                        reduction(max:max_tiempo_espera)
   FOR i = 0 to NUM_CLIENTES:
      id_cliente = i
      dinero_cliente = random(50, 200)  // dinero que trae cada cliente
      tipo_bebida = elegir_bebida_random()
      tiempo_inicio = get_time()

      // Procesar cliente individual
      procesar_cliente(id_cliente, tipo_bebida, dinero_cliente)

      tiempo_servicio = get_time() - tiempo_inicio
      ventas_totales += calcular_precio(tipo_bebida)
```

```
    max_tiempo_espera = max(max_tiempo_espera, tiempo_servicio)
    total_clientes_satisfechos++
  END FOR

  // Operaciones paralelas de las estaciones
  #pragma omp parallel sections shared(cola_pedidos, cola_bebidas_listas, inventario)
  {
    #pragma omp section  // Estación de pedidos
    {
      WHILE hay_clientes_esperando():
        tomar_pedido()
        agregar_a_cola_pedidos()
      END WHILE
    }

    #pragma omp section  // Estación de preparación
    {
      WHILE hay_pedidos_pendientes():
        pedido = tomar_de_cola_pedidos()
        preparar_bebida(pedido)
        agregar_a_cola_bebidas_listas()
        actualizar_inventario()  // Critical section
      END WHILE
    }

    #pragma omp section  // Estación de cobro
    {
      WHILE hay_bebidas_listas():
        bebida = tomar_de_cola_bebidas()
        procesar_pago()
        entregar_bebida()
        #pragma omp critical
        {
          caja_registradora += precio
          estadisticas_ventas[tipo_bebida]++
        }
      END WHILE
    }
  }

  mostrar_estadisticas_finales()
END MAIN

FUNCTION procesar_cliente(id, tipo_bebida, dinero):
  // Variables privadas para cada cliente
  int tiempo_preparacion = obtener_tiempo_preparacion(tipo_bebida)
  float precio = obtener_precio(tipo_bebida)

  IF dinero >= precio AND hay_ingredientes(tipo_bebida):
    simular_preparacion(tiempo_preparacion)
    RETURN SUCCESS
  ELSE:
    RETURN FAILED
  END IF
```

```
END FUNCTION

FUNCTION preparar_bebida(tipo_bebida):
    SWITCH tipo_bebida:
        CASE CAFE:
            usar_ingredientes(cafe=1, leche=1)
            tiempo = 30  // segundos
        CASE TE:
            usar_ingredientes(te=1)
            tiempo = 20
        CASE SMOOTHIE:
            usar_ingredientes(fruta=2, hielo=1)
            tiempo = 60
        CASE FRAPPE:
            usar_ingredientes(cafe=1, hielo=2, leche=1)
            tiempo = 45
    END SWITCH

    // Critical section para modificar inventario
    #pragma omp critical
    {
        actualizar_inventario(ingredientes_usados)
    }

    simular_tiempo_preparacion(tiempo)
END FUNCTION
```

Implementación y Ejecución

https://github.com/nancygmm/corto5-paralela.git