Universidad Rafael Landívar

Facultad de Ingeniería

Ingeniería en Informática y Sistemas

Laboratorio de Pensamiento computacional, Sección 04

Docente: Ing.

**Proyecto 2-A**

“Sistema de gestión de estacionamiento”

Estudiante: Galindo Girón, Luis Rodrigo

Carné: 1174225

Guatemala, día 07 de mayo de 2025

Contents

[Acciones del programa: 3](#_Toc197550288)

[Datos de entrada: 4](#_Toc197550289)

[Variables: 5](#_Toc197550290)

[Métodos: 6](#_Toc197550291)

[Clases: 7](#_Toc197550292)

[Condiciones y cálculos: 8](#_Toc197550293)

[Condiciones: 8](#_Toc197550294)

[Calculos: 8](#_Toc197550295)

[Diagrama de flujo: 9](#_Toc197550296)

[Diagrama de flujo de funciones 10](#_Toc197550297)

# Acciones del programa:

* Inicio del programa:

Antes de iniciarse el programa se crea la clase de vehículos que se usará en el programa y todas las funciones pertinentes a este último.

Al iniciarse el programa se le solicita al usuario las filas y columnas necesarias para construir las matrices que se usaran para representar el estacionamiento. Tras ello se solicitará la cantidad de parqueos de moto y SUV que habrá en el estacionamiento y a partir de eso el resto de los estacionamientos serán de tipo Sedan.

* Menú de opciones:

Tras haber creado la matriz y haber designado los tipos de estacionamiento en el parqueo, se le mostrara al usuario un menú de opciones. Escribiendo el número que represente la opción seleccionada, accederá a las siguientes opciones:

1. Ingresar un vehículo manualmente:

En esta opción se solicitara al usuarió ingresar manualmente los datos de un vehículo; los cuales se componen de marca, color, placa (Debe poseer 6 caracteres en mayuscula), tipo de vehículo y la hora de entrada del vehículo (Un número del 6 al 20).

Tras ello se mostrara un mapa del parqueo con los parqueos ocupados o que no puede usar su vehículo, se le permitira elegir el estacionamiento donde se parqueara al este ingresar el código del respectivo parqueo, verificando que el espacio no este ocupado o no pueda ser usado por el usuario.

1. Ingresar un lote de vehículos:

Al seleccionar esta opción el usuario colocara la cantidad de vehículos que desea ingresar y las características de estos se definiran aleatoriamente de los siguientes datos. Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Cada vehículo se ingresara a la matriz del parqueo en caso de que haya espacio disponible. Al finalizar se mostrara una lista de las placas y codigos de estacionamiento donde estan.

1. Encontrar un vehículo:

El usuario debera ingresar el código del parqueo en el que se encuentra el vehículo y el sistema devolvera todos los datos del vehículo en cuestión.

1. Retirar un vehículo

Se retirara el vehículo de la ubicación seleccionada por el usuario y se cobrara el parqueo según el tiempo de estadía del vehículo, el cual será un número aleatorio que se obtendrá de la resta de 24 menos la hora de entrada. Con esto se calculara el monto a pagar y el usuario podrá seleccionar la opción de pagar con tarjeta / sticker recargable o en efectivo. En el caso de pagar en efectivo se deberá calcular el vuelto del usuario según la cantidad ingresada al cajero. El cajero solo acepta y devuelve billetes de 5, 10, 20, 50 y 100 quetzales.

1. Salir

El usuario cerrara el programa en caso de que se escriba

# Datos de entrada:

* Int pisos: El usuario ingresara la cantidad de pisos que tendrá el parqueo, esto se usara para la cantidad de filas que tendrá la matriz que representara el parqueo.
* Int parqueos: el usuario ingresa la que será la cantidad de parqueos que poseerá cada piso, se usaran como las columnas de la matriz.
* Int estacionamientoMotos: numero de estacionamientos tipo moto que poseerá el parqueo.
* Int estacionamientoSUV: numero de estacionamientos tipo SUV que poseerá el parqueo.
* Int Opción: Se ingresa el número de opción del menú principal del programa el cual ejecutara la acción seleccionada por el usuario a traves de un switch case.
* String miMarca: El usuario ingresa la marca del vehículo que ingresara al parqueo.
* String miColor: El usuario ingresa el color del vehícuolo que ingresara al parqueo.
* String miPlaca: El usuario ingresa un string de 6 caracteres el cual sera la placa del vehículo que ingresara.
* String miTipo: El usuario ingresara el tipo del vehículo que ingresara al parqueo.
* Int miHoraIngreso: Se ingresara la hora de entrada, siendo esta un número entre 6 y 20.
* String Codigo: El usuario ingresara un string de 2 caracteres que representara el código del parqueo.
* Int numVehiculos: Dato que ingresado por el usuario que representa la cantidad de vehiculos que quiere generar en un lote.
* String placaBuscar: Para buscar un vehiculo determinado, el usuario ingresa la placa para buscarlo en el parqueo.
* String codigoRetirar: cadena del codigo del parqueo del vehiculo que el usario desea retirar del parqueo.
* String opcion2: Se le pide al usuario ingresar la letra “T” o “E” para seleccionar como pagara el parqueo el usuario.
* Int monto: El usuario ingresa un monto que sea multiplo de 5 con el que pagara el parqueo.

# Variables:

1. String tipoMoto, tipoSUV, tipoSedan: Variables usadas para almacenar una cadena del respectivo tipo de vehículo al que se refieren.
2. Int filaX, columnaX = 0: Variable que se usara para almacenar la siguiente fila y columna en la que se empezara a asignar un tipo de vehículo en la matriz de tipos del parqueo.
3. Int parqueos, pisos = Read(): variables usadas para pedir al usuario la cantidad de filas y columnas que tendran las matrices del parqueo.
4. vehiculo [ ] torreParqueos = new vehiculo [pisos,parqueos]: Matriz donde se almacenaran los objetos de tipo vehículo.
5. string [ ] matrizTipos = new string [pisos,parqueos]: Matriz que se usara para indicar el tipo de vehículo permitido en los estacionamientos.
6. Int estacionamientosMoto, estacionamientosSUV = Variables que almacenarán la cantidad de estacionamientos de cada tipo que poseerá el parqueo.
7. Int estacionamientosSedan = (pisos\*parqueos)-estacionamientosMoto -estacionamientoSUV : Usando la cantidad de estacionamientos del parqueo y restandole la cantidad de estacionamientos de los demás tipos de vehículo, se obtiene el número de estacionamientos tipo Sedan.
8. Int opción = Read(): Esta variable toma el número ingresado por el usuario y la evalua con un switch case que permite seleccionar una opción dependiendo del número que ingreso.
9. String miMarca, miColor, miPlaca, miTipo; Int miHoraIngreso = Read(): Variables que se usan para almacenar los atributos del objeto Vehículo que se creara para ingresarse a el parqueo.
10. String codigo = Read(): El usuario ingresara el codigo del estacionamiento donde desea estacionarse, este se traducira a números para usarse como indice de la matriz.
11. int [ ] miIndice = LetraANum(codigo[0]): Variable que almacena la lista con el indice de la matriz donde se estacionara el vehículo.
12. int numVehículos = Read(): Variable que obtendra del usuario la cantidad de vehículos que se desean ingresar en un lote de vehículos.
13. String placaBuscar = Read(): Variable usada para almacenar la placa del vehículo buscado por el usuario.
14. Int códigoRetirar = Read(): Variable que pide al usuario el código del estacionamiento del vehículo que desea retirar para luego convertirlo en un índice.

int pago = torreParqueos[indiceRetiro[0],indiceRetiro[1]].pagoParqueo(): Usando el método de la clase Vehículo, se obtiene la cantidad a pagar del parqueo.

1. Int [ ] indiceRetiro = LetraANum(codigoRetirar): Arreglo que almacena el índice de la matriz del vehículo que se desea retirar.
2. string opcion2 = Read(): Variable que se usa para evaluar las opciones “T” y “E” en un switch case.
3. Int monto = Read(): El usuario ingresa la cantidad que usara para pagar el parqueo.

## Métodos:

1. FUNCTION String numALetra():

Función que traduce un número a una letra para usarla en el código. Usando el número este se usara como índice para un arreglo de letras en la que se seleccionara la letra según el índice.

1. FUNCTION int [] letraANum():

Esta función se usa para pasar un código de string de 2 caracteres a una lista con 2 ints, que se usaran como índices dentro de la matriz torreParqueos. Se usa una lista de letras para encontrar la letra del código y converitrla en el índice de la respectiva letra en la lista. Al ser un número este se vuelve un int y se le resta 1 para poder usarse como índice en la matriz.

1. PROCEDURE mostrarMatrizVisual():

Se itera a traves de la matriz torreParqueos usando bucles For y se traduce cada índice a un código de parqueo. En caso de no ser del tipo de vehículo que se estacionara, se coloca una X.

1. PROCEDURE asignarTipoParqueo():

Usando bucles for se itera sobre la matriz asignada y se va colocando el tipo de vehículo seleccionado por la cantidad de espacios del tipo. Se usa un bucle while con un contador para solo colocar la cantidad adecuada de tipos y luego se designa el índice de la columna específcia donde se termino de iterar para cuando se quiera volver a usar el método y continuar desde donde se dejo.

1. PROCEDURE ingresarLote():

Usando un número determinado por el usuario, se crean x cantidad de objetos Vehículo, los cuales tienen atributos aleatorios seleccionados de un rango específico de datos. Tras ello se crean índices aleatorios para ingresar los datos a la matriz.

1. PROCEDURE buscarVehículo():

Se le brinda una placa al método para que este itere usando bucles for por la matriz, evaluando el atributos placa de cada objeto que encuentre hasta que encuentre la placa igual a la ingresada, con ello se imprimirá la información y el código del parqueo en el que esta.

1. FUNCTION string calculoVuelto():

Usando el método propio de la clase Vehículo pagoParqueo(), se obtiene el monto a pagar e iterando sobre una lista con todas las cantidades de billetes disponibles, se divide el vuelto hasta encontrar las cantidades necesarias de cada billete para brindar el vuelto.

1. Vehículo FUNCTION pagoParqueo():

Se resta la hora de entrada a 24, según el rango de datos en el que este esa resta, se devolverá un monto diferente que el usuario deberá pagar.

1. Vehículo PROCEDURE mostrarInfo():

Muestra todos los atributos de un objeto de la clase Vehículo de manera amigable.

## Clases:

* Vehículo:
  + Atributos:
    - String marca
    - String color
    - String tipo
    - String placa
    - Int horaEntrada
* Metodos:
  + - FUNCTION int calcularPago()
    - PROCEDURE mostrarInfor()

# Condiciones y cálculos:

## Condiciones:

* En el switch case de opción, se pueden elegir 5 opciones:
  + Case 1:

El usuario ingresara en las variables miMarca, miColor, miPlaca, miTipo y miHoraIngreso los datos necesarios para los atributos de la clase Vehículo y así crear un objeto con esta, el cual ingresara a la matriz torreParqueos ingresando un código en la variable codigo, el cual sera traducido a un índice para la matriz con la función letraANum().

* Si el índice seleccionado por el usuario esta vacío en la matriz torreParqueos y tiene el mismo tipo que la variable miTipo, se asignara el objeto a ese índice de la matriz; sino se volverá a pedir al usuario un estacionamiento válido.
  + Case 2:

El usuario ingresara un número el cual determinara la cantidad de vehículos que se ingresaran a la función ingresarLote(), esta función creara objetos de la clase Vehículo haciendo uso de valores predeterminados que se seleccionaran aleatoriamente y también ingresaran en espacios de parqueo aleatorios.

* + Case 3:

En el case 3, se ingresa la placa de un vehículo y usando la función buscarVehículo(), se itera por la matriz torreParqueos buscando el objeto con la placa correspondiente; tras ello se muestra la información del vehículo y donde se encuentra estacionado.

* + Case 4:

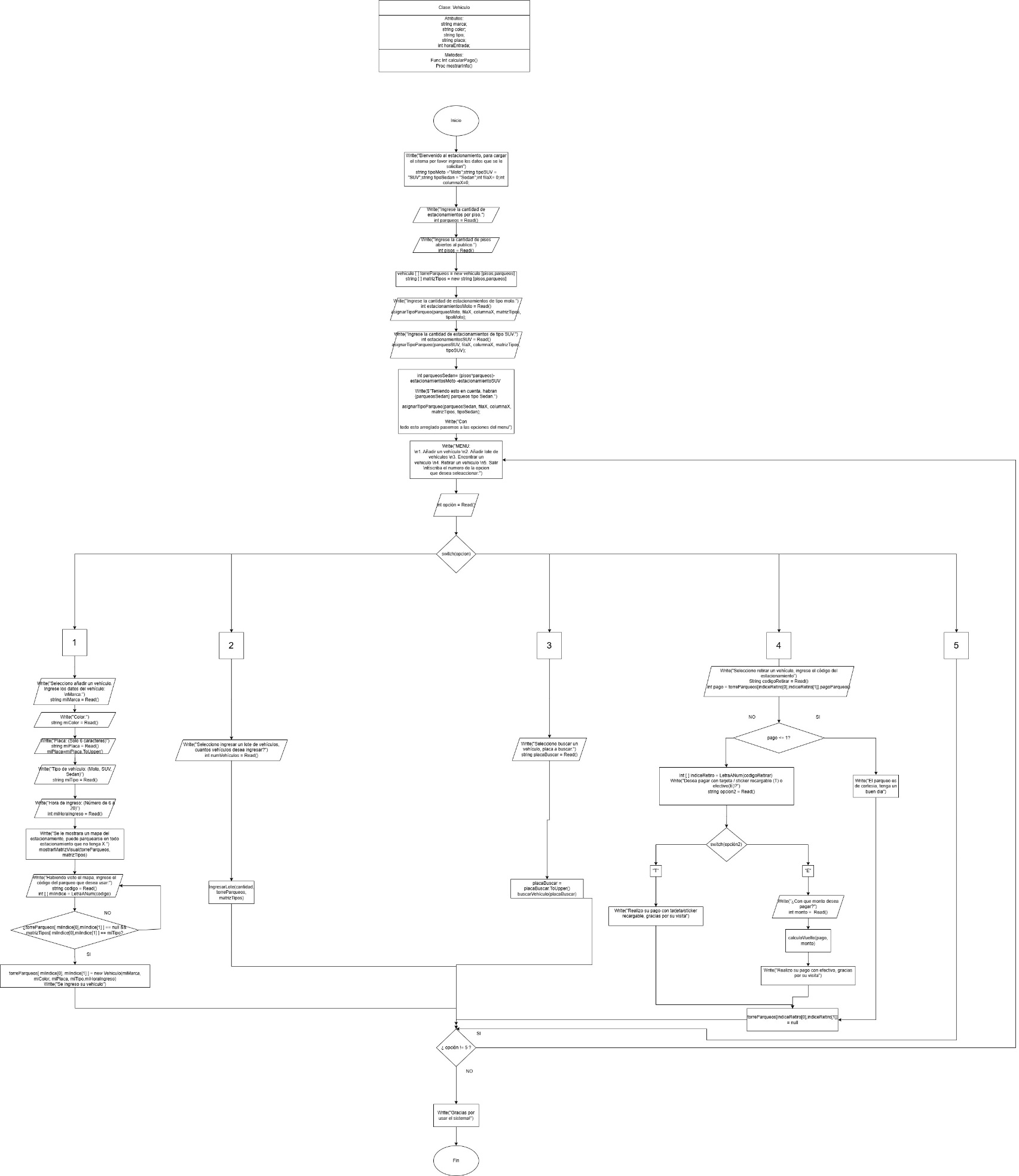
Esta opción permite retirar un vehículo del parqueo, y cobrarle según su tiempo de estadía haciendo uso de los métodos pagoParqueo() y calcularVuelto(). Tras ello se retira el vehículo del espacio correspondiente.

* + - Se crea otro switch case con las opciones “T” y “E”
      * Case “T”: Se usa Write() para indicar que el usuario ha pagado y el programa continua.
      * Case “E”: Usando el monto con el que pagara el usuario y el costo del parqueo, se calcula la cantidad de billetes de cada valor que se le debe retornar al usuario.

## Calculos:

* Cantidad de estacionamientos Sedan: (pisos\*parqueos)-estacionamientosMoto -estacionamientoSUV
* Horas de estadía: 24 – horaEntrada
* Cantidad de billetes: numBilletes = Vuelto / Billetes[i]

# Diagrama de flujo:

Diagrama de flujo del Main()

## Diagrama de flujo de funciones

Diagrama, Escala de tiempo

El contenido generado por IA puede ser incorrecto.