75.41 - Algoritmos y Programación II

Cátedra Ing. Patricia Calvo - 2do cuatrimestre 2019

Trabajo Práctico 2: CIUDAD DIGITAL EN MOVIMIENTO V1.0

Objetivo

Generar una pieza de software que simule viajes por la ciudad a partir de datos de los archivos de texto del gobierno de la ciudad.

Enunciado

Se debe descargar el siguiente archivo con información de la las estaciones de metrobus del gobierno de la ciudad:

Estaciones de ferrocarril

https://data.buenosaires.gob.ar/dataset/estaciones-ferrocarril/archivo/102.1

Estaciones de metrobús

https://data.buenosaires.gob.ar/dataset/metrobus/archivo/143.1

Estaciones de subte:

https://data.buenosaires.gob.ar/dataset/95d90f6f-7fbd-4c12-90ae-e1220182b266/archivo/c9b9628b-6ca5-4867-bf9d-ad11997f951f

Garage:

https://data.buenosaires.gob.ar/dataset/garajes-comerciales/archivo/bb4655bb-32c7-482a-a82c-fe5641d38d54

Colectivos:

https://data.buenosaires.gob.ar/dataset/colectivos-paradas/archivo/d0e599d2-3e78-4fb2-9255-30a2be0525f8

Donde cada archivo contiene entre otros datos, la coordenada de ubicación. El objetivo del TP es que el usuario pueda ingresar un origen (X1 Y1) y un destino (X2, Y2) y que el TP le resuelva el recorrido.

La primera tarea que debe hacer el grupo es un conjunto de hipótesis para limitar el alcance del TP. Por ejemplo, limitar la cantidad de intercambio de transportes que tiene el recorrido elegido, es decir, que primero busca un viaje directo y luego un viaje con un cambio de transporte, pero no más. Otra hipótesis podría ser la distancia máxima entre “paradas” para hacer la combinación.

Las hipótesis deben ser aprobadas por el ayudante a cargo del grupo.

Interfaz de usuario

Toda la interfaz de usuario debe estar basada en texto. El estado del tablero tiene que mostrarse usando caracteres dispuestos en filas y columnas. Por ejemplo, usando ‘ ’ para representar un casillero vacío, ‘X’ para representar un casillero ocupado por un jugador y ‘O’ para el otro.

No es necesario que se limpie la pantalla, simplemente escribir el estado del tablero luego de cada jugada.

Luego de cada resultado se deberá dibujar en un bitmap con el recorrido según la librería Easybmp del campus.

Cuestionario

Responder el siguiente Cuestionario:

1) ¿Que es un SVN?
2) ¿Que es “GitHub”?
3) ¿Que es “collabnet” y “Tigris”?

Normas de entrega

Trabajo práctico individual: 5 persona. Cada grupo deberá ponerse un nombre.

Reglas generales: respetar el Apéndice A.

Se deberá subir un archivo comprimido al campus, en un link que se habilitará para esta entrega. Este archivo deberá tener un nombre formado de la siguiente manera:

Nombre del Grupo-TP2.zip

Deberá contener los archivos fuentes (no los binarios), el informe del trabajo realizado, las respuestas al cuestionario, el manual del usuario y el manual del programador (Todo en el mismo PDF).

La fecha de entrega vence el día lunes 31/10/19 a las 23.59hs.

Se evaluará: funcionalidad, eficiencia, algoritmos utilizados, buenas prácticas de programación, modularización, documentación, gestión de memoria y estructuras de datos.

Apéndice A

1) Usar las siguientes convenciones para nombrar identificadores.
    a) Clases y structs: Los nombres de clases y structs siempre deben comenzar con la primera letra en mayúscula en cada palabra, deben ser simples y descriptivos. Se concatenan todas las palabras. Ejemplo: Coche, Vehiculo, CentralTelefonica.
    b) Métodos y funciones: Deben comenzar con letra minúscula, y si está compuesta por 2 o más palabras, la primera letra de la segunda palabra debe comenzar con mayúscula. De preferencia que sean verbos. Ejemplo: arrancarCoche(), sumar().
    c) Variables y objetos: las variables siguen la misma convención que los métodos. Por Ejemplo: alumno, padronElectoral.
    d) Constantes: Las variables constantes o finales, las cuales no cambian su valor durante todo el programa se deben escribir en mayúsculas, concatenadas por "_". Ejemplo: ANCHO, VACIO, COLOR_BASE.

e) Todas las palabras deben ser completas, no se puede poner padElec en vez de padronElectoral.

2) El lenguaje utilizado es C++, esto quiere decir que se debe utilizar siempre C++ y no C, por lo tanto una forma de darse cuenta de esto es no incluir nada que tenga .h, por ejemplo #include <iostream>. (Salvo las librerías propias)
3) No usar sentencias 'using namespace' en los .h, solo en los .cpp. Por ejemplo, para referenciar el tipo string en el .h se pone std::string.
4) No usar 'and' y 'or', utilizar los operadores '&&' y '||' respectivamente.
5) Compilar en forma ANSI. Debe estar desarrollado en linux con eclipse y g++. Utilizamos el estándar C++98.
6) Chequear memoria antes de entregar. No tener accesos fuera de rango ni memoria colgada.
7) Si el trabajo práctico requiere archivos para procesar, entregar los archivos de prueba en la entrega del TP. Utilizar siempre rutas relativas y no absolutas.
8) Entregar el informe explicando el TP realizado, manual de usuario y manual del programador.
9) Comentar el código en todos los archivos, tanto los métodos en los .cpp como los Pre y Pos en los .h.
10) Modularizar el código. No entregar 1 o 2 archivos, separar cada clase o struct con sus funcionalidades en un .h y .cpp. Cada metodo o funcion no puede ocupar más de 60 líneas.
11) No inicializar valores dentro del struct o .h.
12) En el TP 2 y TP 3 no se pueden utilizar Arreglos, la estructura dinámica debe ser una vista en Clases.
13) En el TP 2 y TP 3 no se pueden utilizar objetos estáticos, todas las instancias deben ser dinámicas.
14) Las fechas de entrega son para respetarlas, la no entrega en tiempo y forma es causal de desaprobar el TP.
15) Si cualquier estructura de control tiene 1 línea, utilizar {} siempre, por ejemplo:

```
for(int i = 0; i < 10; i++) {
      std::cout << i;
}
```