UBA – Facultad de Ciencias Exactas y Naturales – Departamento de Computación

Algoritmos y Estructuras de Datos I

Primer cuatrimestre de 2019

Versión 1 (8 de abril de 2019)

Trabajo Práctico de Especificación

Fecha de entrega: 26 de abril (hasta las 16:30 hs)
Devolución y coloquio: 6 de mayo
Fecha límite recuperatorio: 10 de junio

Condiciones

El trabajo práctico se realiza de manera grupal, pero su aprobación será individual. Para aprobar el trabajo se necesita:

- Que todos los ejercicios estén resueltos.
- Que las soluciones sean correctas.
- Que la solución sea declarativa.
- Que el lenguaje de especificación esté bien utilizado.
- Que las soluciones sean prolijas: evitar repetir especificaciones innecesariamente y usar adecuadamente las funciones y predicados auxiliares.
- Que demuestren en el coloquio que entienden la solución de cualquiera de los ejercicios y puedan explicarlas con sus palabras.

Pautas de Entrega

- Se debe enviar un e-mail a la dirección tpalgo1@gmail.com. Dicho mail debe cumplir con el siguiente formato:
  - El título debe ser `[ALGO1;TPE]` seguido inmediatamente del nombre del grupo.
  - En el cuerpo del email deberán indicar: Nombre, apellido, libreta universitaria de cada integrante.
  - El informe deberá estar adjuntado en el email con formato .pdf.
- Deberán entregar una versión impresa de la solución en un folio.

1. Introducción

El gobierno de la república de Córdoba está interesado en mejorar su sistema de transporte. Para ello, ubicaron dispositivos GPS en cada uno de los colectivos de línea con el objetivo de mejorar el servicio provisto por las empresas. Estos dispositivos registran la posición en longitud y latitud de los colectivos cada 20 segundos. Luego, de acuerdo con la señal telefónica que dispongan los dispositivos, la información es transmitida a un control central en donde se lleva a cabo el procesamiento de dichos datos.

Por otra parte, el gobierno liberará parte de esta información para que cualquier desarrollador interesado pueda utilizar estos datos y así construir aplicaciones útiles para los usuarios. En este trabajo, nos proponemos resolver problemas relacionados al mundo del transporte con las herramientas que iremos aprendiendo en la materia.

2. Definiciones

Utilizaremos la palabra **recorrido** para referirnos a puntos GPS que indican por donde debería ir un colectivo, incluyendo paradas y otros puntos registrados (en orden) por la compañía cada 20 metros aproximadamente, desde el principio hasta el final del trayecto. Por ejemplo, un recorrido es el de la línea `160-ramal-A-ida`.

Cuando hablemos de **viaje** nos referiremos al trayecto en puntos GPS que indica por dónde fue un colectivo según el dispositivo propio. Estos puntos **no necesariamente están ordenados** pero contienen una marca temporal asociada a cada punto que indica en qué momento fue registrado. Cada viaje está asignado a un recorrido.

Un punto **GPS** estará representado como un par de números reales ($latitud \times longitud$) medidos en grados. Un valor de *latitud* es válido si se encuentra en el rango $[-90.0, 90.0]$; un valor de *longitud* es válido si se encuentra en el rango $[-180.0, 180.0]$.

Para nuestra especificación utilizaremos los siguientes renombres de tipos:

type $Tiempo = \mathbb{R}$ (medido en segundos desde la medianoche (UTC/GMT) del 1/1/1970)

type $Dist = \mathbb{R}$ (medido en metros)

type $GPS = \mathbb{R} \times \mathbb{R}$ (medido en grados de latitud y grados de longitud)

type $Recorrido = seq\langle GPS \rangle$

type $Viaje = seq\langle Tiempo \times GPS \rangle$

type $Nombre = \mathbb{Z} \times \mathbb{Z}$

type $Grilla = seq\langle GPS \times GPS \times Nombre \rangle$

Para los ejercicios pueden utilizar sin definir el auxiliar **aux dist(p1: *GPS*, p2: *GPS*) : *Dist*** que dados dos puntos GPS calcula su distancia en metros (ver por ejemplo `https://es.wikipedia.org/wiki/Formula_del_haversine`). En caso de utilizar esta auxiliar con valores de GPS fuera de rango, la función se indefine.

3. Ejercicios: especificar los siguientes problemas

Ejercicio 1. proc enTerritorio(in v: *Viaje*, in r: *Dist*, out res : *Bool*) Que chequea que todos los puntos registrados en el viaje se encuentren dentro de un círculo de radio r kilómetros.

Ejercicio 2. proc excesoDeVelocidad(in v: *Viaje*, out res : *Bool*) Que dado un viaje devuelva verdadero si el colectivo superó los 80 km/h en algún momento del viaje.

Ejercicio 3. proc tiempoTotal(in v: *Viaje*, out t : *Tiempo*) Que dado un viaje, determine el tiempo total que tardó el colectivo. Este valor debe ser calculado como el tiempo transcurrido desde el primer punto registrado y hasta el último.

Ejercicio 4. proc distanciaTotal(in v: *Viaje*, out d : *Dist*) Que dado un viaje, determine la distancia recorrida aproximada utilizando toda la información registrada en el viaje, es decir, utilizando la información registrada de todos los tramos.

Ejercicio 5. proc flota(in v: $seq\langle$*Viaje*$\rangle$, in t_0 : *Tiempo*, in t_f : *Tiempo*, out res : $\mathbb{Z}$) Que dada una lista de viajes, calcule la cantidad de colectivos que se encontraban realizando un viaje en la franja de tiempo $[t_0, t_f]$.

Ejercicio 6. proc recorridoCubierto(in v: *Viaje*, in r: *Recorrido*, in u : *Dist*, out res : $seq\langle$*GPS*$\rangle$)
Que dado un viaje v, un recorrido r y un umbral u, devuelva todos los puntos del recorrido que no fueron cubiertos por ningún punto del viaje. Se considera que un punto p del recorrido está cubierto si al menos un punto del viaje está a menos de u mts. del punto p.

Ejercicio 7. proc construirGrilla(in esq1: *GPS*, in esq2: *GPS*, in n: $\mathbb{Z}$, in m: $\mathbb{Z}$, out g: *Grilla*) Que dados dos punto GPS, construye una grilla de $n \times m$. Estas grillas están conformadas por celdas cuadradas contiguas del mismo tamaño entre sí. Cada celda estará caracterizada por sus puntos superior izquierdo e inferior derecho (coordenadas GPS) y un nombre (un par ordenado que representa su posición en la grilla desde $(1, 1)$ en el punto que se encuentre en la celda que comienza en la posición $esq1$ y hasta (n, m) en la posición en donde se encuentra la celda con esquina $esq2$) .[1] Por ejemplo, el panel izquierdo de la Figura 1, muestra un ejemplo de grilla de (3×5).

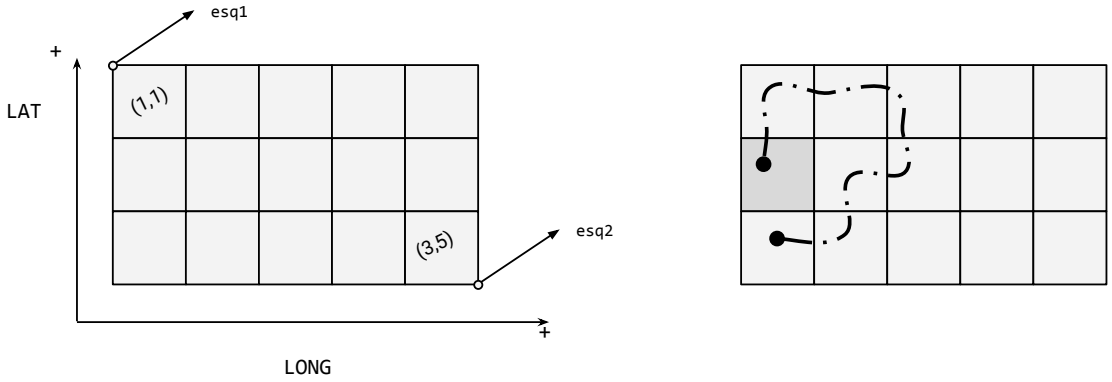


Figura 1: En el lado izquierdo de la figura, un ejemplo de grilla de 3×5. En el panel derecho, un ejemplo de trayecto en donde cada punto simboliza un registro GPS del viaje con comienzo en la celda sombreada.

Ejercicio 8. proc aPalabra(in trayecto: $seq\langle$*GPS*$\rangle$, in g: *Grilla*, out res: $seq\langle$*Nombre*$\rangle$) que dado un trayecto, devuelve la palabra que genera sobre una grilla. Una *palabra* es la secuencia ordenada de regiones visitadas por el colectivo [2]. Por ejemplo, en el panel derecho de la Figura 1, puede verse un trayecto que genera la siguiente secuencia: $< (2,1), (1,1), (1,1), (1,2), (1,3), (2,3), (2,2), (2,2), (3,2), (3,1) >$

Ejercicio 9. proc cantidadDeSaltos(in g: *Grilla*, in v: *Viaje*, out res : $\mathbb{Z}$) Que dado un viaje y una grilla, determine cuántos saltos hay en el viaje. Diremos que hay un *salto* si dos puntos del viaje consecutivos temporalmente se encuentran a dos o más celdas de distancia.

[1] Para este punto decidimos ignorar la curvatura de la tierra.

[2] Ejemplo real de utilización de grillas: "Real-Time Detection of Anomalous Taxi Trajectories from GPS Traces" `https://www.semanticscholar.org/paper/Real-Time-Detection-of-Anomalous-Taxi-Trajectories-Chen-Zhang/77b7a8fea9ac604af487d262ec21b15e96062713`

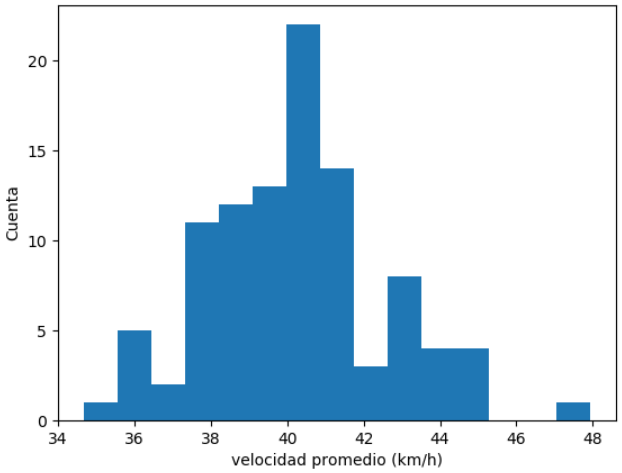
Ejercicio 10. proc completarHuecos(inout v: *Viaje*, in faltantes: $seq\langle\mathbb{Z}\rangle$)

A veces los viajes tienen valores faltantes (o *huecos*) producto de un error en su carga y eso puede provocar saltos o resultados incorrectos en las mediciones. Para este ejercicio, se contará con un viaje incompleto en donde, en las posiciones indicadas por el parámetro $faltantes$, encontrarán el par (t_i, (0,0)) que contendrá el *i-ésimo* tiempo registrado en el viaje y el valor (0,0) indicando un hueco en ese momento.

Se pide completar el viaje de la siguiente manera: donde haya un hueco, se buscan los dos puntos más cercanos temporalmente registrados correctamente y se completa con el punto GPS tal que se encuentre a distancia proporcional al tiempo entre cada uno de los puntos más cercanos.

Ejercicio 11. proc histograma(in xs: $seq\langle$ *Viaje*$\rangle$, in bins: $\mathbb{Z}$, out cuentas : $seq\langle\mathbb{Z}\rangle$, out limites : $seq\langle\mathbb{R}\rangle$)

Que dada una lista de viajes, calcule el histograma de velocidades máximas registradas entre todos los viajes.



Dado un conjunto de números (en este caso velocidades máximas), un *histograma* puede calcularse dividiendo el rango en que se mueven dichos números en intervalos de igual ancho (también llamados *bins*) en los que para cada uno se contabiliza la cantidad de valores incluidos en ellos.

- el parámetro **cuentas** deberá corresponderse con la cantidad de elementos en cada bin.
- el parámetro **limites** deberá contener los límites de cada uno de los bins en orden como se muestra en los siguientes ejemplos:
  - si las velocidades máximas de 4 viajes son $\langle 0, 1, 2, 10\rangle$ y $bins = 5$, el resultado debería ser $cuentas = \langle 2, 1, 0, 0, 1\rangle$, $limites = \langle 0, 2, 4, 6, 8, 10\rangle$
  - si las velocidades máximas de 10 viajes son $\langle 33, 24, 24, 1, 62, 88, 94, 79, 25, 24\rangle$ y $bins = 4$, el resultado debería ser $cuentas = \langle 4, 2, 1, 3\rangle$, $limites = \langle 1.00, 24.25, 47.50, 70.75, 94.00\rangle$.

Aclaración: como puede verse en el ejemplo, los intervalos se consideran cerrado-abierto, salvo el caso del último intervalo que se concederá cerrado-cerrado.

Ejercicio 12. proc limpiar(inout r: $seq\langle$ *Viaje*$\rangle$, out borrados : $seq\langle\mathbb{Z}\rangle$)

Algunos viajes contienen velocidades máximas demasiado altas que deben ser analizados por separado. Llamaremos a estos puntos: *extremos*. Definiremos un viaje como *extremo* si al construir un histograma con 10 bins resulta pertenecer al último bin.

En este ejercicio, se pide remplazar los viajes *extremos* por listas vacías y en la variable *borrados* se indique cuáles fueron los índices en que han sido eliminados.