UTN – FACULTAD REGIONAL BUENOS AIRES
Materia: Algoritmos y Estructura de Datos – 2021
CURSO K1025

TRABAJO PRÁCTICO NÚMERO 1

“Sentencias de asignación y Sentencias selectivas”

de Miguel, Thiago Alexander 2035558 tdemiguel@frba.utn.edu.ar Github: tdemiguelUTN

Link del repositorio: https://github.com/tdemiguelUTN/TRABAJOS.git

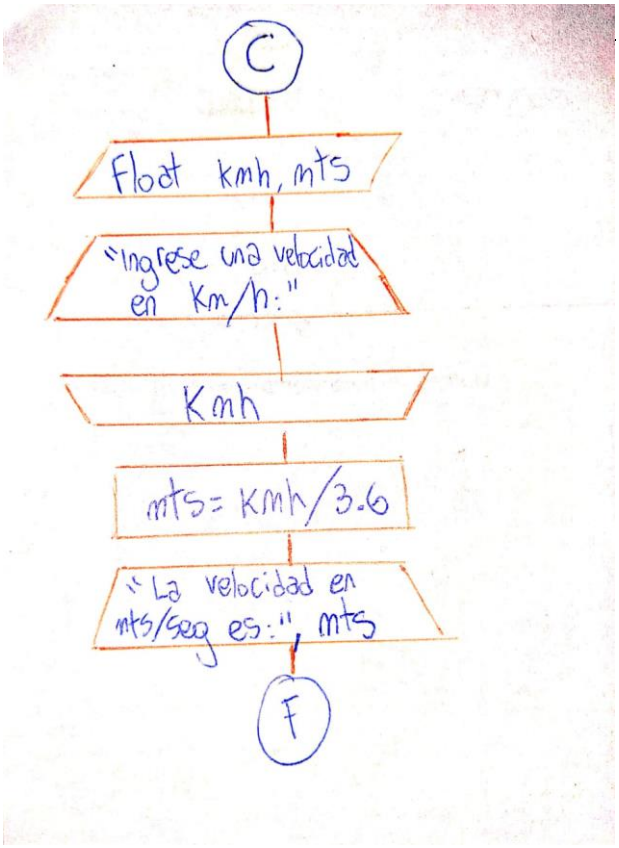
FECHA DE PRESENTACIÓN: 05/05/2021 FIRMA PROFESOR _____

FECHA DE DEVOLUCIÓN: CALIFICACIÓN _____

Análisis y diagramas de los ejercicios realizados:

3)a) En este ejercicio lo primero que hice es leerlo por completo, para poder ver cuál es el objetivo, a partir de ahí poder tomar el camino más efectivo para la situación.
Primero elegí los tipos de datos que voy a usar, en este caso son float ya que se usa para números con decimales. Luego queda definir las variables con las que vamos a trabajar. Coloque nombres adecuados para reconocer fácilmente lo datos, y así poder trabajar de forma mas cómoda.
El objetivo de este ejercicio es poder leer una velocidad en km/h y transformarla a mts/seg.
Se le tiene que pedir al usuario los valores para que el programa los pueda almacenar en la variable (en este caso es "kmh").
A partir de esto, queda dividir el valor que quedo en la variable kmh por 3,6 (con esto podemos pasarlo a mts/seg). Después almacenamos el resultado en la variable "mts".
Finalmente mostramos en pantalla cual es el valor de la variable mts.

Diagrama.



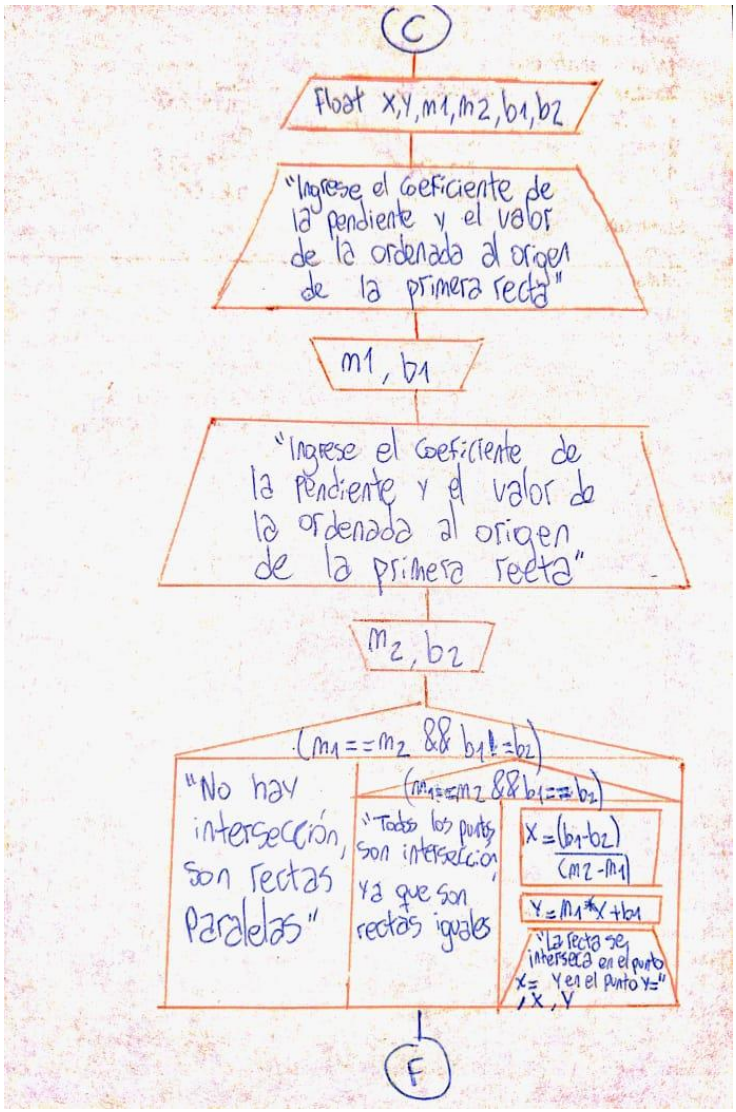
b) En este caso, se realiza lo mismo que con el anterior, es fundamental leer el enunciado de forma completa para poder enfocarnos en lo que tenemos que realizar.
Primero elegi los tipos de datos que voy a usar, nuevamente vuelve a ser float (aunque también podría servir el double). Luego, se definen las variables con las que vamos a trabajar. Los nombres están relacionados con la forma de la ecuación de la recta, como también está escrito en el enunciado.
Primero pido al usuario que ingrese los datos de su primera recta (coeficiente de la pendiente y la ordenada al origen). Se almacena en las variables correspondientes.

Después volvemos a pedir al usuario que ingrese los datos, pero ahora de su segunda recta. Se vuelven a almacenar en otras variables para no sobrescribir los valores anteriores y así poder trabajar de forma mas ordenada (cada recta tiene sus propios datos).
Con estos datos ya almacenados, ahora surgen varias posibilidades:

1. *Que las rectas sean paralelas*: significa que no hay intersección. Esto sucede porque las pendientes son iguales, pero las ordenadas al origen son distintas.
2. *Rectas iguales*: Si las dos rectas son iguales, significa que cualquier punto que tomemos entre esas dos, van a intersecarlas.
3. *Puntos específicos donde se intersecan las dos rectas*: Cuando los valores son diferentes entre sí, debemos calcular la intersección entre dos rectas. Para ello tenemos que igualarlas entre sí, calcular el valor de "x" y reemplazarlo en cualquiera recta, así poder sacar el valor de "y".

Entonces debemos definir las condiciones con if para los distintos casos. Si no se cumple el punto 1 se muestra un mensaje correspondiente para que el usuario tenga una noción del caso específico, lo mismo sucede con el punto 2. Entonces si no cumple las condiciones del 1 y del 2, se ejecuta el caso 3. En este se realiza la formula de despeje para poder averiguar el valor de x, luego conseguimos el valor de y.
Finalmente se muestra al usuario los valores del punto específico en el cual se intersecan las rectas.

<u>Diagrama.</u>



4) JavaScript tiene tres tipos de declaraciones de variables:

`Var`: Declara una variable, opcionalmente la inicia a un valor.
`Let`: Declara una variable local con ámbito de bloque, opcionalmente la inicia a un valor.
`Const`: Declara un nombre de constante de solo lectura y ámbito de bloque.

Se puede declarar una variable de dos formas:

- Con la palabra clave `var`. Por ejemplo, `var x = 42`. Esta sintaxis se puede utilizar para declarar variables locales y globales.
- Con la palabra clave `const` o `let`. Esta sintaxis se puede utilizar para declarar una variable local con ámbito de bloque.

También puedes simplemente asignar un valor a una variable. Por ejemplo, `x = 42`. Este formulario crea una variable `global no declarada`. También genera una advertencia estricta de JavaScript. Las variables globales no declaradas a menudo pueden provocar un comportamiento inesperado. Por lo tanto, se desaconseja utilizar variables globales no declaradas.

Con respecto a la sentencia if en JavaScript (Ejecuta una sentencia si una condición especificada es evaluada como verdadera. Si la condición es evaluada como falsa, otra sentencia puede ser ejecutada) se mantiene muy parecida a c++.

La sintaxis es esta:

`if (`*`condición`*`) `*`sentencia1`* `[else `*`sentencia2`*`]`

`condición`: Una expresión que puede ser evaluada como verdadera o falsa.

`sentencia1`: Sentencia que se ejecutará si `condición` es evaluada como verdadera. Puede ser cualquier sentencia, incluyendo otras sentenccias `if` anidadas. Para ejecutar múltiples sentencias, use una sentencia `block` ({ ... }) para agruparlas.

`sentencia2`
Sentencia que se ejecutará si `condición` se evalúa como falsa, y exista una cláusula `else`. Puede ser cualquier sentencia, incluyendo sentencias block y otras sentencias `if` anidadas.