Universidad Tecnológica Nacional
Facultad Regional Buenos Aires
Algoritmos y Estructuras de Datos
Curso K1024
Ing. Pablo D. Mendez
Trabajo práctico individual NRO 1

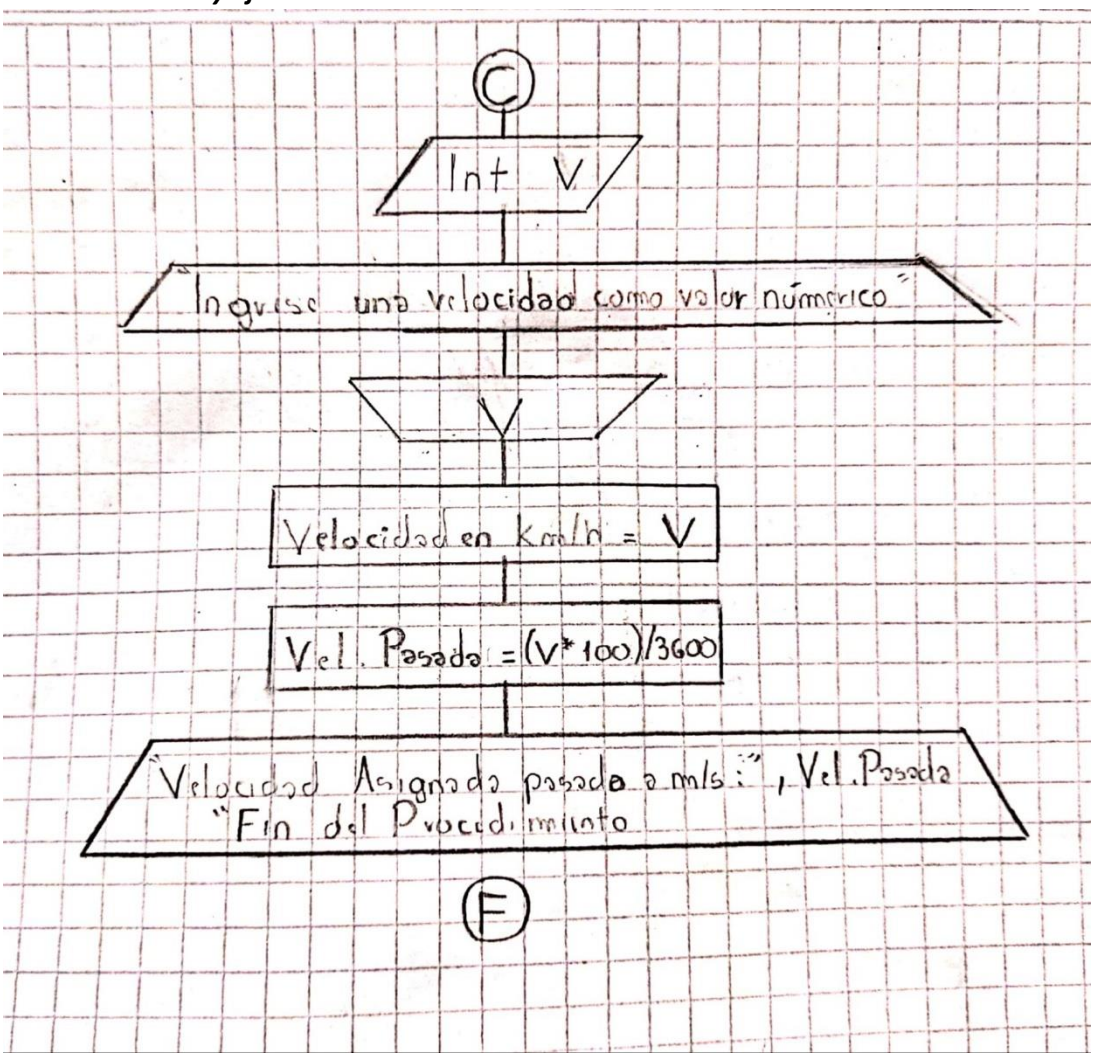
Sentencias de asignación y sentencias Selectivas

Fecha de entrega: 15/5

Alumno: Maximiliano Alexander Vargas
Legajo: 2041364
Correo Electrónico: maxivargas@frba.utn.edu.ar
Usuario de GitHub: Maximilian04lexanderV4rgas
Link al Repositorio: https://github.com/Maximilian04lexanderV4rgas/RepositorioPrivado-1.git
Curso: K1024
Año: 2021

3. ***Análisis del Ejercicio A***: El ingreso de un valor numérico "V" que represente una velocidad medida primeramente en Km/h conforma el primer parte del procedimiento, posterior al ingreso del primer valor se lo multiplica por 1 y por 1000. El resultado de la cuenta dada puede variar dependiendo del valor de "V" (como por ejemplo el valor en m/s de 7 Km/h y 9Km/h difieren entre sí, otro elemento que puede variar en el resultado final es si dicho valor asignado es negativo o no ya que representaría en la cuenta final un cambio que representaría que dicha velocidad es negativa o positiva), posterior a este procedimiento dicho resultado se lo divide por 3600 y se informa el resultado final que representaría el pasaje del valor "V" en Km/h a m/s teniendo siempre en cuenta el primer valor que se asignó al principio del procedimiento con todo y sus propios valores (como por ejemplo si dicho valor numérico estaba representado con valor positivo o negativo o también así si "V" originalmente pertenece al conjunto de números enteros, fraccionarios o irracionales, en cuyo caso esto último podría afectar el resultado final).

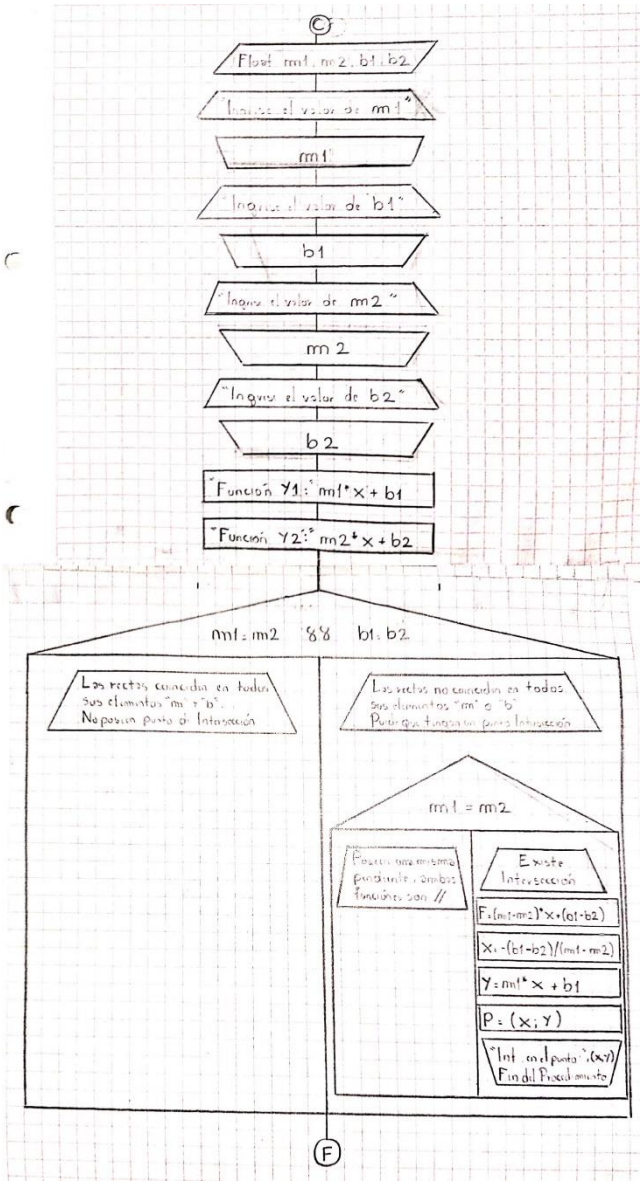
Cuadro de Lindsay Ejercicio A:



Análisis del Ejercicio B: El ingreso de las variables m1,m2,b1,b2 conforman los elementos de dos funciones que pueden intersectar entre si dependiendo de la variables antes mencionadas, de este hecho cabe aclarar que dichas variables son independientes una de la otra (todos los elementos conformantes pueden ser positivos, negativos, enteros ,fraccionarios como también irracionales conformando así la primer variable del sistema), así mismo la intersección entre ambas funciones lineales también puede variar ya

los elementos de las mismas pueden cumplir ciertos requisitos para no intersectarse como por ejemplo que m1=m2 y b1=b2 (representaría que ambas funciones son iguales y su intersección seria infinita sin tener un punto especifico de intersección ya fijado) o como también así que la pendiente "m" de ambas funciones sea la misma (las funciones lineales que poseen las misma pendientes son paralelas entre sí, no poseen un punto de intersección) dichos casos si llegasen a efectuarse darían una conclusión al procedimiento. Por otro lado, si ambos casos no llegasen a ocurrir la intersección entre ambas funciones sería posible independientemente de las características de sus componentes "m" y "b" ya que ambas funciones de un modo u otro al ser lineales (toda función lineal posee un dominio como imagen que se expande por todos los $\mathbb{R}$ del eje de abscisas y ordenadas) van a concordar en un punto que represente la intersección entre ambas funciones lineales.

Cuadro de Lindsay Ejercicio B:



4. En "JavaScript" primeramente si se requiere declarar una variable se empieza Ingresando la palabra "var" (representación de variable en el sistema "JavaScript") seguida del nombre de la variable (texto o enunciado independiente), uno de las restricciones que posee dicho procedimiento es el hecho no admitir en la parte de "nombre de la variable" valores

numéricos sin ninguna letra, ya que automáticamente el proceso de inicio saltaría con error si se ingresaran solamente números como por ejemplo (“var 1110” sería un elemento erróneo en la inicialización del software utilizado (“JavaScript”) mientras que si es escrito de esta la forma “var mil_ciento_diez” sería más asertivo con el proceso de inicialización funcionando correctamente .

Posterior a esto si la variable debe de almacenar información, se ingresa al final de lo hecho anteriormente el signo = esta misma sirve como apertura para la información que se desea almacenar. Sabido esto entre “” se ingresa lo requerido por el operador y cerrando la misma con; creando así la primera variable del “Sist.JavaScript”. Así mismo el ingreso de las variables va forma descendente dependiendo del momento en el cual las misma son inscriptas en el sistema. Cómo, por ejemplo:

“JavaScript” (Ingreso de una Variables)

```
1    var  variable_1 ;
2    variable_1 =  “Variable número uno” ;
3    variable_1 =  “Variable número dos”  ;
4    variable_1 =  “Variable número dos”  ;
```

A diferencia de lo hecho anteriormente en “CodeBlocks” la estructura entre ambos difiere por diferentes aspectos como por ejemplo el intérprete del copilador (elemento que adjunta y analiza lo asignado mediante un conjunto de mecanismos de orden) el cual en JavaScript tiende a ser un tanto mas directo y simple a diferencia de lo usado con frecuencia (CodeBlocks) de este hecho la comprensión como dirección de ambos elementos esencialmente difieren en el la utilización de sus herramientas ya asignadas como por ejemplo a la hora de ingresar una ingreso de la variable (Int), y el flujo de entrada y salida del elemento asignado (“cin” y “cout”) dando al conjunto o elemento singular asignado el carácter de variable, entre lo mas utilizado para asignar una variable al proceso realizado, dichos elementos no poseen lugar en la estructura analítica del procedimiento que toma “JavaScript” . Como, por ejemplo, esta sería la estructura que utilizaría un sistema que no fuera “JavaScript” para ingresar una variable a la operación en general:

“CodeBlocks” (Ingreso de una Variable)

```
1 {
2    int  V ;
3    cout << "Ingrese una velocidad como valor numérico (Lectura en Km/h):";
4    cin >> (V);
5  }
```

Dando una conclusión a la diferencia entre ambos procedimientos cabe aclarar como el software de ejecución del “JavaScript” es mas directo con respecto al procedimiento e ingresos de diferentes variables ya que su conformación no depende de tanto factores (mostrado en el ejemplo de su grafica en procedimiento) como el “Int” o el “Cout y Cin”. En si ambos procedimientos muestran la representación de una variable pero difieren en el proceso de ejecución siendo uno mas directo y simple en primera forma (JavaScript) y el otro (CodeBlocks) siendo mas complejo y sistemático a la hora de complementar y asignar las variables pero al fin de cuentas ambos sistemas analizan e interpretan de diferentes como sus sistemas se los permite el ingreso de una o mas variables que el operdador asigne.

Por otro lado el elemento “If” en el sistema JavaScript es una estructura de control utilizada para tomar decisiones a base de diferentes condiciones que deben o no de cumplirse dependiendo de la misma variable. De este cumpliendo la función de condicional, esta misma sirve para realizar unas u otras operaciones en función de una sola expresión declarada. Funcionando principalmente por la dogma de “Si pasa esto, entonces se ejecuta esto” si un determinado elemento se cumple entonces dicho resultado ya preinscripto es correcto, por otra parte, si esto no se llegase a cumplir entonces el postulado es erróneo y se llega a otra conclusión. Analizándolo desde el aspecto operativo el elemento “If” en JavaScript, se ingresa primero una variables “var“ con su respectivo nombre posteriormente se ingresa “If” y () siendo estos paréntesis los contenedores de los condicionales de la operación y después añadir al final de los inscripto las llaves { } que contendrían el primer resultado si se cumpliera la condición siguiente a esto se repite el mismo procedimiento pero esta vez poniendo en el condicional () lo opuesto del primero y las llaves { } conteniendo un segundo resultado que exprese el cumplimiento de la segunda condición. Un ejemplo seria:

“JavaScript” (Ingreso de If)

```
1    var  variable_1 ;
2    if (variable_1 >= 2 ) {
3   “La variable es mayor a 2”  }
4   if (variable_1 < 2 ) {
5    “La variable es menor a dos” }
```

Como se llega a observar la asignación de variables en JavaScrpit tiende a ser mas directa con respecto al ingreso de las diferentes herramientas que acompañan a la operación, ya que desde un punto de vista mas respectivo el copilador como el sistema pueden llegar a identificar como interpretar una variable sin la necesidad de tantos elementos que justifiquen que esta efectivamente se trata de una variable (en aspectos mas aclarados el sistema analítico del JavaScript es mas directo y simple al respecto de lectura y análisis de variables ya que no necesita de tantos elementos para conforma una).Entre la capacidad de interpretación que el sistema posee tambien hay un elemento que resalta, este mismo se trataría del formato en el cual este se ejecuta ya que presenta entre sus características el hecho de manejarse en paneles del 1 al numero limite (marcando el orden de los pasos en orden de como ingresan a la operación) dicha característica la compate tambien con los sistemas antes mencionados, para demostrar esto tenemos este esquema del sistema “CodeBlocks” presentando lo mismo que hiso JavaScript con anterioridad.

“CodeBlocks” (Ingreso de If)

```
1     float m1 ;
2     cout << "Ingrese el valor de m1;" << endl;
3      cin >> m1 ;
4        if (m1 == 2)
5  {
6     cout << "m1 es igual 2." << endl ;
7     return 0;
8  }
9      else
10  {
11     cout << "m1 es diferente de 2" << endl ;
```

Viendo como ambas presentaciones de “Ingreso de If” exponen, se puede observar como comparten el uso de paneles para asignar cada paso utilizado y dar orden al ingreso de diferentes mecanismos (en este caso If) en el procedimiento como también el uso del elemento “If” en el inicio del proceso , pero también cabe aclarar como difieren a la hora de presentarlas ya que el sistema CodeBlocks presenta “If” de una forma mas especifica y puntual sobre lo que debe cumplirse o no, mientras que el sistema de “JavaScript” tiende a ser mas directo con respecto al análisis y comprensión de lo pedido (el uso mas escaso de herramientas analíticas),llegando a la conclusión de que ambos procedimientos comparten tanto similitudes como diferencias pero remarcan siempre el hecho de demostrar como identificar en el procedimiento el ingreso de variables como también así de condicionales que remarquen un resultado ya asignado.