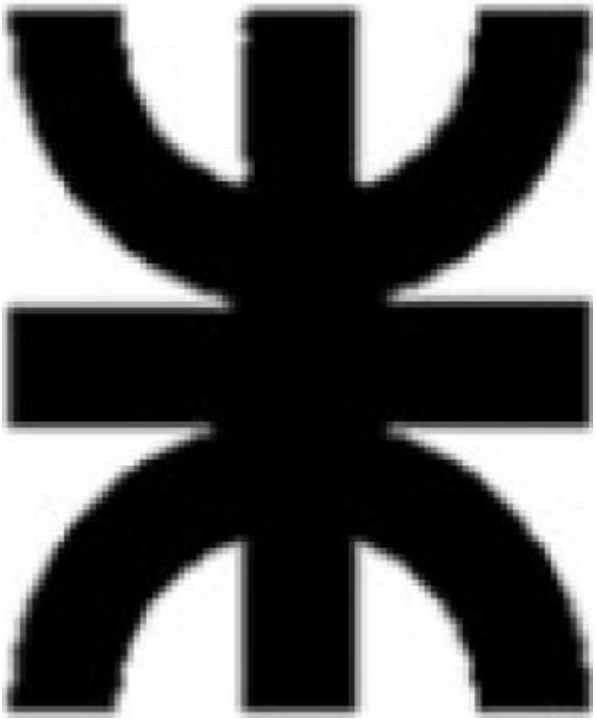


UNIVERSIDAD TECNOLÓGICA NACIONAL
Facultad Regional Reconquista

Programación en Computación

Ciclo Lectivo: 2020

Trabajo Práctico N.º 3

Tema Unidad 3: **Tipos de Datos**

GRUPO Nº: 12
INTEGRANTES: Fulano, Mengano, Zutano

Guía Unidad 3:
ENTIDADES PRIMITIVAS PARA EL DESARROLLO DE ALGORITMOS

SITUACIONES PROBLEMÁTICAS:

1. Indicar en que orden se resuelven y el resultado que obtiene:

a) 9 + 7 * 8 – 36 / 5;
b) 15 / 2 * (7 + (68 – 15 * 33 + (45 ^ 2 / 16) / 3) / 15) + 19
c) X = 6, B = 7,8 (X * 5 + B ^ 3 / 4) **<=** (X ^ 3 div B)
d) ((1580 mod 6 * 2 ^ 7) **>** (7 + 8 * 3 ^ 4)) **>** ((15 * 2) **=** (60 * 2 / 4))

2. Indicar en que orden se resuelven y el resultado que obtiene:

a) NO (15 >= 7 ^ 2) **O** (43 – 8 * 2 div 4 <> 3 * 2 div 2)
b) (15 >= 7 * 3 ^ 2 **Y** 8 > 3 **Y** 15 > 6) **O NO** (7 * 3 < 5 + 12 * 2 div 3 ^ 2)
c) NO ((7 * 3 div 2 * 4) > (15 / 2 * 6 >= 15 * 2 / 17 = 15))

3. ¿Cuál de los siguientes son Datos válidos para procesar en una PC?

a) 3,14159;	**b)** 0,0014;	**c)** 12345,0;	**d)** 15,0E-04;
e) 2,234E2;	**f)** 12E+6;	**g)** 1,1E-3;	**h)** -15E-0,4;
i) 12,5E,3;	**j)** ,123E4;	**k)** 5A4,14;	**l)** A1,E04.

4. ¿Cuál de los siguientes Identificadores son válidos?

a) Renta;	b) Alquiler;	c) Constante;
d) Tom’s;	e) Dos Pulgadas;	f) C3PO;
g) Bienvenido#5;	h) Elemento,	i) 4A2D2;
j) 13Nombre;	k) Nombre_Apellido;	l) NombreApellido;

5. Dados:

eI,	**eJ**,	**eAcum**	de tipo	***Entero;***
rRea,	**rSum**		de tipo	***Real;***
cChar			de tipo	***Carácter;***
bAND			de tipo	***Booleano;***

Realizar las siguientes asignaciones:

a) eI ← 0;	**b)** eI ← eI + 1;
c) eAcum ← 0;	**d)** eJ ← 5 ^ 2 div 3;
e) cChar ← ’a’;	**f)** eAcum ← eJ div eI;
g) rRea ← eAcum / 3;	**h)** bAND ← (8>5) Y (15 < 2 ^ 3);
i) rSum ← eAcum * 5 / eJ ^ 2;	**j)** eI ← eI * 3;
k) rRea ← rRea / 5;	**l)** eI ← rRea;
o) cChar← eJ;	**p)** bAND ← bAND O (eI = eJ);

Construir una tabla con los valores que van tomando las variables en memoria.

6. Diseñar un algoritmo que lea cuatro variables y calcule y muestre: su producto, su suma y su media aritmética.

7. Hacer Tabla que muestre el contenido de las variables A, B y C después de ejecutar las siguientes instrucciones:

a) A ← 3; **b)** B ← 20; **c)** C ← A + B; **d)** B ← A + B; **e)** A ← B;

8. Hay 3 variables A, B y C. Escribir instrucciones necesarias para intercambiar sus valores:

B ← A; C ← B; A ← C;

9. Escribir las siguientes expresiones en un algoritmo.

a) $\frac{M}{N}+4$; **b)** $M+\frac{N}{P-Q}$;

10. Escribir las siguientes expresiones en un algoritmo.

$$\frac{\sin x \cos y}{\tan x}$$

11. Escribir las siguientes expresiones en un algoritmo.

$$\frac{M+N}{P-Q}$$

12. Escribir la siguiente expresión en forma algorítmica.

$$\frac{-b+\sqrt{b^2-4ac}}{2a}$$

ACTIVIDADES:
Resolver las situaciones problemáticas anteriores.

Conocimientos necesarios:

U 2) Algoritmos. Fases en la Creación de Programas. [Programa = Algoritmos + Estructuras de DATOS].

U 3) Tipos de Datos. Expresiones. Operadores y Operandos. Identificadores de Constantes y Variables. Instrucciones Básicas: (1- Asignación, 2- Entrada, 3- Salida, 4- Expresiones Aritméticas, 5- Expresiones Lógicas y 6- Palabras Reservadas).

RECOMENDACIÓN:
Para proceder a la realización de los TPs de esta Guía, es recomendable releer la siguiente bibliografía para un repaso de los conceptos teóricos involucrados en resolución de estos problemas:

Diseño_de_Algoritmos.pdf,	En especial Capítulos 1 y 2.
Curso de Algoritmia.pdf,	En especial Capítulos 2 y 3.

Operatoria:
Varía con cada situación problemática presentada, pero se resuelve en base a los conocimientos necesarios considerados.

Resumen de estos:
Dato: es la expresión general que describe los objetos con los cuales opera el Algoritmo. El ***tipo de dato*** determina la forma de almacenamiento en Memoria y las operaciones válidas.
Llamamos ***identificador*** al **nombre** de las casillas de Memoria que contienen el dato.

El ***identificador*** debe cumplir ciertas reglas:
1- El primer carácter debe ser una letra.

2- Los siguientes pueden ser letras o números y el símbolo especial _

En casi cualquier Lenguaje *(Ojo que Raptor es un caso muy especial)* existen los siguientes ***tipos de datos predefinidos:***

Entero: Subconjunto Finito de Enteros, cuyo rango depende del Lenguaje.
Real: Subconjunto de Reales limitados en tamaño y precisión según el Lenguaje.
Lógico: Conjunto formado por los valores Verdadero y Falso
Carácter: Conjunto finito y ordenado de los caracteres que la PC reconoce.

Cadena o String: Datos de este tipo están formados por serie finita de caracteres. Algunos Lenguajes permiten al programador definir ***nuevos tipos de datos***, que en general están formados ***agrupando*** **valores de** ***datos predefinidos***, pero es importante considerar los posibles ***valores*** **y las** ***operaciones*** que pueden efectuarse con el ***nuevo tipo de dato.***
Para indicar un ***tipo de dato*****,** se lo declarara en el ***identificador***, con el nombre del ***tipo***.

Los Datos pueden venir indicados como:
1. Constantes,
2. Variables,
3. Expresiones,
4. Funciones.

Observaciones:
Falso es menor que Verdadero, y No se puede comparar número con valor lógico.

Tabla de Verdad de los Operadores Lógicos

A	B	~A	~B	A o B	A y B
VERDADERO	VERDADERO	FALSO	FALSO	VERDADERO	VERDADERO
VERDADERO	FALSO	FALSO	VERDADERO	VERDADERO	FALSO
FALSO	VERDADERO	VERDADERO	FALSO	VERDADERO	FALSO
FALSO	FALSO	VERDADERO	VERDADERO	FALSO	FALSO

Tabla de Jerarquía o Prioridad de los Operadores

Operadores	Jerarquía o Prioridad
( )	(mayor)
^	|
*, /, divE [\ ], Mod	|
+, --	|
=, <>, <, >, <=, >=	|
NO [NOT]	|
Y [AND]	|
O [OR]	(menor)

A continuación, realizaremos un problema numérico, cuya consigna es:

12 **Escribir la siguiente expresión en forma algorítmica.**

$$\frac{-b+\sqrt{b^2-4\,ac}}{2\,a}$$

- Siguiendo las directivas dadas en el Diagrama de Flujo de la página 2 de la Guía 2, en el punto **Fase de Resolución de un Programa**, procederemos a realizar:
- El ***Análisis del Problema***, lo que implica Examinar cuidadosamente el problema para tener una idea clara de lo que se solicita y determinar los DATOS (de ***Entrada, Salida y Auxiliares***).
- Aquí es evidente que estamos en presencia de la Resolvente de la Ecuación Cuadrática.
- Los datos de ***Entrada*** son tres valores numéricos que representan los coeficientes de: **f(x) = a*X^2 + b*X +c.**
  A estos coeficientes los identificaremos con: **coef_A, coef_B** y **coef_C.** y los definiremos como datos de tipo Real. Hay lenguajes fuertemente tipeados (C#) que lo exigen, Raptor NO.
- En este caso, es necesario como dato ***Auxiliar***, el valor que está bajo la raíz, (conocido como Discriminante), ya que como no manejamos el Cálculo Complejo no podríamos realizar ese cálculo si su valor fuera negativo. A este coeficiente lo identificamos con: ***rDiscr.*** Además, definiremos otra variable ***Auxiliar*** de Texto (string o cadena) para guardar mensajes de salida, y la llamaremos ***sMensaje***.

- El ***Diseño del Algoritmo, [Secuencia ordenada de pasos*** - Sin Ambigüedades – ***que conducen a la solución de un Problema],*** que debe ser:
  i – Preciso ii - Definido iii – Finito.
- En este caso, el Algoritmo o ***Proceso*** viene dado por la fórmula de la Resolvente de la Ecuación, y con esta fórmula ya podemos continuar para obtener los valores **X_1 y X_2** que serán las soluciones solicitadas, si existen.

- ***La Salida, que*** simplemente debe ser un Mensaje que imprime los valores de **X_1 y X_2**

Ahora codificamos todo el proceso en "*Pseudo-Código*":

1) Variables a Utilizar:

DE ENTRADA - Número Real: **coef_A, coef_B, coef_C**

DE SALIDA - Número Real: ***X1, X2***

Cadena de Texto: ***sMensaje***

AUXILIAR - Número Real: ***rDiscr***

Inicio

Leer: **coef_A, coef_B, coef_C.**

Asignar: ***rDiscr*** = (**coef_B^2 – (4*coef_A* coef_C))**

Asignar: ***X1= (-coef_B+Raiz2(rDiscr)) / (2 * coef_A)***

Asignar: ***X2= (-coef_B-Raiz2(rDiscr))/ (2 * coef_A)***

Asignar: ***sMensaje = "Raiz X_1 = " + X1 + ", Raiz X_2 = " + X2***

Imprimir: ***sMensaje***

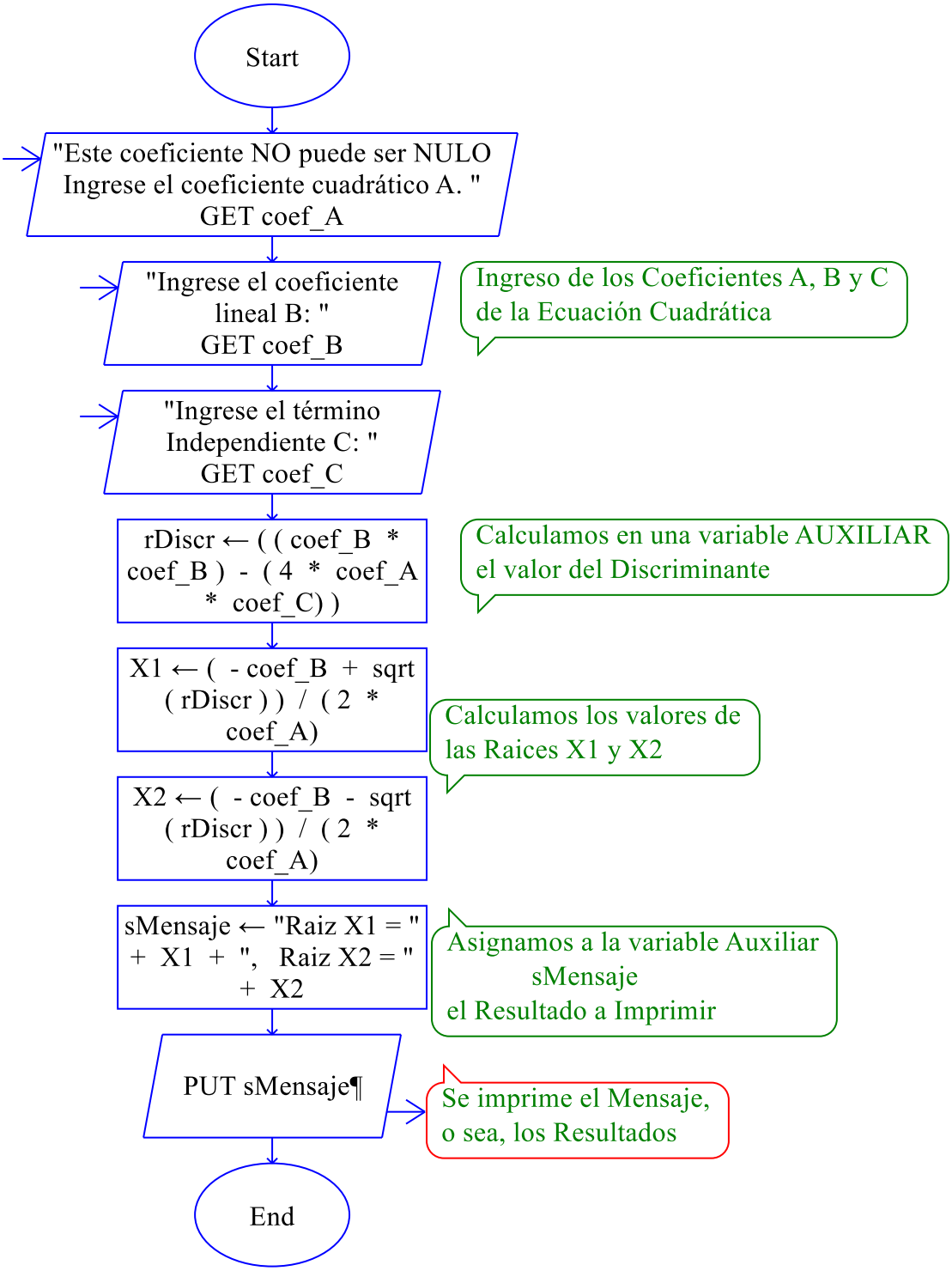
Fin

C)- A continuación, se debería verificar que las respuestas dadas son correctas:

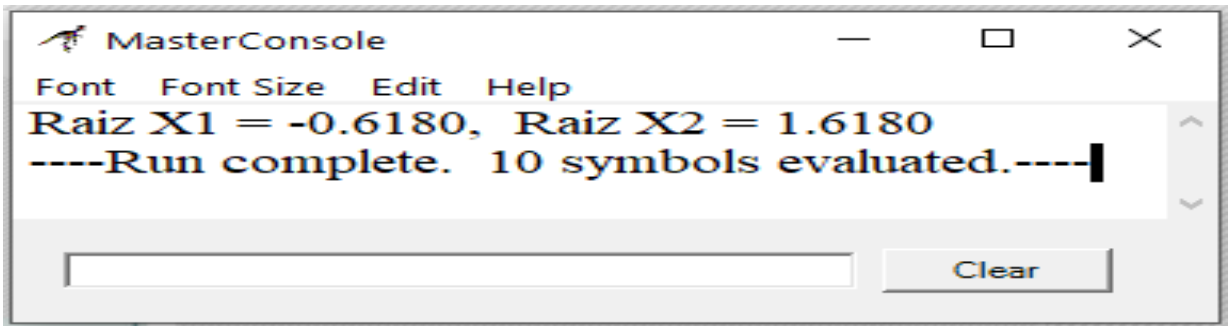
- Si a este programa lo probamos con valores del ***coef_A = 0,*** nos dará una **división por 0**. Operación no posible en Matemática Discreta.
- Si los coeficientes ingresados nos dan un valor del ***rDisc < 0,*** tendremos la **raíz cuadrada de un número negativo**, que no tiene solución en el campo de los Reales, por lo tanto, deberíamos evitar que esta situación ocurra.

La solución a los dos problemas anteriores exige el uso de Condicionales y/o Ciclos, que veremos más adelante como resolverlo.

Completado el programa en **"*Pseudo-Código*",** procederemos a realizar el programa en ***Diagrama de Flujo – Ejecutable***, o sea en RAPTOR.

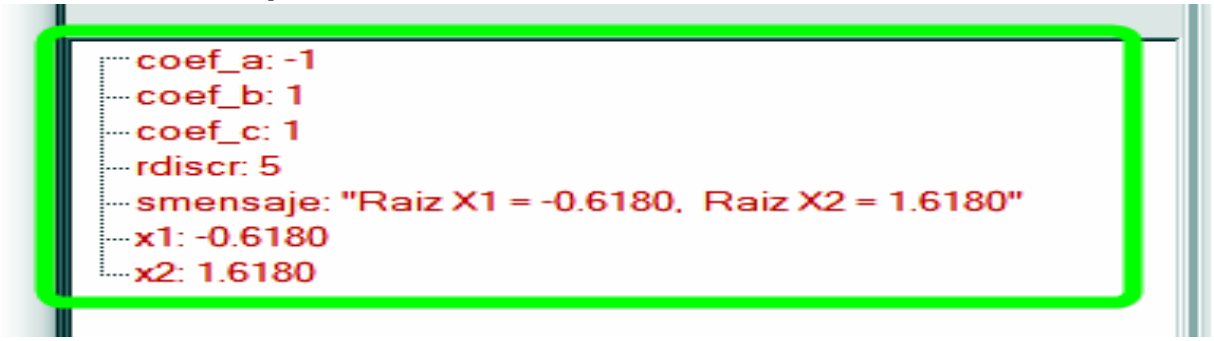


El resultado de la impresión que se muestra en la **Consola de Salida**:



Aquí vemos que la salida es incompleta, ya que si queremos ver a que coeficientes corresponden esas Raíces solo podremos verlas en la ventana de Inspección de RAPTOR, en consecuencia, sería una buena costumbre informar también en la ventana de Salida a que coeficientes corresponden esas raíces.

Ventana de Inspección de RAPTOR:



Con los coeficientes anteriores, si hacemos A=0, tendremos un **Error de ejecución** del programa:

“Can't divide by zero.” = “No se puede dividir por 0”.

Si hacemos A=1 y conservamos B y C, tendremos un **Error de ejecución** del programa:

“Can't take square root of negative number” =
“No se puede sacar la raíz cuadrada del número negativo "

Tendríamos que considerar otro Error posible cuando esperamos el ingreso de Números y el usuario ingresa una Letra, tendremos un **Error de ejecución** del programa del tipo:

“Can't multiply string: B” = “No se puede multiplicar letra B ”

Para evitar este problema, deberíamos **VALIDAR** o Verificar que los valores numéricos ingresados por el Usuario sean **NUMEROS**.