INSTITUTO POLITÉCNICO NACIONAL

UNIDAD PROFESIONAL INTERDISCIPLINARIA EN INGENIERÍA Y TECNOLOGÍAS AVANZADAS

Protocolo

“Prototipo de dron avanzado para el transporte aéreo de suministros, mediante cuatro hélices, que permiten el ascenso y descenso por medio de estructura bípeda.”

Presentan:

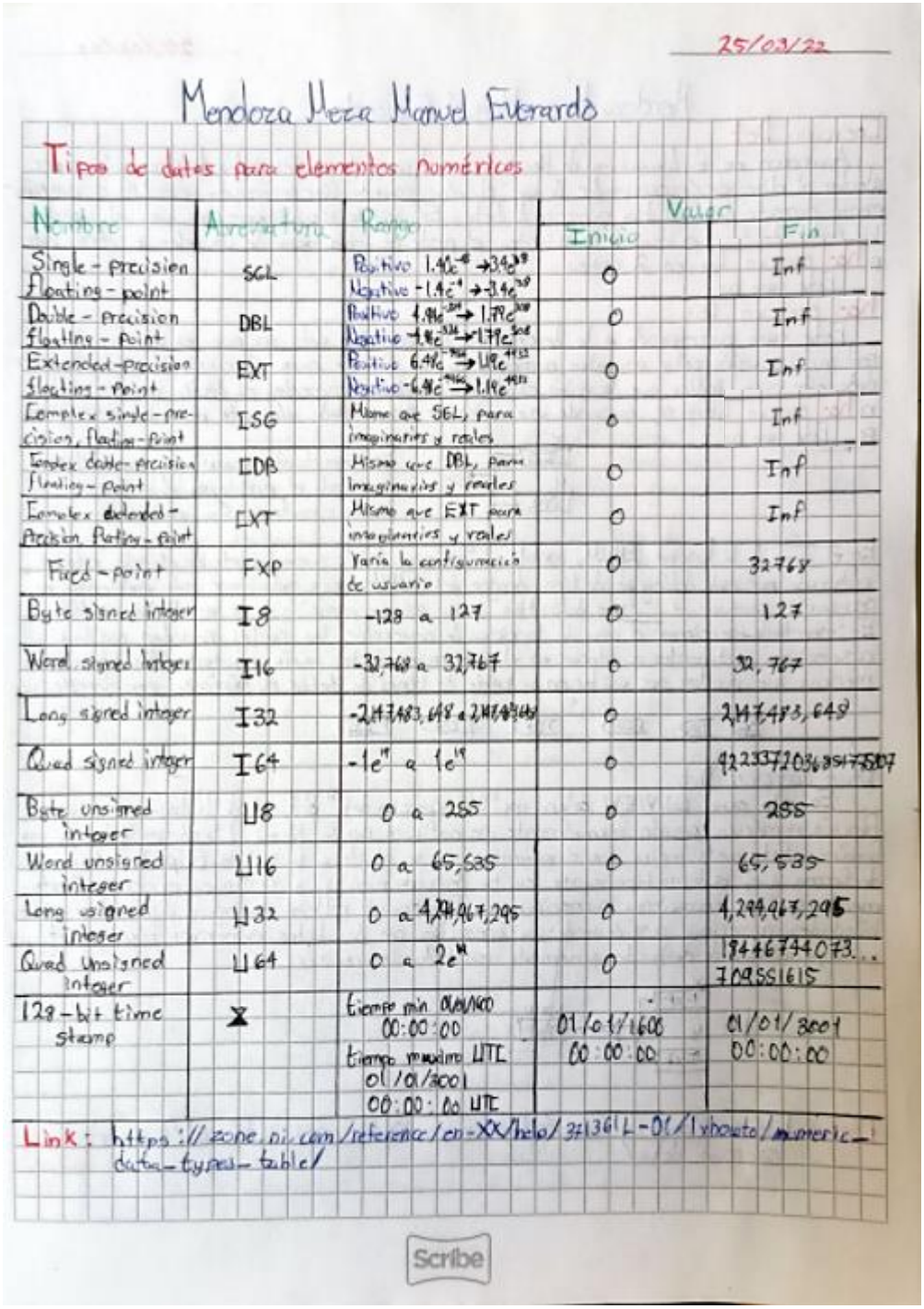
Mendoza Meza Manuel Everardo

Profesor:

Huitron Ramírez Erick

25 de marzo de 2022

upiita-ipn



25/03/22

Mendoza Meza Manuel Everardo

Tipos de datos para elementos numéricos

Nombre	Abreviatura	Rango	Valor Inicio	Valor Fin
Single-precision floating-point	SGL	Positivo $1.4e^{-45} \rightarrow 3.4e^{38}$ Negativo $-1.4e^{-45} \rightarrow -3.4e^{38}$	0	Inf
Double-precision floating-point	DBL	Positivo $4.9e^{-324} \rightarrow 1.79e^{308}$ Negativo $-4.9e^{-324} \rightarrow -1.79e^{308}$	0	Inf
Extended-precision floating-point	EXT	Positivo $6.4e^{-4951} \rightarrow 1.1e^{4932}$ Negativo $-6.4e^{-4951} \rightarrow -1.1e^{4932}$	0	Inf
Complex single-precision floating-point	CSG	Mismo que SGL, para imaginarios y reales	0	Inf
Complex double-precision floating-point	CDB	Mismo que DBL, para imaginarios y reales	0	Inf
Complex extended-precision floating-point	CXT	Mismo que EXT para imaginarios y reales	0	Inf
Fixed-point	FXP	Varía la configuración de usuario	0	32767
Byte signed integer	I8	-128 a 127	0	127
Word signed integer	I16	-32,768 a 32,767	0	32,767
Long signed integer	I32	-2,147,483,648 a 2,147,483,647	0	2,147,483,647
Quad signed integer	I64	$-1e^{19}$ a $1e^{19}$	0	9223372036854775807
Byte unsigned integer	U8	0 a 255	0	255
Word unsigned integer	U16	0 a 65,535	0	65,535
Long unsigned integer	U32	0 a 4,294,967,295	0	4,294,967,295
Quad unsigned integer	U64	0 a $2e^{19}$	0	18446744073... 709551615
128-bit time stamp	⧗	tiempo mín 01/01/100 00:00:00 tiempo máximo UTC 01/01/3001 00:00:00 UTC	01/01/1600 00:00:00	01/01/3001 00:00:00

Link: https://zone.ni.com/reference/en-XX/help/371361L-01/lvhowto/numeric_data_types_table/

20/03/22

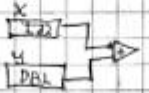
Mendoza Meza Manuel Everardo

Coercion Dot

Aparecen en el diagrama de bloque, específicamente en los nodos de coerción para alertar al diseñador/programador de que el valor ocupado (que genere una unión) no es numéricamente correcto dado el tipo numérico de dato. Es posible visualizarse, ya que, al conectar los nodos de dos funciones o constantes, el punto de unión pasa de un color a otro, esto puede presentarse en 2 casos.

Red coercion Dot

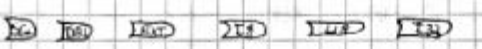
Este caso corresponde a la descripción del coercion dot, ya que es necesario realizar una unión entre un valor numérico y una función para observar el comportamiento dado este caso, indica que la unión es tipo numérica incorrecta, es decir, el valor valido no corresponde entre sí, causando una unión no recomendado y/o apto para el programa por ejemplo:



Es importante mencionar que incluso mostrado el problema el programa puede ejecutarse, sin problema.

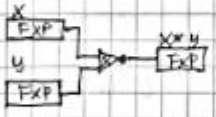
En el caso de la función ENUM, existe el "coercion dot", normalmente es el rojo, esto es provocado por mala definición de tipo, dentro del ENUM, para solucionar este problema es necesario cambiar el "type definition" para obtener precisión y un resultado válido.

Es importante mencionar el uso de funciones de conversión, las cuales cuidan con las conversiones pertinentes a colocar en el programa, estas varía en todos los tipos numéricos que pueden ser vistos en la tabla de tipos de datos numéricos. por ejemplo:



Blue coercion Dot

En este caso LabVIEW coloca un "Blue coercion dot" al final de la terminal de una función numérica cuando manualmente se configura en la forma "Fixed-point" para una función. Este punto indica que el programa no se ajusta a la salida (tipo de dato), de forma que, automáticamente evita desbordamientos y continuos errores. Ciertamente realizar operaciones numéricas con fixed-point refleja muchas y diferentes posibilidades numéricas, considerando una operación con dos datos numéricos fixed point que llega al bole de desborde generando resultados imprecisos.



Link: https://zone.ni.com/reference/en-XX/help/371361R-01/lvconcepts/coercion_dots/

25/03/22

Mendoza Meza Manuel Eduardo

Polymorphic Functions (Funciones Polimórficas)

Polimorfismo es la habilidad de funciones para (de forma automática) adaptar y aceptar los datos de entrada de diferentes tipos de datos. En este sentido, las funciones pueden variar, no variar, tener alguna o todas sus entradas de forma polimórfica. Sin embargo algunas pueden aceptar valores numéricos o cadenas de caracter, otras pueden tener arreglos de clusters de valores numéricos y de esa forma para muchas otras configuraciones.

Polimorfismo para Numericos (Funciones)

Las funciones aritméticas pueden tomar data numerica en su entrada, considerando no caer en los coercion dots, siguiendo la siguiente fórmula es como se puede tener una idea del polimorfismo en funciones numéricas

Numeric type = numeric scalar OR array OR cluster
[numeric type] [numeric type]

Sin embargo, lo descrito sirve para una entrada, a continuación se presentan las posibles combinaciones para 2 entradas

- Similar: Ambas entradas tienen la misma estructura y salida
- One scalar: Una entrada es numerica y la otra es un arreglo o cluster
- Array of: Una entrada es numerica array y la otra es numerica tipo si mismo

Polimorfismo para Booleanos (Funciones)

Además de numericas las funciones lógicas son aquellas que aceptan valores booleanos de entrada, numericos y error clusters. Todas las entradas numericas, de tipo entero, LabVIEW procederá con una operación bitwise ejecutando el programa lanzando un resultado con el mismo tipo. Mostrando igualmente una formula para funciones booleanas.

Logical type = Boolean scalar OR numeric scalar OR array OR cluster
[logical type/s]

Adición de Funciones y Arreglos+Arreglo

Existen igualmente casos para Arreglos donde la mayoría acepte n-dimensional arrays de cualquier tipo. String como son: length, Upper Case, Lower Case, Reverse String and Byte String Conversion Functions (String) donde se ocupan Path to String y String to Path para valores escalares, arreglos y escalares, cluster de escalar, etc. Additional String to Number Functions donde: Number to Decimal, To Hex String, To Octal String, To Engineering String, To Fractional String y To Exponential String son aceptados. Cluster en el cual se ocupan las funciones Bundle y Unbundle no mostrando ninguna salida hasta que sea colocado algo o esas removidas. Comparison Functions donde: Equal?, Not Equal? and Select aceptan cualquier tipo de entrada del mismo tipo, mientras que: Greater Or Equal?, Less Or Equal?, Less?, Greater?, Max & Min y In Range and Coerce toman entradas a excepción de complejos. Logarithms toma cualquier tipo de entrada numerica siguiendo: Numeric type = Numeric scalar OR Array OR Cluster
[numeric type/s]
Siguiendo las 2 reglas:

- Similar: Ambas entradas tienen la misma estructura y salida
- One scalar: Una entrada es escalar numerico y la otra arreglo/cluster cuya salida es arreglo o cluster

Link: https://zone.ni.com/reference/en-XX/help/371361R-01/lvconcepts/polymorphic_functions/