Guía Fortran

Fortran – lenguaje de programación que realiza cálculos numéricos e intenta resolver problemas numéricos.

**Editores de programas**

* Gedit
* Emacs
* Geany
* Blue fish

**Comandos**

* Cd – change directory
* Mkdir – crear carpetas
* PRINT – muestra el mensaje en la pantalla
* READ – lee el mensaje que el usuario escribió en el programa.
* -o – cambia el nombre del ejecutable
* ./a.out = ./ejecutable

**Lenguaje de alto nivel** – usa el lenguaje del usuario (inglés) en vez del lenguaje de la computadora (binaria)

* **Por interpretación** – interpretación inmediata de las instrucción

\*Rápida detección de errores

\*Los cálculos son algo lentos

Ejemplos: Python, mathematica, maple

* **Por compilación** – las instrucciones se escriben en un compilador.

\*compilador Front-End: verifica las instrucciones

\*compilador Back-End: produce un archivo ejecutable.

-se diseñan los algoritmo

-los cálculos son rapidos

Ejemplos: C, C++, Fortran

**Variables** – valores que el usuario puede introducir en el programa para hacer los cálculos.

**Constantes** – valores asignados (o predeterminados) en la ejecución de un programa que no pueden modificarse después.

Al declarar una variable esta debe de tener

* Puras letras y numeros
* Puede tener \_
* No debe de tener espacios, (no sin comillas)
* Siempre debe de empezar con letras, no números.
* Deberá ser compuesta mayormente por letras, no números
* No deben de terminar con punto

**Tipos Intrínsecos en Fortran**

Nos ayudan a manejar variables y constantes de diferente naturaleza.

* **CHARACTER** – manejo de variables en forma de letras o palabras.

Si la variable está entre comillas entonces es de tipo CHARACTER

CHARACTER(len=10) la cantidad de letras en la variable (contando el espacio)

* **INTEGER** – permite manejar cantidades numéricas sin punto decimal.
* **REAL** – manejo de cantidades numéricas con punto decimal

Problemas!!!

~ **Overflow** – número máximo de valores representables

~ **Underflow** – número mínimo de valores representables

~ **Error de redondeo** – hay un número entre dos variables representadas y la maquina usa el valor anterior o superior.

¡Para evitar esto se incrementan los bits disponibles!

\*32 bits (precisión sencilla)

\*64 bits (precisión doble)

\*128 bits (precisión cuádruple)

\*REAL(8) :: etiqueta

\*REAL(4) :: etiqueta

\*REAL(16) :: etiqueta

* SELECTED\_REAL\_KIND

**Características**

~precisión – cantidad de dígitos significativos de la variable

~rango – intervalo de números que representan a una variable

* **COMPLEX** – manejo de cantidades complejas (reales e imaginarias)

**Funciones Intrínsecas**

**~REAL(C1):** regresa la parte real del complejo C1

PRINT\*, “la parte real de C1 es”, REAL(C1)

**~INT(C1):** regresa la parte real en tipo INTEGER.

PRINT\*, “la parte real de C1 es”, INT(C1)

**~AIMAG(C1):** regresa la parte imaginaria del complejo

PRINT\*, “la parte imaginaria de C1 es”, AIMAG(C1)

**~CABS(C1):** regresa la magnitud C1=

PRINT\*, “la magnitud de C1 es”, CABS(C1)

**~CONJG(C1):** muestra el conjugado del complejo

PRINT\*, “el conjugado es”, CONJG(C1)

**~CMPLX(C1):** almacena dos reales (a1, a2) en el complejo C1.

PRINT\*, “la parte real de C1”

READ\*, C1

PRINT\*, “la parte imaginaria de C1”

READ\*, C1

* **LOGICAL** – variables que solo admiten dos respuestas posibles: verdadero o falso.

~Permiten controlar ciertas secciones del código.

~Expresiones lógicas: expresiones que solo tienen respuestas verdaderas o falsas.

**Sintaxis de IF**

**IF (expresión lógica) THEN**

**Secuencia de instrucciones**

**END IF**

**Si la expresión lógica es verdadera se mostrara el mensaje, si no, se omite**

**Existen dos tipos de operadores**

1. **Operadores de relación:** determina la relación entre dos tipos de cantidades

==) igualdad

>) mayor que

<) menor que

>=) mayor o igual que

<=) menor o igual que

/=) diferente a

**El resultado a todos estos debe de ser verdadero o falso.**

1. **Operadores de condición:** permite combinar dos o más expresiones.

.AND. Operador y

.OR. Operador o

.EQV. Equivalencia

.NEQV. No equivalencia

.NOT. Negación

**Jerarquía de Operadores**

1. Todas las operaciones aritméticas
2. Operadores de relación
3. .NOT.
4. .AND.
5. .OR.
6. .EQV. + .NEQV.

**¡Paréntesis cambia el orden de operación!**

Tabla de verdad para determinar el resultado de los operadores de combinación

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| L1 | L2 | L1 .AND. L2 | L1 .OR. L2 | L1 .EQV. L2 | L1 .NEQV. L2 |
| T | **T** | **T** | **T** | **T** | **F** |
| T | **F** | **F** | **T** | **F** | **T** |
| F | **T** | **F** | **T** | **F** | **T** |
| F | **F** | **F** | **F** | **T** | **F** |

**Aritmética en Fortran**

Operadores

* Suma(+)
* Resta(-)
* Producto(\*)
* División(/)
* Exponenciación(\*\*)

2 o más operadores no pueden aparecer seguidos

**2\*\*-3**

**2\*\*(-3)**

¡No existe la multiplicación implícita!

**X(y+z)**

**X\*(y+z)**

**Orden de jerarquías**

1. Potencia (se realiza de derecha a izquierda)
2. Producto y división (de izquierda a derecha)
3. Suma y resta (de izquierda a derecha)

* INTEGER = aritmética entera
* REAL = aritmética con punto decimal

**REGLA:** toda operación realizada con cantidades de un tipo particular tiene resultados del mismo tipo

¾=0 resultado debe de ser entero

Para obtener números con punto decimal

3./4.=0.75

**Buenas prácticas de un programador**

* Especificación
* Diccionario de variables
* **¡Comentarios!**
* Nombres significativos de constantes y variables
* Algoritmo utilizado
* Codificación
* Funcionamiento

Portátil

Fácil Modificación

**Como convertir números reales a enteros.**

1. FLOOR – cambia el número real del complejo al entero inmediato inferior (3.78 – 3)
2. CEILING – cambia el número real a un entero inmediato superior (2.21 – 3)
3. NINT – redondea al entero más cercano (4.68 -5)
4. INT – trunca al entero (8.71 – 8) **borra los decimales y deja al entero solo.**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

* ABS - muestra el valor absoluto de una variable (parte real del complejo)
* ATAN – valor absoluto de una variable imaginaria
* 0.0001, 1.0E4 - no son el mismo valor
* *“Who are you?”* (constante) – **VÁLIDA :** es válida como constante debido al uso de comillas.
* PROGRAM 3rd – **INVÁLIDA:** no debe de empezar con un número
* **SELECTED\_REAL\_KIND** – ayuda a establecer el número de cifras que se puede usar en variables reales

Integer, parameter :: k = selected\_real\_kind (p, r)

* **IMPLICIT NONE** - que se compile con un Fortran actualizado.

¿Qué valor se guarda en el resultado después de realizar las siguientes instrucciones? \_

REAL :: a, b, c, resultado

a = 10.

b = 1.5

c = 5.

Resultado = FLOOR(a/b) + b\*c\*\*2

Solo el (a/b) se convierte en entero

 ¿Qué valor se guarda en resultado después de realicen las siguientes instrucciones?

Respuesta = 43.5