Programación y Diseño Algorítmico

F. Hidrobo, K. Tucci, M. Uzcátegui

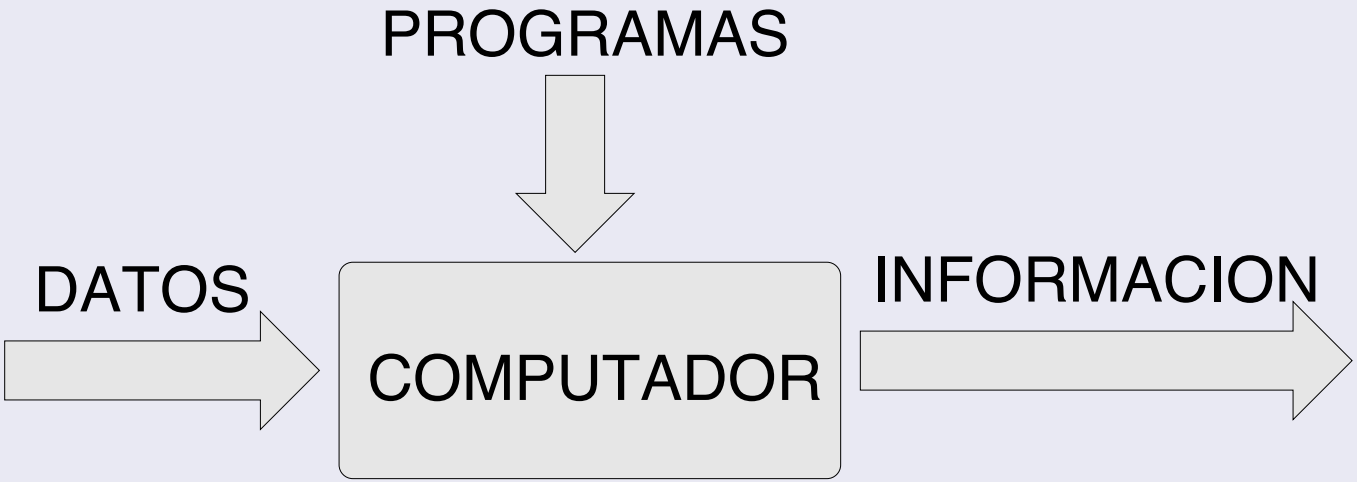
Universidad de Los Andes
Facultad de Ciencias
SUMA

{hidrobo,kay,maye}@ula.ve

Objetivo

Este curso tiene como objetivo brindar los elementos necesarios para que el estudiante sea capaz de resolver adecuadamente problemas concretos haciendo uso del computador como herramienta de trabajo.

Procesamiento de Información

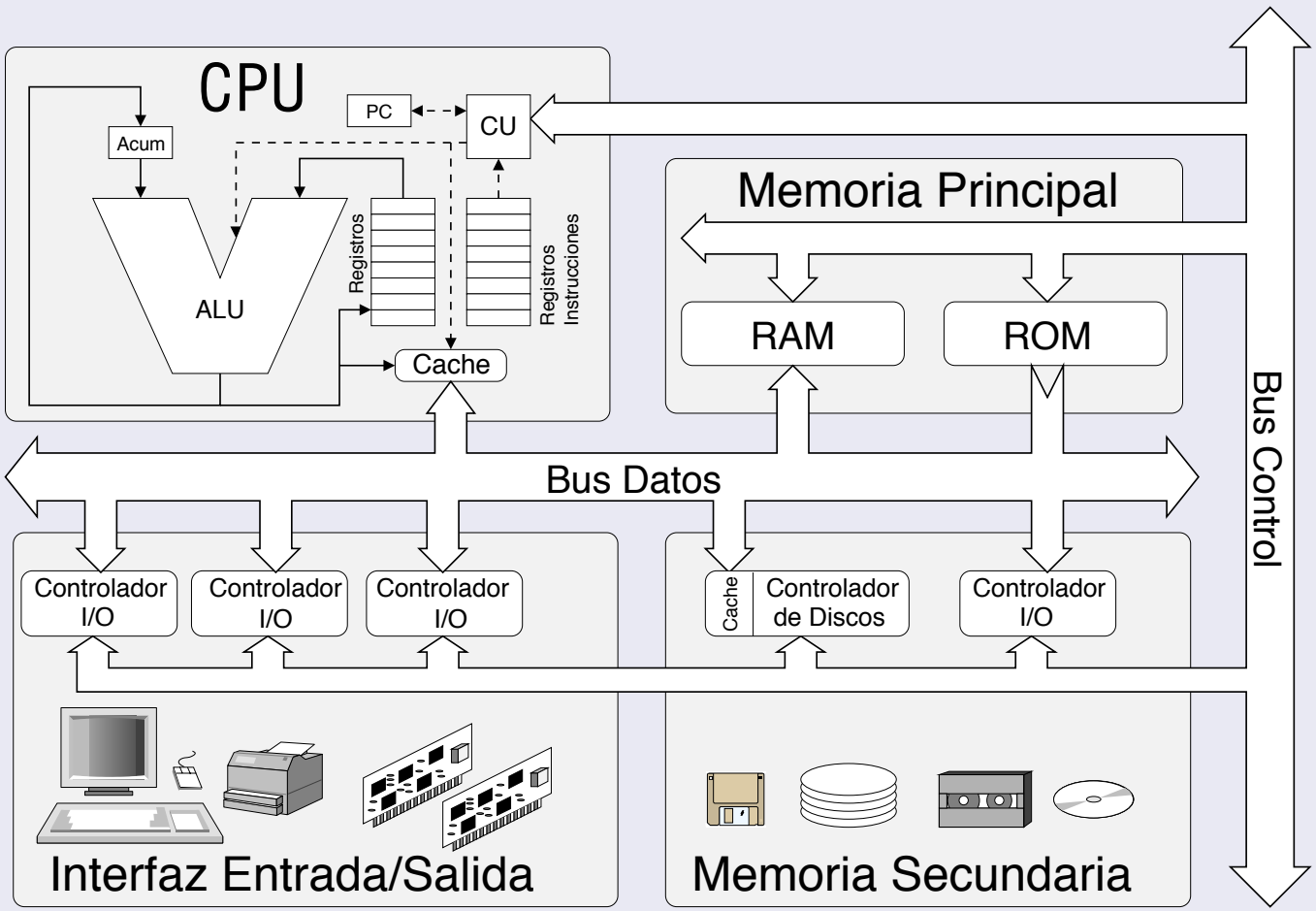


Hardware: Componentes físicos y dispositivos de E/S

Programa: Instrucciones a ser ejecutadas por un computador

Software: Programas escritos para un computador

Computador


CPU
PC
CU
Acum
ALU
Registros
Registros Instrucciones
Cache
Memoria Principal
RAM
ROM
Bus Datos
Bus Control
Controlador I/O
Controlador I/O
Controlador I/O
Cache
Controlador de Discos
Controlador I/O
Interfaz Entrada/Salida
Memoria Secundaria

Dispositivos E/S

Los dispositivos de E/S permiten la comunicación ente el usuario y el computador.

Dispositivos de Entrada

- Teclado
- Ratón
- Lector de discos
- Scanner $\cdots$

Dispositivos de Salida

- Impresora
- Monitor
- Red $\cdots$

Memoria

Memoria principal (primaria)

- Conjunto de celdas de memoria que se direccionan a través de un único nombre.
- Acceso directo por referencia para:
  Carga y Recuperación de información
- Residen los programas en ejecución

Memoria auxiliar (secundaria)

- Permite almacenar gran cantidad de información
- Información persistente
- Ejemplos: Discos duros, CDRom, DVD, $\cdots$

Unidad Central de Procesamiento

Unidad de Control

- Carga instrucciones en memoria
- Interpreta
- Devuelve el resultado de la ejecución

Unidad Aritmética y Lógica

- Procesa operaciones aritméticas y lógicas
- Provee decisión a la Unidad de Control

Solución de Problemas

Análisis $\Rightarrow$ Diseño Algorítmico $\Rightarrow$ Programa

Análisis: El problema debe ser claramente especificado y entendido.

Diseño algorítmico: Construcción de una solución del problema posible de ser ejecutada.

Programa: Traducción del algoritmo a un lenguaje de computación.

Análisis

- Debe acotarse el campo de acción del problema
- Requiere la clara definición del problema
- Debe indicarse que va hacer el programa y cual ve a ser el resultado.
- Debe detallarse las especificaciones de entrada y salida
- En la resolución de un problema complejo, se divide en varios sub problemas y seguidamente se vuelven a dividir los sub problemas en otros mas sencillos, hasta que puedan implementarse en el computador.

Término Algoritmo

proviene del matemático persa, *Muhammad ibn Mūsā al − Khwārizmī* (780-850). Conocido como *Al − Khwārizmī*
Escribió libros sobre geografía, astronomía y matemática.
En su obra Artimética, traducida al latín en el siglo 12 como

Algoritmi de numero Indorum

explica con detalle el funcionamiento del sistema decimal que usaban en la India y los algoritmos para calcular con este sistema de numeración.
Esta obra contribuyó a la difusión en occidente del sistema de numeracion indio y al conocimiento del cero.

Algoritmo

Se refiere a una secuencia ordenada y finita de pasos, excenta de ambigüedades, que seguidas en su orden lógico nos conduce a la solución de un problema específico.

	Estar compuesto de pasos o instrucciones simples.
	Seguir un orden preestablecido.
Claridad:	Expresado en términos específicos, describir las situaciones excepcionales.
Riguroso:	No tener ambigüedades.
Efectivo:	Resolver el problema de forma general y en un número finito de pasos.
Preciso:	Bajo las mismas condiciones el algoritmo siempre debe arrojar los mismos resultados.

Algoritmo

En general las instrucciones básicas que se pueden dar en un algoritmo son:

Solicitud:	Los datos provienen de alguna fuente externa al algoritmo
Asignación:	Darle un valor específico a una variable
Decisión:	Acción de seleccionar, de acuerdo a alguna condición, la siguiente instrucción
Repetición:	Indica que una o varias instrucciones se deben efectuar más de una vez
Generación:	proporcionar los resultados.

Algoritmo

En general las instrucciones básicas que se pueden dar en un algoritmo son:

Solicitud:	Los datos provienen de alguna fuente externa al algoritmo
Asignación:	Darle un valor específico a una variable
Decisión:	Acción de seleccionar, de acuerdo a alguna condición, la siguiente instrucción
Repetición:	Indica que una o varias instrucciones se deben efectuar más de una vez
Generación:	proporcionar los resultados.

Pascal

- Lenguaje de alto nivel que usaremos en el curso.
- Fue diseñado y desarrollado por N. Wirth alrededor del 1968.
- Este lenguaje permite expresar principios de programación y de diseño de solución de problemas en forma abstracta y estructurada.
- Una instrucción Pascal equivale a varias instrucciones de lenguaje de máquina.

Estructura

NombrePrograma.pas

```
Program NombrePrograma;

Declaraciones

Begin

   Secuencia de Instrucciones

End.
```

Primer programa

Hola mundo...

uno.pas

```
Program uno;
Uses Crt;
Begin


  Write('Hola mundo...');
End.
```

```
% fpc uno.pas
% ./uno
```

Primer programa

Hola mundo...

uno.pas

```
Program uno;
Uses Crt;
Begin
  clrscr;

  Write('Hola mundo...');
End.
```

```
% fpc uno.pas
% ./uno
```

Primer programa

Hola mundo...

uno.pas

```
Program uno;
Uses Crt;
Begin
  clrscr;
  gotoxy(10,3);
  Write('Hola mundo...');
End.
```

```
% fpc uno.pas
% ./uno
```

Primer programa

Hola mundo...

uno.pas

```
// Primer programa en pascal
Program uno;
Uses Crt;
Begin
  clrscr;
  gotoxy(10,3);
  Write('Hola mundo...');
End.
```

```
% fpc uno.pas
% ./uno
```

Matlab

- Lenguaje de alto nivel que usaremos en el curso.
- Fue creado por Cleve Moler en 1984, con base en el lenguaje M (1970) que proporcionaba un sencillo acceso al software de matrices LINPACK y EISPACK sin tener que usar Fortran.
- Este lenguaje permite expresar principios de programación y de diseño de solución de problemas en forma abstracta y estructurada.
- Una instrucción Matlab equivale a varias instrucciones de lenguaje de máquina.

Primer programa matlab

Hola mundo...

uno.m

```
>> disp('Hola mundo');
```

Primer programa matlab

Hola mundo...

uno.m

```
% Primer programa en matlab
>> disp('Hola mundo');
```

Diseño descendiente

Técnica de diseño en la que se ataca primero el problema general, descomponiéndolo en problemas más pequeños, los cuales dependiendo de su complejidad se resolverán descomponiéndolos.

Este proceso continúa hasta que los problemas restantes son triviales.

Representación

Los algoritmos pueden ser expresados de muchas maneras, incluyendo:

- lenguaje natural,
- pseudocódigo,
- diagramas de flujo,

Las descripciones en lenguaje natural tienden a ser ambiguas y extensas. El usar pseudocódigo y diagramas de flujo evita muchas ambigüedades del lenguaje natural. Dichas expresiones son formas más estructuradas para representar algoritmos.

independientes de un lenguaje de programación específico

Pseudocódigo

- Es la descripción de un algoritmo que asemeja a un lenguaje de programación pero con algunas convenciones del lenguaje natural.
- Tiene varias ventajas entre las que se destaca el poco espacio que se requiere para representar instrucciones complejas.
- No está regido por ningún estándar.
- pseudo viene de falso y por ende es un código que no se aplica al proceso que debe realizar la maquina.

Identificadores

- Los identificadores se usan para darle nombre a las cosas en el algoritmo.
- Un identificador está formado por una letra o el caracter _ seguido o no por una secuencia de letras, dígitos y/o _
- Son identificadores válidos `A345x` , `_234` , `mI_nomBrE` y `Peso_2_1` .
- No se puede incluir espacio en blanco entre las palabras en los identificadores.
- Los identificadores no pueden comenzar con un dígito.

Palabras reservadas

Existen una serie de secuencias válidas como identificadores que no pueden ser usadas con tal fin debido a que tienen una función preestablecida.
Entre las palabras reservadas tenemos a todos identificadores predefinidos de

constantes: (`Pi`, ...),

funciones: (`sin`, `cos`, `ln`, `exp`, ...)

procedimientos: (`obtener`, `devolver`, ...).

secuencias para estructuras de programación: (`si`, `sino`, `según`, `mientras`, `repita`, `desde`, ...).

Símbolos especiales

El conjunto de caracteres ASCII consta de 256 elementos algunos de los cuales por tener un significado especial en los algoritmos y en los lenguajes de programación se denominan símbolos especiales:

- operadores aritméticos: + - * / ÷ %
- operadores lógicos: ¬ ∧ ∨
- símbolos relacionales: = < > ≠ ≥ ≤
- símbolos de agrupación: { } ( ) [ ]
- signos de puntuación: . , ; : ! ? ”

Por Su uso está definido por la reglas sintácticas y semanticas del pseudocódigo y de los lenguajes de programación.

Tipos de Datos

- Los datos en el computador son almacenados como secuencias binarias, es decir secuencias de ceros y unos.
- En el algoritmo y en los programas tales secuencias se interpretan asociando un tipo de dato a cada dato y a cada expresión.
- Los datos son valores de determinado tipo; se asignan a variables, se asocian a constantes o resultan de la evaluación de expresiones.
- Datos ordinales son aquellos datos que pueden ordenarse realizando una asociación uno a uno con el conjunto (o un subconjunto) de los números enteros.

Tipos de Datos

- datos
  - simples
    - predefinidos
      - Entero* $\mathcal{Z}$
      - Real $\mathcal{R}$
      - Caracter* ch
      - Lógico* $\mathcal{L}$
    - construidos
      - Enumerado*
      - Rango*
  - estructurados
    - predefinidos
      - Archivo
    - construidos
      - Arreglo
      - Cadena $\mathcal{C}$
      - Registro
      - Conjunto

*: Ordinal

Constantes

Valores asociados a un identificador que permanecen inalterados durante la solución del problema

- **Numéricas enteras**
- **Numéricas punto flotante**
- **Caracter**
- **Cadenas de caracteres**

Variables

- Se asocian a un identificador y son utilizadas para almacenar datos que pueden ser modificados a medida que el algoritmo avanza.
- El identificador de cada variable tiene asociado un tipo de dato, el cual no debe cambiar a lo largo del algoritmo. Lo que puede cambiar es el valor contenido en la variable.
- Siempre se debe asignarle el valor a una variable antes de que esta intervenga en una expresión.
- El valor es lo que se utiliza en las expresiones una vez haya sido interpretado según el tipo correspondiente a esa variable.

Cálculo de área de triangulo

Planteamiento

Dado un triángulo, a partir de las dimensiones correspondientes a su altura y a la base, calcular el área del mismo.

Cálculo de área de triangulo

Diseño

Entrada

$$base, altura \in \mathcal{R}^+$$

Salida

$$area \in \mathcal{R}$$

Pseudocódigo

Obtener($base$, $altura$)

$area \leftarrow base * altura/2$

Devolver($area$)

Cálculo de área de triangulo

Código Pascal

areatriangulo.pas

```
Program areatriangulo;
Uses Crt;
Var
  altura, base, area: Real;
Begin
  Read(altura,base);
  area :=  base * altura/2;
  Write(area:8:2);
End.
```

```
% fpc areatriangulo.pas
% ./areatriangulo
```

Cálculo de área de triangulo

Código Matlab

areatriangulo.m

```
% Programa areatriangulo
altura = input('Altura: ');
base = input('Base: ');
area =  base * altura/2;
fprintf('Area = %.2f',area);
```

Sentencias

Las instrucciones simples son aquellas que realizan una única acción y de ellas no depende directamente la ejecución o no de otras instrucciones. Pueden ser de dos tipos:

Asignación: La asignación es la instrucción que permite almacenar un valor en una variable, es decir, el resultado de evaluar una expresión es almacenado en el espacio de memoria reservado a una variable.

Llamada a subprograma: permite hacer uso de las librerías del sistema y de los módulos desarrollados por el programador.

Asignación

Hay que destacar que el tipo de dato resultante de la evaluación de la expresión debe ser compatible con el tipo de dato de la variable, esto es, que el resultado de la expresión pueda ser almacenado e interpretado correctamente como el valor contenido en la variable.

Pseudocódigo

IdVariable ← *Valor*

Código Pascal	Código Matlab
IdVariable := Valor;	IdVariable = Valor;

al asignarle un valor a una variable se pierde irremediablemente el valor que tenía asignado anteriormente

Compatibilidad de Tipos

Variable	Expresión
$\mathcal{N}$	$\mathcal{N}$
$\mathcal{Z}$	$\mathcal{N}$, $\mathcal{Z}$
$\mathcal{R}$	$\mathcal{N}$, $\mathcal{Z}$, $\mathcal{R}$
$\mathcal{L}$	$\mathcal{L}$
ch	*ch*

Además de la compatibilidad, es importante garantizar, ya en el momento de la implementación del código del programa, que el resultado de la expresión no desborde la capacidad del tipo de dato de la variable

Llamada a subprogramas

En caso de que exista un tipo de dato arrojado al ejecutar un subprograma este es determinado directamente por él.
Los subprogramas básicos de entrada salida de datos son:

Pseudocódigo

Obtener(lista de variables) *Leer(lista de variables)*
Devolver(lista de valores) *Escribir(lista de valores)*

Código Pascal

Read(*lista de variables*);
Write(*lista de valores*);

Código Matlab

variable = input('*mensaje*');
variable

Expresiones

- Las expresiones son estructuras utilizadas en los algoritmos para calcular nuevos valores.
- Su estructura está compuesta por uno o más operandos separados por operadores.
- Operandos y operadores pueden ser agrupados para darle claridad a una expresión o para forzar un orden específico de evaluación.
- Los operandos están conformados por expresiones (constantes, variables, funciones, o una combinación de ellos)

Clasificación de los Operadores

- Pueden ser unarios y binarios según el número de operandos sobre los que actúan.
- De acuerdo al tipo de datos sobre los cuales operan tenemos:

  Aritméticos: que operan sobre valores numéricos;
  Relacionales: que comparan dos valores;
  Lógicos: que actúan sobre valores booleanos.

Operadores

Operación		Pascal	Matlab	A	B	Operación
Asignación:						
Asignación	$\leftarrow A$	:=	=	$\mathcal{Z}$ $\mathcal{R}$ ch $\mathcal{C}$ $\mathcal{L}$		$\mathcal{Z}$,$\mathcal{R}$ $\mathcal{R}$ ch,$\mathcal{C}$ $\mathcal{C}$ $\mathcal{L}$
Lógicos:						
Conjunción	$A \wedge B$	AND	&&	$\mathcal{L}$	$\mathcal{L}$	$\mathcal{L}$
Disjunción	$A \vee B$	OR	||	$\mathcal{L}$	$\mathcal{L}$	$\mathcal{L}$
Negación	$\neg A$	NOT	$\sim$	$\mathcal{L}$		$\mathcal{L}$

Aritméticos:

Suma	$A + B$	+	+	$\mathcal{Z}$	$\mathcal{Z}$	$\mathcal{Z}$
				$\mathcal{Z}$	$\mathcal{R}$	$\mathcal{R}$
				$\mathcal{R}$	$\mathcal{Z}$	$\mathcal{R}$
				$\mathcal{R}$	$\mathcal{R}$	$\mathcal{R}$
Resta	$A - B$	-	-	$\mathcal{Z}$	$\mathcal{Z}$	$\mathcal{Z}$
				$\mathcal{Z}$	$\mathcal{R}$	$\mathcal{R}$
				$\mathcal{R}$	$\mathcal{Z}$	$\mathcal{R}$
				$\mathcal{R}$	$\mathcal{R}$	$\mathcal{R}$
Multiplicación	$A * B$	*	*	$\mathcal{Z}$	$\mathcal{Z}$	$\mathcal{Z}$
				$\mathcal{Z}$	$\mathcal{R}$	$\mathcal{R}$
				$\mathcal{R}$	$\mathcal{Z}$	$\mathcal{R}$
				$\mathcal{R}$	$\mathcal{R}$	$\mathcal{R}$
División	A/B	/	/	$\mathcal{Z}$,$\mathcal{R}$	$\mathcal{Z}$,$\mathcal{R}$	$\mathcal{R}$
Cociente	$A \div B$	`DIV`		$\mathcal{Z}$	$\mathcal{Z}$	$\mathcal{Z}$
Residuo	$A\%B$	`MOD`	`rem`	$\mathcal{Z}$	$\mathcal{Z}$	$\mathcal{Z}$

Relacionales:

Igualdad	$A = B$	=	==	$\mathcal{Z}$,$\mathcal{R}$ *ch*,$\mathcal{C}$ $\mathcal{L}$	$\mathcal{Z}$,$\mathcal{R}$ *ch*,$\mathcal{C}$ $\mathcal{L}$	$\mathcal{L}$ $\mathcal{L}$ $\mathcal{L}$
Desigualdad	$A \neq B$	<>	~=	$\mathcal{Z}$,$\mathcal{R}$ *ch*,$\mathcal{C}$ $\mathcal{L}$	$\mathcal{Z}$,$\mathcal{R}$ *ch*,$\mathcal{C}$ $\mathcal{L}$	$\mathcal{L}$ $\mathcal{L}$ $\mathcal{L}$
Mayor	$A > B$	>	>	$\mathcal{Z}$,$\mathcal{R}$ *ch*,$\mathcal{C}$ $\mathcal{L}$	$\mathcal{Z}$,$\mathcal{R}$ *ch*,$\mathcal{C}$ $\mathcal{L}$	$\mathcal{L}$ $\mathcal{L}$ $\mathcal{L}$
Mayor o igual	$A \geq B$	>=	>=	$\mathcal{Z}$,$\mathcal{R}$ *ch*,$\mathcal{C}$ $\mathcal{L}$	$\mathcal{Z}$,$\mathcal{R}$ *ch*,$\mathcal{C}$ $\mathcal{L}$	$\mathcal{L}$ $\mathcal{L}$ $\mathcal{L}$
Menor	$A < B$	<	<	$\mathcal{Z}$,$\mathcal{R}$ *ch*,$\mathcal{C}$ $\mathcal{L}$	$\mathcal{Z}$,$\mathcal{R}$ *ch*,$\mathcal{C}$ $\mathcal{L}$	$\mathcal{L}$ $\mathcal{L}$ $\mathcal{L}$
Menor o igual	$A \leq B$	<=	<=	$\mathcal{Z}$,$\mathcal{R}$ *ch*,$\mathcal{C}$ $\mathcal{L}$	$\mathcal{Z}$,$\mathcal{R}$ *ch*,$\mathcal{C}$ $\mathcal{L}$	$\mathcal{L}$ $\mathcal{L}$ $\mathcal{L}$

Funciones Básicas

Descripción		Pascal	Matlab	x	f(x)
Potencias:					
Cuadrática	x^2	`SQR(x)`	`pow2(x)`	$\mathcal{R}$	$\mathcal{R}+$
Radical	$\sqrt{x}$	`SQRT(x)`	`sqrt(x)`	$\mathcal{R}+$	$\mathcal{R}+$
Trigonométricas:					
Seno	$sin(x)$	`SIN(x)`	`sin(x)`	$\mathcal{R}$	$\mathcal{R}$
Coseno	$cos(x)$	`COS(x)`	`cos(x)`	$\mathcal{R}$	$\mathcal{R}$
Arco tangente	$tan^{-1}(x)$	`ARCTAN(x)`	`atan(x)`	$\mathcal{R}$	$\mathcal{R}$
Exponenciales y Logarítmicas:					
Exponencial	e^x	`EXP(x)`	`exp(x)`	$\mathcal{R}$	$\mathcal{R}\oplus$
Logaritmo	$ln(x)$	`LN(x)`	`log(x)`	$\mathcal{R}\oplus$	$\mathcal{R}+$
Ordenamiento:					
Siguiente	$sig(x)$	`SUCC(x)`		$\mathcal{Z}$ ch $\mathcal{L}$	$\mathcal{Z}$ ch $\mathcal{L}$
Anterior	$ant(x)$	`PRED(x)`		$\mathcal{Z}$ ch	$\mathcal{Z}$ ch

Funciones Básicas

Conversión:

Truncamiento	$trunc(x)$	`TRUNC(x)`	`fix(x)`	$\mathcal{Z}$,$\mathcal{R}$	$\mathcal{Z}$
Redondeo	$redon(x)$	`ROUND(x)`	`round(x)`	$\mathcal{Z}$,$\mathcal{R}$	$\mathcal{Z}$
Ordinal	$ord(x)$	`ORD(x)`	`abs(x)`	$\mathcal{Z}$,ch,$\mathcal{L}$	$\mathcal{Z}$
Caracter	$chr(x)$	`CHR(x)`	`char(x:x)`	$\mathcal{Z}$	ch
Impar	$impar(x)$	`ODD(x)`	`rem(x,2)`	$\mathcal{Z}$	$\mathcal{L}$

Otras:

Valor Absoluto	$\lvert x \rvert$	`ABS(x)`	`abs(x)`	$\mathcal{Z}$ $\mathcal{R}$	$\mathcal{Z}$ $\mathcal{R}$
Parte Entera	$[\![x]\!]$	`INT(x)`	`fix(x)`	$\mathcal{Z}$,$\mathcal{R}$	$\mathcal{R}$
Parte Decimal	$frac(x)$	`FRAC(x)`	`x-fix(x)`	$\mathcal{Z}$,$\mathcal{R}$	$\mathcal{R}$
Aleatorio	$aleatorio(x)$	`RANDOM(x)`	`randi(x)`	$\mathcal{Z}$	$\mathcal{Z}$

Evaluación de Expresiones

$$2 - 6/2 * 3 - 1$$

¿Al evaluar la expresión anterior cúal es el resultado correcto?
-7, -1, -8.

El problema es la ambigüedad en la forma de evaluar las expresiones en la vida cotidiana. En los algoritmos la evaluación de expresiones se efectúa según una reglas, llamadas precedencia de operadores, que asignan prioridades relativas.

Precedencia de Operadores

1	Paréntesis	()
2	Operadores Unarios y Funciones	$\neg$ sin cos ln exp $\cdots$
3	Multiplicativos	$*/\div\ \%\wedge$
4	Aditivos	$+\ -\ \vee$
5	Relacionales	$=\neq<>\leq\geq$

El resultado de la evaluación de una expresión debe ser alojado en memoria mediante una instrucción de asignación o utilizando los subprogramas para el manejo de las operaciones de entrada y salida.

Ejercicios

Sean $A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$
Calcular el valor de X en cada caso

$$X \leftarrow A/B - C + D * E - A - C$$

$$X \leftarrow A/(B - C) + D * (E - A) - C$$

$$X \leftarrow A/B - C > D * E - A - C$$

Ejercicio 1

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$X \leftarrow A/B - C + D * E - A - C$

Ejercicio 1

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/B - C + D * E - A - C$$
$$X \leftarrow 4/4 - 2 + 3 * 2 - 4 - 2$$

Ejercicio 1

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/B - C + D * E - A - C$$
$$X \leftarrow 4/4 - 2 + 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 + 3 * 2 - 4 - 2$$

Ejercicio 1

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/B - C + D * E - A - C$$
$$X \leftarrow 4/4 - 2 + 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 + 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 + 6 - 4 - 2$$

Ejercicio 1

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/B - C + D * E - A - C$$
$$X \leftarrow 4/4 - 2 + 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 + 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 + 6 - 4 - 2$$
$$X \leftarrow -1 + 6 - 4 - 2$$

Ejercicio 1

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/B - C + D * E - A - C$$
$$X \leftarrow 4/4 - 2 + 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 + 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 + 6 - 4 - 2$$
$$X \leftarrow -1 + 6 - 4 - 2$$
$$X \leftarrow 5 - 4 - 2$$

Ejercicio 1

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/B - C + D * E - A - C$$
$$X \leftarrow 4/4 - 2 + 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 + 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 + 6 - 4 - 2$$
$$X \leftarrow -1 + 6 - 4 - 2$$
$$X \leftarrow 5 - 4 - 2$$
$$X \leftarrow 1 - 2$$

Ejercicio 1

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$
\begin{aligned}
X &\leftarrow A/B - C + D * E - A - C \\
X &\leftarrow 4/4 - 2 + 3 * 2 - 4 - 2 \\
X &\leftarrow 1 - 2 + 3 * 2 - 4 - 2 \\
X &\leftarrow 1 - 2 + 6 - 4 - 2 \\
X &\leftarrow -1 + 6 - 4 - 2 \\
X &\leftarrow 5 - 4 - 2 \\
X &\leftarrow 1 - 2 \\
X &\leftarrow -1
\end{aligned}
$$

Ejercicio 1

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$
\begin{aligned}
X &\leftarrow A/B - C + D * E - A - C \\
X &\leftarrow 4/4 - 2 + 3 * 2 - 4 - 2 \\
X &\leftarrow 1 - 2 + 3 * 2 - 4 - 2 \\
X &\leftarrow 1 - 2 + 6 - 4 - 2 \\
X &\leftarrow -1 + 6 - 4 - 2 \\
X &\leftarrow 5 - 4 - 2 \\
X &\leftarrow 1 - 2 \\
X &\leftarrow -1
\end{aligned}
$$

$X \in \mathcal{R}$

Ejercicio 2

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/(B - C) + D * (E - A) - C$$

Ejercicio 2

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/(B - C) + D * (E - A) - C$$
$$X \leftarrow 4/(4 - 2) + 3 * (2 - 4) - 2$$

Ejercicio 2

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/(B-C) + D * (E-A) - C$$
$$X \leftarrow 4/(4-2) + 3 * (2-4) - 2$$
$$X \leftarrow 4/2 + 3 * (2-4) - 2$$

Ejercicio 2

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/(B-C) + D * (E-A) - C$$
$$X \leftarrow 4/(4-2) + 3 * (2-4) - 2$$
$$X \leftarrow 4/2 + 3 * (2-4) - 2$$
$$X \leftarrow 4/2 + 3 * -2 - 2$$

Ejercicio 2

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/(B-C) + D * (E-A) - C$$
$$X \leftarrow 4/(4-2) + 3 * (2-4) - 2$$
$$X \leftarrow 4/2 + 3 * (2-4) - 2$$
$$X \leftarrow 4/2 + 3 * -2 - 2$$
$$X \leftarrow 2 + 3 * -2 - 2$$

Ejercicio 2

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/(B-C) + D * (E-A) - C$$
$$X \leftarrow 4/(4-2) + 3 * (2-4) - 2$$
$$X \leftarrow 4/2 + 3 * (2-4) - 2$$
$$X \leftarrow 4/2 + 3 * -2 - 2$$
$$X \leftarrow 2 + 3 * -2 - 2$$
$$X \leftarrow 2 - 6 - 2$$

Ejercicio 2

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$
\begin{aligned}
X &\leftarrow A/(B-C) + D*(E-A) - C \\
X &\leftarrow 4/(4-2) + 3*(2-4) - 2 \\
X &\leftarrow 4/2 + 3*(2-4) - 2 \\
X &\leftarrow 4/2 + 3*-2 - 2 \\
X &\leftarrow 2 + 3*-2 - 2 \\
X &\leftarrow 2 - 6 - 2 \\
X &\leftarrow -4 - 2
\end{aligned}
$$

Ejercicio 2

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$
\begin{aligned}
X &\leftarrow A/(B-C) + D*(E-A) - C \\
X &\leftarrow 4/(4-2) + 3*(2-4) - 2 \\
X &\leftarrow 4/2 + 3*(2-4) - 2 \\
X &\leftarrow 4/2 + 3*-2 - 2 \\
X &\leftarrow 2 + 3*-2 - 2 \\
X &\leftarrow 2 - 6 - 2 \\
X &\leftarrow -4 - 2 \\
X &\leftarrow -6
\end{aligned}
$$

Ejercicio 2

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$
\begin{aligned}
X &\leftarrow A/(B-C) + D * (E-A) - C \\
X &\leftarrow 4/(4-2) + 3 * (2-4) - 2 \\
X &\leftarrow 4/2 + 3 * (2-4) - 2 \\
X &\leftarrow 4/2 + 3 * -2 - 2 \\
X &\leftarrow 2 + 3 * -2 - 2 \\
X &\leftarrow 2 - 6 - 2 \\
X &\leftarrow -4 - 2 \\
X &\leftarrow -6
\end{aligned}
$$

$X \in \mathcal{R}$

Ejercicio 3

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$X \leftarrow A/B - C > D * E - A - C$

Ejercicio 3

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$X \leftarrow A/B - C > D * E - A - C$

$X \leftarrow 4/4 - 2 > 3 * 2 - 4 - 2$

Ejercicio 3

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/B - C > D * E - A - C$$
$$X \leftarrow 4/4 - 2 > 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 > 3 * 2 - 4 - 2$$

Ejercicio 3

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/B - C > D * E - A - C$$
$$X \leftarrow 4/4 - 2 > 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 > 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 > 6 - 4 - 2$$

Ejercicio 3

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/B - C > D * E - A - C$$
$$X \leftarrow 4/4 - 2 > 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 > 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 > 6 - 4 - 2$$
$$X \leftarrow -1 > 6 - 4 - 2$$

Ejercicio 3

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/B - C > D * E - A - C$$
$$X \leftarrow 4/4 - 2 > 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 > 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 > 6 - 4 - 2$$
$$X \leftarrow -1 > 6 - 4 - 2$$
$$X \leftarrow -1 > 2 - 2$$

Ejercicio 3

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$X \leftarrow A/B - C > D * E - A - C$$
$$X \leftarrow 4/4 - 2 > 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 > 3 * 2 - 4 - 2$$
$$X \leftarrow 1 - 2 > 6 - 4 - 2$$
$$X \leftarrow -1 > 6 - 4 - 2$$
$$X \leftarrow -1 > 2 - 2$$
$$X \leftarrow -1 > 0$$

Ejercicio 3

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$
\begin{aligned}
X &\leftarrow A/B - C > D * E - A - C \\
X &\leftarrow 4/4 - 2 > 3 * 2 - 4 - 2 \\
X &\leftarrow 1 - 2 > 3 * 2 - 4 - 2 \\
X &\leftarrow 1 - 2 > 6 - 4 - 2 \\
X &\leftarrow -1 > 6 - 4 - 2 \\
X &\leftarrow -1 > 2 - 2 \\
X &\leftarrow -1 > 0 \\
X &\leftarrow F
\end{aligned}
$$

Ejercicio 3

$A \leftarrow 4, \quad B \leftarrow 4, \quad C \leftarrow 2, \quad D \leftarrow 3, \quad E \leftarrow 2$

$$
\begin{aligned}
X &\leftarrow A/B - C > D * E - A - C \\
X &\leftarrow 4/4 - 2 > 3 * 2 - 4 - 2 \\
X &\leftarrow 1 - 2 > 3 * 2 - 4 - 2 \\
X &\leftarrow 1 - 2 > 6 - 4 - 2 \\
X &\leftarrow -1 > 6 - 4 - 2 \\
X &\leftarrow -1 > 2 - 2 \\
X &\leftarrow -1 > 0 \\
X &\leftarrow F
\end{aligned}
$$

$X \in \mathcal{L}$

Incremento y Decremento

Muchas veces el valor a una variable está relacionado con el valor que dicha variable tiene en el momento en el que se realiza la asignación.

Por ejemplo, trascurrido un año, hay que incrementar en uno la edad de cierta persona.

```
edad ← edad+1
```

Note que, a una variable sólo se le puede asignar valores. Como primero se evalua la expresión y luego se procede a asignar a la variable edad el resultado de la evaluación, el valor de la edad con esta instrucción se incrementa en uno.

Intercambio

Cuando queremos asignar el valor que tiene una variable en otra y viceversa

caso 1	**caso 2**	**caso 3**
`x ← y`	`y ← x`	`aux ← x`
`y ← x`	`x ← y`	`x ← y`
?	?	`y ← aux`

Este es un problema equivalente al de intercambiar el contenido de dos botellas y para resolverlo necesitamos una tercera botella en la que se almacena temporalmente el contenido de una de las dos botellas a intercambiar.

Identificación de Variables

Tome a como la inicial de su apellido, b y c, como los dos últimos dígitos de su cédula.
Identifique los tipos posibles de cada una de las variables y determine el valor de cada una en cada paso del algoritmo.

Algoritmo

obtener(a,b,c)
d $\leftarrow$ a
b $\leftarrow$ ord(b^2-100*(c+1)>0)* 5
e $\leftarrow$ b % ord(a)
f $\leftarrow$ ch(ord(a)+1) = d
c $\leftarrow$ e/b

Identificación de Variables

Tome a como la inicial de su apellido, b y c, como los dos últimos dígitos de su cédula.
Identifique los tipos posibles de cada una de las variables y determine el valor de cada una en cada paso del algoritmo.

Algoritmo

obtener(a,b,c)
d $\leftarrow$ a
b $\leftarrow$ ord(b^2-100*(c+1)>0)* 5
e $\leftarrow$ b % ord(a)
f $\leftarrow$ ch(ord(a)+1) = d
c $\leftarrow$ e/b

a	b	c	d	e	f
'U'	1	1			

Identificación de Variables

Tome a como la inicial de su apellido, b y c, como los dos últimos dígitos de su cédula.
Identifique los tipos posibles de cada una de las variables y determine el valor de cada una en cada paso del algoritmo.

Algoritmo

obtener(a,b,c)
d $\leftarrow$ a
b $\leftarrow$ ord(b^2-100*(c+1)>0)* 5
e $\leftarrow$ b % ord(a)
f $\leftarrow$ ch(ord(a)+1) = d
c $\leftarrow$ e/b

a	b	c	d	e	f
'U'	1	1			
'U'	1	1	'U'		

Identificación de Variables

Tome a como la inicial de su apellido, b y c, como los dos últimos dígitos de su cédula.
Identifique los tipos posibles de cada una de las variables y determine el valor de cada una en cada paso del algoritmo.

Algoritmo

obtener(a,b,c)
d $\leftarrow$ a
b $\leftarrow$ ord(b^2-100*(c+1)>0)* 5
e $\leftarrow$ b % ord(a)
f $\leftarrow$ ch(ord(a)+1) = d
c $\leftarrow$ e/b

a	b	c	d	e	f
'U'	1	1			
'U'	1	1	'U'		
'U'	0	1	'U'		

Identificación de Variables

Tome a como la inicial de su apellido, b y c, como los dos últimos dígitos de su cédula.
Identifique los tipos posibles de cada una de las variables y determine el valor de cada una en cada paso del algoritmo.

Algoritmo

obtener(a,b,c)
d $\leftarrow$ a
b $\leftarrow$ ord(b^2-100*(c+1)>0)* 5
e $\leftarrow$ b % ord(a)
f $\leftarrow$ ch(ord(a)+1) = d
c $\leftarrow$ e/b

a	b	c	d	e	f
'U'	1	1			
'U'	1	1	'U'		
'U'	0	1	'U'		
'U'	0	1	'U'	0	

Identificación de Variables

Tome a como la inicial de su apellido, b y c, como los dos últimos dígitos de su cédula.
Identifique los tipos posibles de cada una de las variables y determine el valor de cada una en cada paso del algoritmo.

Algoritmo

obtener(a,b,c)
d $\leftarrow$ a
b $\leftarrow$ ord(b^2-100*(c+1)>0)* 5
e $\leftarrow$ b % ord(a)
f $\leftarrow$ ch(ord(a)+1) = d
c $\leftarrow$ e/b

a	b	c	d	e	f
'U'	1	1			
'U'	1	1	'U'		
'U'	0	1	'U'		
'U'	0	1	'U'	0	
'U'	0	1	'U'	0	F

Identificación de Variables

Tome a como la inicial de su apellido, b y c, como los dos últimos dígitos de su cédula.
Identifique los tipos posibles de cada una de las variables y determine el valor de cada una en cada paso del algoritmo.

Algoritmo

obtener(a,b,c)
d $\leftarrow$ a
b $\leftarrow$ ord(b^2-100*(c+1)>0)* 5
e $\leftarrow$ b % ord(a)
f $\leftarrow$ ch(ord(a)+1) = d
c $\leftarrow$ e/b

a	b	c	d	e	f
'U'	1	1			
'U'	1	1	'U'		
'U'	0	1	'U'		
'U'	0	1	'U'	0	
'U'	0	1	'U'	0	F
'U'	error	0	'U'	0	F

Identificación de Variables

Tome a como la inicial de su apellido, b y c, como los dos últimos dígitos de su cédula.
Identifique los tipos posibles de cada una de las variables y determine el valor de cada una en cada paso del algoritmo.

Algoritmo

obtener(a,b,c)
d $\leftarrow$ a
b $\leftarrow$ ord(b^2-100*(c+1)>0)* 5
e $\leftarrow$ b % ord(a)
f $\leftarrow$ ch(ord(a)+1) = d
c $\leftarrow$ e/b

a	b	c	d	e	f
'U'	1	1			
'U'	1	1	'U'		
'U'	0	1	'U'		
'U'	0	1	'U'	0	
'U'	0	1	'U'	0	F
'U'	error	0	'U'	0	F

$a, d \in ch \quad b, e \in \mathcal{Z} \quad c \in \mathcal{R} \quad f \in \mathcal{L}$

Potenciación

Dados dos números x, y calcular x^y

$5^2 = 5 \times 5$ $\quad 5^{-2} = \frac{1}{5^2} = \frac{1}{5 \times 5}$ $\quad 5{,}25^{2{,}45}$

$-2^3 = -2 \times -2 \times -2$ $\quad -3^{-2} = \frac{1}{-3^2} = \frac{1}{-3 \times -3}$ $\quad -3{,}124^{-2{,}12}$

Diseño

Entrada:
Salida:
Proceso:

Algoritmo:

Potenciación

Dados dos números x, y calcular x^y

$$5^2 = 5 \times 5 \qquad 5^{-2} = \frac{1}{5^2} = \frac{1}{5 \times 5} \qquad 5{,}25^{2{,}45}$$

$$-2^3 = -2 \times -2 \times -2 \qquad -3^{-2} = \frac{1}{-3^2} = \frac{1}{-3 \times -3} \qquad -3{,}124^{-2{,}12}$$

Diseño

Entrada: $x, y \in \mathcal{R}$
Salida: $z \in \mathcal{R}$
Proceso:

Algoritmo:

Potenciación

Dados dos números x, y calcular x^y

$$5^2 = 5 \times 5 \qquad 5^{-2} = \frac{1}{5^2} = \frac{1}{5 \times 5} \qquad 5{,}25^{2{,}45}$$

$$-2^3 = -2 \times -2 \times -2 \qquad -3^{-2} = \frac{1}{-3^2} = \frac{1}{-3 \times -3} \qquad -3{,}124^{-2{,}12}$$

Diseño

Entrada: $x, y \in \mathcal{R}$, $x > 0$
Salida: $z \in \mathcal{R}$
Proceso:

$$z = e^{y ln(x)}, x > 0$$

Algoritmo:

Potenciación

Dados dos números x, y calcular x^y

$5^2 = 5 \times 5$ $\quad 5^{-2} = \frac{1}{5^2} = \frac{1}{5 \times 5}$ $\quad 5{,}25^{2{,}45}$

$-2^3 = -2 \times -2 \times -2$ $\quad -3^{-2} = \frac{1}{-3^2} = \frac{1}{-3 \times -3}$ $\quad -3{,}124^{-2{,}12}$

Diseño

Entrada: $x, y \in \mathcal{R}$, $x > 0$
Salida: $z \in \mathcal{R}$
Proceso:

$$z = e^{y ln(x)}, x > 0$$

Algoritmo:

Obtener(x, y)
$z \leftarrow e^{y * ln(x)}$
Devolver(z)

Potenciación

potencia.pas

```
Program potencia;
Uses Crt;
Var
   x,y,z : Real;
Begin
   Readln(x,y);
   z:=exp(y*ln(x));
   writeln(z);
End.
```

Potenciación

potencia.m

```
%Programa potencia;
    x = input('x = ');
    y = input('y = ');
    z = exp(y*log(x));
    fprintf('%0.2f',z);
```

Ecuación de Segundo Grado

Planteamiento

Dada la ecuación

$$ax^2 + bx + c = 0$$

Encuentre la solución a dicha ecuación para valores de $a, b, c \in \mathcal{R}$

Solución

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}, \qquad a \neq 0$$

Donde realmente estamos indicando con el signo $\pm$ dos posibles valores para la solución, por lo que tenemos que calcular $x1$ para el caso positivo y $x2$ para el caso negativo

Ecuación de Segundo Grado

Análisis

No nos informan nada respecto al tipo de dato de la solución por ello debemos asumir el caso más general: la cantidad subradical puede ser negativa por lo que el resultado pertenece al conjunto de los números complejos

$$d = b^2 - 4ac, \quad x = \begin{cases} \frac{-b}{2a} \pm \frac{\sqrt{d}}{2a}, & d \geq 0 \\ \\ \frac{-b}{2a} \pm \frac{\sqrt{|d|}}{2a}\ i, & oc \end{cases}$$

Donde realmente estamos encontrando dos valores reales para cada solución, correspondientes a la parte real y a la parte imaginaria respectivamente

$$d = b^2 - 4ac,$$
$$v1 = -\frac{-b}{2a},$$
$$v2 = \frac{\sqrt{|d|}}{2a}$$

$$x = \begin{cases} v1 \pm v2, & d \geq 0 \\ \\ v1 \pm v2\ i, & oc \end{cases}$$

$$d \geq 0 \begin{cases} x1 = v1 + v2 \\ x2 = v1 - v2 \end{cases}$$

Dos soluciones reales

$$d < 0 \begin{cases} x1 = v1 + v2\ i \\ x2 = v1 - v2\ i \end{cases}$$

Cuatro valores reales:
Parte entera y parte real
de cada una de las soluciones

Ecuación de Segundo Grado

Diseño

Entrada: $a, b, c \in \mathcal{R}, \quad a \neq 0$

Salida:

Algoritmo:

Variables:

Ecuación de Segundo Grado

Diseño

Entrada: $a, b, c \in \mathcal{R}, \quad a \neq 0$

Salida: v1, v2 $\in \mathcal{R}$

Algoritmo:

Variables:

Ecuación de Segundo Grado

Diseño

Entrada: $a, b, c \in \mathcal{R}, \quad a \neq 0$

Salida: v1, v2 $\in \mathcal{R}$

Algoritmo:

Obtener(a, b, c)

Variables:

Ecuación de Segundo Grado

Diseño

Entrada: $a, b, c \in \mathcal{R}, \quad a \neq 0$

Salida: v1, v2 $\in \mathcal{R}$

Algoritmo:

Obtener(a, b, c)

$d \leftarrow b * b - 4 * a * c$

Variables:

Ecuación de Segundo Grado

Diseño

Entrada: $a, b, c \in \mathcal{R}, \quad a \neq 0$

Salida: v1, v2 $\in \mathcal{R}$

Algoritmo:

Obtener(a, b, c)

$d \leftarrow b * b - 4 * a * c$

$v1 \leftarrow -b/(2 * a)$

$v2 \leftarrow \sqrt{|d|}/(2 * a)$

Variables:

Ecuación de Segundo Grado

Diseño

Entrada: $a, b, c \in \mathcal{R}, \quad a \neq 0$

Salida: v1, v2 $\in \mathcal{R}$

Algoritmo:

Obtener(a, b, c)

$d \leftarrow b * b - 4 * a * c$

$v1 \leftarrow -b/(2 * a)$

$v2 \leftarrow \sqrt{|d|}/(2 * a)$

Devolver$(v1 + ord(d \geq 0) * v2, ord(d < 0) * v2,$
$v1 - ord(d \geq 0) * v2, -ord(d < 0) * v2)$

Variables:

Ecuación de Segundo Grado

Diseño

Entrada: $a, b, c \in \mathcal{R}, \quad a \neq 0$

Salida: v1, v2 $\in \mathcal{R}$

Algoritmo:

Obtener(a, b, c)

$d \leftarrow b * b - 4 * a * c$

$v1 \leftarrow -b/(2 * a)$

$v2 \leftarrow \sqrt{|d|}/(2 * a)$

Devolver$(v1 + ord(d \geq 0) * v2, ord(d < 0) * v2,$
$v1 - ord(d \geq 0) * v2, -ord(d < 0) * v2)$

Variables: d $\in \mathcal{R}$

Ecuación de Segundo Grado

ec2grado.pas

```
Program ec2grado;
Uses Crt;
Var
   a,b,c,d,v1,v2: Real;
Begin
   Read(a,b,c);
   d:= b*b - 4 *a * c;
   v1:= -b/(2*a);
   v2:= sqrt(abs(d))/(2*a);
   Write(v1 + ord(d>=0)*v2, ord(d<0)*v2,
         v1 - ord(d>=0)*v2, -ord(d<0)*v2);
End.
```

Ecuación de Segundo Grado

ec2grado.m

```
% Programa ec2grado;
   a = input('a = ');
   b = input('b = ');
   c = input('c = ');
   d = b*b - 4 *a * c;
   v1 = -b/(2*a);
   v2 = sqrt(abs(d))/(2*a);
   fprintf('%0.2f %0.2f %0.2f %0.2f',
         v1 +(d>=0)*v2,(d<0)*v2,
         v1 -(d>=0)*v2, -(d<0)*v2);
```

Selección simple

Pseudocódigo

Si Condición *Entonces*
    Bloque de Instrucciones

Código Pascal

```
If Condición Then
   Instrucción;
```

```
If Condición Then
Begin
   Instrucción 1;
   ...
End;
```

Código Matlab

```
if Condición
   Instrucción;
end;
```

```
if Condición
   Instrucción 1;
   ...
end;
```

Selección simple

Pseudocódigo

Si Condición *Entonces*
    Bloque de Instrucciones 1
Sino
    Bloque de Instrucciones 2

Código Pascal

```
If Condición Then
    Bloque de Instrucciones 1
Else
    Bloque de Instrucciones 2;
```

Código Matlab

```
if Condición
    Bloque de Instrucciones 1;
else
    Bloque de Instrucciones 2;
end;
```

Selección múltiple

Pseudocódigo

Segun Valor Ordinal
rango1 : Bloque Instrucciones 1
...
Sino
Bloque Instrucciones n

Código Pascal

Case *Valor Ordinal* of
rango1: *Bloque Instrucc 1*;
...
Else
Bloque Instrucc n;

Código Matlab

switch *Valor Ordinal*
case *rango1*
Bloque Instrucc 1; ...
otherwise
Bloque Instrucc n;
end

Ecuación de Segundo Grado

Diseño

Entrada: $a, b, c \in \mathcal{R}, \quad a \neq 0$

Salida:

Algoritmo:

Obtener(a, b, c)

$d \leftarrow b * b - 4 * a * c$

$v1 \leftarrow -b/(2 * a)$

$v2 \leftarrow \sqrt{|d|}/(2 * a)$

Variables: d $\in \mathcal{R}$

Ecuación de Segundo Grado

Diseño

Entrada: $a, b, c \in \mathcal{R}, \quad a \neq 0$
Salida: v1, v2 $\in \mathcal{R}$
Algoritmo:

Obtener(a, b, c)
$d \leftarrow b * b - 4 * a * c$
$v1 \leftarrow -b/(2 * a)$
$v2 \leftarrow \sqrt{|d|}/(2 * a)$
Si $(d >= 0)$ Entonces
Devolver$(v1 + v2, v1 - v2)$

Variables: d $\in \mathcal{R}$

Ecuación de Segundo Grado

Diseño

Entrada: $a, b, c \in \mathcal{R}, \quad a \neq 0$
Salida: v1, v2 $\in \mathcal{R}$
Algoritmo:

Obtener(a, b, c)
$d \leftarrow b * b - 4 * a * c$
$v1 \leftarrow -b/(2 * a)$
$v2 \leftarrow \sqrt{|d|}/(2 * a)$
Si $(d >= 0)$ Entonces
  Devolver$(v1 + v2, v1 - v2)$
Sino
  Devolver$(v1, v2, v1, -v2)$

Variables: d $\in \mathcal{R}$

Ecuación de Segundo Grado

ec2if.pas

```
Program ec2if;
Uses Crt;
Var
   a,b,c,d,v1,v2: Real;
Begin
   Read(a,b,c);
   d:= b*b - 4 *a * c;
   v1:= -b/(2*a);
   v2:= sqrt(abs(d))/(2*a);
   If (d>=0) Then
      Write(v1+v2, v1-v2);
   Else
      Write(v1, v2, v1, -v2);
End.
```

Ecuación de Segundo Grado

ec2if.m

```
%Programa ec2if;
   a = input('a = ');
   b = input('b = ');
   c = input('c = ');
   d = b*b - 4 *a * c;
   v1 = -b/(2*a);
   v2 = sqrt(abs(d))/(2*a);
   if (d>=0)
      fprintf('%0.2f %0.2f',v1+v2, v1-v2);
   else
      fprintf('%0.2f %0.2f %0.2f %0.2f',v1, v2, v1, -v2);
   end;
```

Transformar Notas

Planteamiento

Dado un valor numérico para una nota, encontrar la letra correspondiente según el código

Letra	Rango	Condición
A	[19,20]	Sobresaliente
B	[16,18]	Eximido
C	[10,15]	Aprobado
D	[5,9]	Reprobado
E	[1,4]	Deficiente
X		No Valida

Transformar Notas

Diseño

Entrada: $nota \in \mathcal{Z}$
Salida:
Algoritmo:

Variables:

Transformar Notas

Diseño

Entrada: $nota \in \mathcal{Z}$
Salida: $letra \in ch$
Algoritmo:

Variables:

Transformar Notas

Diseño

Entrada: $nota \in \mathcal{Z}$
Salida: $letra \in ch$
Algoritmo:

Obtener($nota$)
Según $nota$
$19,20 : letra \leftarrow \ 'A'$
$16 .. 18 : letra \leftarrow \ 'B'$
$10 .. 15 : letra \leftarrow \ 'C'$
$5 .. 9 : letra \leftarrow \ 'D'$
$1 .. 4 : letra \leftarrow \ 'E'$
Sino
$letra \leftarrow' X'$
Devolver($letra$)

Variables:

Transformar Notas

Diseño

Entrada: $nota \in \mathcal{Z}$
Salida: $letra \in ch$, $letra \in \{A, B, C, D, E, X\}$
Algoritmo:

Obtener($nota$)
Según $nota$
$19,20 : letra \leftarrow 'A'$
$16 .. 18 : letra \leftarrow 'B'$
$10 .. 15 : letra \leftarrow 'C'$
$5 .. 9 : letra \leftarrow 'D'$
$1 .. 4 : letra \leftarrow 'E'$
Sino
$letra \leftarrow' X'$
Devolver($letra$)

Variables:

Transformar Notas

notas.pas

```
Program notas;
Uses Crt;
Var
   nota: Byte;
   letra: Char;
Begin
   Read(nota);
   Case (nota) of
      19,20 :  letra:= 'A';
      16..18 :  letra:= 'B';
      10..15 :  letra:= 'C';
      5..9 :  letra:= 'D';
      1..4:  letra:= 'E'
    Else
       letra:='X';
    Write(letra);
End.
```

Transformar Notas

notas.m

```
%Programa notas;
   nota = input('nota = ');
   switch (nota)
      case 19,20
           letra = 'A';
      case 16,17,18
           letra = 'B';
      case 10,11,12,13,14,15
           letra = 'C';
      case 5,6,7,8,9
           letra = 'D';
      case 1,2,3,4
           letra = 'E'
   otherwise
      letra ='X';
   end;
   disp(letra);
```

Repita para ...un número determinado de veces

Hace uso de una variable auxiliar que sirve de contador de ciclos de repetición, la cual puede ir en ascenso o descenso

Pseudocódigo

Para *IdVariable* ← *Vi* .. *Vf*
Bloque de Instrucciones

Código Pascal

For *IdVariable* := *Vi* To *Vf* Do
Bloque de Instrucciones;

For *IdVariable* := *Vi* Downto *Vf* Do
Bloque de Instrucciones;

Código Matlab

for *IdVariable* = *Vi* : *Vf*
Bloque de Instrucciones;

---

for *IdVariable* = *Vi* :-1: *Vf*
Bloque de Instrucciones;

Repita mientras ... solo si es necesario

El ciclo del lazo de repetición se inicia si la condición se cumple y se detiene cuando deja de cumplirse.

Pseudocódigo

Mientras Condición
    Bloque de Instrucciones

Código Pascal

```
While Condición Do Begin
    Bloque de Instrucciones
End;
```

Código Matlab

```
while Condición
    Bloque de Instrucciones
end;
```

Si se necesita un contador de repeticiones éste debe controlarse en el bloque de instrucciones

Repita hasta ... al menos una vez

El ciclo del lazo de repetición se detiene si la condición se cumple.

Pseudocódigo

Repita
    Bloque de Instrucciones
Hasta Condición

Código Pascal

Repeat
    Bloque de Instrucciones
Until Condición;

Si se necesita un contador de repeticiones éste debe controlarse en el bloque de instrucciones

Cálculo de promedio de notas

Calcular el promedio de n valores aleatorios correspondientes a las notas de un curso

$$\bar{x} = \sum_{i=1}^{n} \frac{x_i}{n} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

Diseño

Entrada:

Salida:

Proceso:

Cálculo de promedio de notas

Calcular el promedio de n valores aleatorios correspondientes a las notas de un curso

$$\bar{x} = \sum_{i=1}^{n} \frac{x_i}{n} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

Diseño

Entrada: $n \in \mathcal{Z},\ n > 0$

Salida: $prom \in \mathcal{R}$

Proceso:

$$prom = \frac{1}{n} \sum_{i=1}^{n} x_i$$

Cálculo de promedio de notas

Diseño

Algoritmo:

Obtener(n)
prom $\leftarrow 0$
Para $i \leftarrow 1..n$
  $x \leftarrow$ *Aleatorio*(21)
  $prom \leftarrow prom + x$
$prom \leftarrow prom/n$
Devolver(prom)

Variables: $x \in \mathcal{R}$, $x \in [0, 20]$
  $i \in \mathcal{Z}$

Cálculo de promedio de notas

promnotas.pas

```
Program promnotas;
Uses Crt;
Var
   n,i,x : Integer;
   prom : Real;
Begin
   Readln(n);
   prom:=0;
   For i:=1 to n do
   Begin
      x:=random(21);
      prom:=prom +x;
   End;
   prom:=prom/n;
   writeln(prom);
End.
```

Cálculo de promedio de notas

promnotas.m

```
%Programa promnotas;
    n = input('n = ');
    prom = 0;
    for i=1:n
       x = randi(21)-1
       prom = prom + x;
    end;
    prom = prom/n;
    disp(prom);
```

Cálculo de producto de impares

Calcule el producto de los n primeros números impares.

$$1 \times 3 \times 5 \times 7 \cdots$$

$$p = \prod_{i=1}^{n} 2i - 1$$

Diseño

Entrada: $n \in \mathcal{Z}$

Salida: $p \in \mathcal{Z}$

Algoritmo:

Obtener(n)
$p \leftarrow 1$
Para $i \leftarrow 1..n$
$p \leftarrow p * (2 * i - 1)$
Devolver(p)

Variables: $i \in \mathcal{Z}$

Cálculo de producto de impares

prodimpar.pas

```
Program prodimpar;
Uses Crt;
Var
   n,p,i : Integer;
Begin
   Readln(n);
   p:=1;
   For i:=1 to n do
      p:=p*(2*i-1);
   writeln(p);
End.
```

Cálculo de producto de impares

prodimpar.m

```
%Programa prodimpar;
   n = input('n = ');
   p = 1;
   for i=1:n
      p= p*(2*i-1);
   end;
   disp(p);
```

Potencia de números enteros

Dados dos números enteros a, b calcular a^b

$$2^5 = 2 \times 2 \times 2 \times 2 \times 2 \qquad 5^{-2} = \frac{1}{5^2} = \frac{1}{5 \times 5}$$
$$-2^3 = -2 \times -2 \times -2 \qquad -3^{-2} = \frac{1}{-3^2} = \frac{1}{-3 \times -3}$$

Diseño

Entrada:
Salida:
Proceso:

Potencia de números enteros

Dados dos números enteros a, b calcular a^b

$$2^5 = 2 \times 2 \times 2 \times 2 \times 2 \qquad 5^{-2} = \frac{1}{5^2} = \frac{1}{5 \times 5}$$
$$-2^3 = -2 \times -2 \times -2 \qquad -3^{-2} = \frac{1}{-3^2} = \frac{1}{-3 \times -3}$$

Diseño

Entrada: $a, b \in \mathcal{Z}$
Salida: $p \in \mathcal{R}$
Proceso:

$$p = \prod_{i=1}^{|b|} a, \qquad \begin{cases} 1 & b = 0 \\ \frac{1}{p}, & b < 0, \quad a \neq 0 \end{cases}$$

Potencia de números enteros

Diseño

Algoritmo:

$Obtener(a, b)$
$p \leftarrow 1$
$Para\ i \leftarrow 1..|b|$
    $p \leftarrow p * a$
$Si\ ((b < 0) \wedge (a \neq 0))$
    $p \leftarrow 1/p$
$Devolver(p)$

Variables: $i \in \mathcal{Z}$

Potencia de números enteros

potencia2.pas

```
Program potencia2;
Uses Crt;
Var
   a,b,i : Integer;
   p : real;
Begin
   Readln(a,b);
   p:=1;
   For i:=1 to abs(b) do
      p:= p * a;
   If ((b<0) and (a<>0)) Then
      p:= 1/p;
   writeln(p);
End.
```

Potencia de números enteros

potencia2.m

```
%Programa potencia2;
   a = input('a = ');
   b = input('b = ');
   p = 1;
   for i = 1:abs(b)
      p = p * a;
   end;
   if ((b<0) && (a~=0))
      p = 1/p;
   end;
   disp(p);
```

Cálculo de la arco-tangente

Calcular los n primeros términos de la aproximación

$$tan^{-1}(x) = \begin{cases} x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} \cdots, & |x| < 1 \\ \pm\frac{\pi}{2} - \frac{1}{x} + \frac{1}{3x^3} - \frac{1}{5x^5} \cdots, & \begin{cases} + : x \geq 1 \\ - : x \leq -1 \end{cases} \end{cases}$$

Análisis

$$tan^{-1}(x) = \begin{cases} x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \frac{x^7x^2}{7+2} - \frac{x^9x^2}{9+2} \cdots, & |x| < 1 \\ \pm\frac{\pi}{2} - \frac{1}{x} + \frac{1}{3x^3} - \frac{1}{(3+2)x^3x^2} + \frac{1}{(5+2)x^5x^2} \cdots, & \begin{cases} + : x \geq 1 \\ - : x \leq -1 \end{cases} \end{cases}$$

Cálculo de la arco-tangente

Diseño

Entrada: $x \in \mathcal{R},\ n \in \mathcal{Z},\ n > 0$

Salida: $atan \in \mathcal{R}$

Procedimiento:

$$atan = \begin{cases} \sum_{i=1}^{n} \frac{n_i}{d_i}, & |x| < 1 \quad \begin{cases} n_1 = x \\ n_i = -n_{i-1}x^2 \\ d_1 = 1 \\ d_i = d_{i-1} + 2 \end{cases} \\ \pm\frac{\pi}{2} + \sum_{i=2}^{n} \frac{n_i}{d1_i d2_i}, & \begin{cases} + : x \geq 1 \\ - : x \leq -1 \end{cases} \quad \begin{cases} n_1 = -1 \\ n_i = -n_{i_1} \\ d1_1 = 1 \\ d1_i = d1_{i-1} + 2 \\ d2_1 = x \\ d2_i = d2_{i-1}x^2 \end{cases} \end{cases}$$

Cálculo de la arco-tangente

Algoritmo

$Obtener(n, x)$
$atan \leftarrow 0$
$Si(|x| < 1)$
  $n1 \leftarrow x$
  $d1 \leftarrow 1$
  $Para\ i \leftarrow 1..n$
    $atan \leftarrow atan + n1/d1$
    $n1 \leftarrow -n1 * x * x$
    $d1 \leftarrow d1 + 2$
$Sino$
  $n1 \leftarrow -1$
  $d11 \leftarrow 1$
  $d21 \leftarrow x$
  $Para\ i \leftarrow 2..n$
    $atan \leftarrow atan + n1/(d11 * d21)$
    $n1 \leftarrow -n1$
    $d11 \leftarrow d11 + 2$
    $d21 \leftarrow d21 * x * x$
  $Si(x \geq 1)$
    $atan \leftarrow atan + pi/2$
  $Sino$
    $atan \leftarrow atan - pi/2$
$Devolver(atan)$
Variables: $i \in \mathcal{Z}$
$n1, d1, d11, d21 \in \mathcal{R}$

Cálculo de Area

Calcular usando aproximación de los trapecios y de simpson para el área bajo $f(x) = \frac{1}{1-x}$ dividiendo el intervalo $[a, b]$ en n segmentos.

$$\int_a^b f(x)dx \simeq \sum_{i=0}^{n-1} \frac{b-a}{2n} [f(x_i) + f(x_{i+1})]$$

$$\int_a^b f(x)dx \simeq \sum_{i=0}^{n-1} \frac{b-a}{6n} [f(x_i) + 4f(x_m) + f(x_{i+1})]$$

Cálculo de Area

Diseño

Entrada: $a, b \in \mathcal{R} \quad n \in \mathcal{Z},\ a < b,\ n > 1,\ ((b < 1) \vee (a > 1))$

Salida: $at, as \in \mathcal{R}$

Procedimiento:

$$at = \frac{b-a}{2n} \sum_{i=0}^{n-1} \left[\frac{1}{1-x_i} + \frac{1}{1-x_{i+1}} \right]$$

$$as = \frac{b-a}{6n} \sum_{i=0}^{n-1} \left[\frac{1}{1-x_i} + \frac{4}{1-x_m} + \frac{1}{1-x_{i+1}} \right]$$

$$x_0 = a \qquad x_i = x_{i-1} + \frac{b-a}{n} \qquad x_m = \frac{x_i + x_{i+1}}{2} \qquad i \in [1, n]$$

Cubos

Método de Nicómano

Escriba un algoritmo que calcule los primeros n cubos mediante la conjetura de Nicómaco

- El primer impar: 1·
- Sume los dos siguientes impares: $3 + 5 = 8$.
- Sume los tres siguientes impares: $7 + 9 + 11 = 27$.

Análisis

$$k_1 = 1 \qquad k_2 = k_1 + 2 \qquad \cdots$$

$$cubo_1 = k_1 \qquad cubo_i = \sum_{j=1}^{i} k_j, \qquad i \in [2, n]$$

Cubos

Método de Nicómano

Diseño

Entrada: $n \in \mathcal{Z}, \quad n > 0$

Salida: $cubo \in \mathcal{Z}$

Procedimiento:

$k = 1$

$cubo_1 = k$

$$\begin{cases} k = k + 2 \\ cubo_i = \sum_{j=1}^{i} k \quad i \in [2, n] \end{cases}$$

Algoritmo:

Obtener(n)

$k \leftarrow 1$

Devolver(k)

Para $i \leftarrow 2..n$

$cubo \leftarrow 0$

Para $j \leftarrow 1..i$

$k \leftarrow k + 2$

$cubo \leftarrow cubo + k$

Devolver$(cubo)$

Variables: $k, i, j \in \mathcal{Z}$

Subprogramas

Un subprograma es un programa que resuelve un problema en particular y que puede ser llamado desde otro programa.
Como todo programa, un subprograma tiene variables de entrada, variables de salida, variables internas y un cuerpo conformado por el conjunto de instrucciones que permiten ejecutar las tareas para las que fue diseñado.
Para cada subprograma debemos hacer un diseño algorítmico.
Los subprogramas pueden ser Funciones o Procedimientos.
En pascal los subprogramas se incluyen en la sección de Declaraciones: antes del Begin que indica el Inicio del Programa principal.

Funciones — Subprograma que retorna un valor único

Pseudocódigo

s. IdFunción (ListaVariablesEntrada) ∈ Tipo
Bloque de Instrucciones
V: ListaVariablesInternas

Código Pascal

```
Function IdFunción ( ListaVariablesEntrada ): Tipo
Var ListaVariablesInternas;
Begin
   Bloque de Instrucciones;
End;
```

Procedimientos

Subprograma que puede o no retornar valores

Pseudocódigo

Procedimiento IdProced (ListaVarEntrada, ListaVarSalida)
  Bloque de Instrucciones
V: ListaVarInternas

Código Pascal

Procedure IdProced (ListaVarEntrada, *var* ListaVarSalida)
Var ListaVarInternas;
Begin
  Bloque de Instrucciones;
End;

Datos Simples

Enumerados

Permite construir variables ordinales donde los valores asociados están declarados en orden.

Pseudocódigo

IdVariable ∈ (*ListaValores*)

Código Pascal

IdVariable : (*ListaValores*);

Ejemplo

Laborables ∈ (*Lunes*, *Martes*, *Miercoles*, *Jueves*, *Viernes*)

Datos Simples

Rangos

Variable ordinal donde los valores asociados están ordenados y pertenecen a un conjunto de datos definidos previamente.

Pseudocódigo

IdVariable $\in$ *v0..vf*

Código Pascal

IdVariable: *v0..vf* ;

Ejemplo

Nota $\in$ *0..20*

Tipos Definidos por el Usuario

En Pascal se ubican en la sección de Declaraciones antes de la declaración de Variables.

Pseudocódigo

IdTipo $\in$ *DescTipoConstruido*

Código Pascal

Type
IdTipo= *DescTipoConstruido* ;

Datos Compuestos

Vectores

Tipo de dato Compuesto de múltiples valores del mismo tipo.
Los valores están indixados según un rango predefinido.

Pseudocódigo

IdVariable ∈ *Arreglo[Vo..Vf]* ∈ *Tipo*

Código Pascal

IdVariable : Array[Vo..Vf] Of Tipo;

Datos Compuestos

Cadenas

Arreglo de Caracteres de longitud definida.

Pseudocódigo

IdVariable $\in \mathcal{C}$*[long]*

Código Pascal

IdVariable : String[long];

Datos Compuestos

Matrices

Arreglo de dos dimensiones.
En el caso más general podemos escribir arreglos multidimensionales.

Pseudocódigo

$$IdVariable \in Arreglo[V1o..V1f,V2o..V2f\ ] \in Tipo$$

Código Pascal

$$IdVariable : Array[V1o..V1f,V2o..v2f]\ Of\ Tipo;$$

Datos Simples

Conjuntos

Variable ordinal donde los valores asociados están ordenados y pertenecen a un conjunto de datos definidos previamente.

Pseudocódigo

$IdVariable \in \{v0..vf\}$

Código Pascal

IdVariable: Set Of v0..vf ;

Ejemplo

$Mes \in \{1..12\}$

Datos Compuestos — Registros

Tipo de Dato construido a partir de una lista de Variables.

Pseudocódigo

$IdTipo \in \{Bloque\ de\ Declaraciones\ de\ Variables\}$

Código Pascal

```
IdTipo = Record
        Bloque de Declaraciones de Variables
end;
```

Ejemplo

$$Alumno \in \{ \quad nombre \in \mathcal{C} \quad edad \in \mathcal{Z} \ \}$$

de control

00	NULL	(carácter nulo)
01	SOH	(inicio encabezado)
02	STX	(inicio texto)
03	ETX	(fin de texto)
04	EOT	(fin transmisión)
05	ENQ	(consulta)
06	ACK	(reconocimiento)
07	BEL	(timbre)
08	BS	(retroceso)
09	HT	(tab horizontal)
10	LF	(nueva línea)
11	VT	(tab vertical)
12	FF	(nueva página)
13	CR	(retorno de carro)
14	SO	(desplaza afuera)
15	SI	(desplaza adentro)
16	DLE	(esc.vínculo datos)
17	DC1	(control disp. 1)
18	DC2	(control disp. 2)
19	DC3	(control disp. 3)
20	DC4	(control disp. 4)
21	NAK	(conf. negativa)
22	SYN	(inactividad sínc)
23	ETB	(fin bloque trans)
24	CAN	(cancelar)
25	EM	(fin del medio)
26	SUB	(sustitución)
27	ESC	(escape)
28	FS	(sep. archivos)
29	GS	(sep. grupos)
30	RS	(sep. registros)
31	US	(sep. unidades)
127	DEL	(suprimir)

imprimibles

32	espacio	64	@	96	`
33	!	65	A	97	a
34	"	66	B	98	b
35	#	67	C	99	c
36	$	68	D	100	d
37	%	69	E	101	e
38	&	70	F	102	f
39	'	71	G	103	g
40	(	72	H	104	h
41	)	73	I	105	i
42	*	74	J	106	j
43	+	75	K	107	k
44	,	76	L	108	l
45	-	77	M	109	m
46	.	78	N	110	n
47	/	79	O	111	o
48	0	80	P	112	p
49	1	81	Q	113	q
50	2	82	R	114	r
51	3	83	S	115	s
52	4	84	T	116	t
53	5	85	U	117	u
54	6	86	V	118	v
55	7	87	W	119	w
56	8	88	X	120	x
57	9	89	Y	121	y
58	:	90	Z	122	z
59	;	91	[	123	{
60	<	92	\	124	|
61	=	93	]	125	}
62	>	94	^	126	~
63	?	95	_		

ASCII extendido

128	Ç	160	á	192	└	224	Ó
129	ü	161	í	193	┴	225	ß
130	é	162	ó	194	┬	226	Ô
131	â	163	ú	195	├	227	Ò
132	ä	164	ñ	196	─	228	õ
133	à	165	Ñ	197	┼	229	Õ
134	å	166	ª	198	ã	230	µ
135	ç	167	º	199	Ã	231	þ
136	ê	168	¿	200	╚	232	Þ
137	ë	169	®	201	╔	233	Ú
138	è	170	¬	202	╩	234	Û
139	ï	171	½	203	╦	235	Ù
140	î	172	¼	204	╠	236	ý
141	ì	173	¡	205	═	237	Ý
142	Ä	174	«	206	╬	238	¯
143	Å	175	»	207	¤	239	´
144	É	176	░	208	ð	240	≡
145	æ	177	▒	209	Ð	241	±
146	Æ	178	▓	210	Ê	242	‗
147	ô	179	│	211	Ë	243	¾
148	ö	180	┤	212	È	244	¶
149	ò	181	Á	213	ı	245	§
150	û	182	Â	214	Í	246	÷
151	ù	183	À	215	Î	247	¸
152	ÿ	184	©	216	Ï	248	°
153	Ö	185	╣	217	┘	249	¨
154	Ü	186	║	218	┌	250	·
155	ø	187	╗	219	█	251	¹
156	£	188	╝	220	▄	252	³
157	Ø	189	¢	221	¦	253	²
158	×	190	¥	222	Ì	254	■
159	ƒ	191	┐	223	▀	255	nbsp

Potencias:

$x^{-y} \equiv \frac{1}{x^y}$

$x^{yz} \equiv (x^y)^z$

$x^{y+z} \equiv x^y x^z$

$x^y \equiv e^{(y*ln(x))}, x > 0$

Logarítmicas:

$ln(x) \equiv \log_e(x),\ x > 0$

$b^{\log_b(x)} \equiv \log_b(b^x) \equiv x,\ x > 0$

$\log_b(xy) \equiv \log_b(x) + \log_b(y)\ x, y > 0$

$\log_b(x) \equiv \frac{\log_a(x)}{\log_a(b)},\ x, b > 0$

$\log_b(x^y) \equiv y \log_b(x),\ x > 0$

$\log_b(x/y) \equiv \log_b(x) - \log_b(y)\ x, y > 0$

Trigonométricas:

$sin^2(x) + cos^2(x) \equiv 1$

$sin(x \pm y) \equiv sin(x)cos(y) \pm cos(x)sin(y)$

$cos(x \pm y) \equiv cos(x)cos(y) \mp sin(x)sin(y)$

$tan(x) \equiv \frac{sin(x)}{cos(x)},\ x \neq (2n+1)\pi/2,\ n \in \mathcal{Z}$

$sec(x) \equiv \frac{1}{cos(x)},\ x \neq (2n+1)\pi/2,\ n \in \mathcal{Z}$

$csc(x) \equiv \frac{1}{sin(x)},\ x \neq n\pi,\ n \in \mathcal{Z}$

$cosh^2(x) - sinh^2(x) \equiv 1$

$sinh(x) \equiv \frac{1}{2}(e^x - e^{-x})$

$cosh(x) \equiv \frac{1}{2}(e^x + e^{-x})$

$tanh(x) \equiv \frac{sinh(x)}{cosh(x)},\ x \neq 0$

$sin^{-1}(x) + cos^{-1}(x) \equiv \pi/2$

$sin^{-1}(x) \equiv tan^{-1}\left(x/\sqrt{1-x^2}\right),\ |x| < 1$

$cos^{-1}(x) \equiv tan^{-1}\left(\sqrt{1-x^2}/x\right),\ |x| < 1$

$ctan^{-1}(x) \equiv tan^{-1}(1/x),\ x \neq$

$sec^{-1}(x) \equiv cos^{-1}(1/x),\ |x| \geq 1,\ x \neq 0$

$csc^{-1}(x) \equiv sin^{-1}(1/x),\ |x| \geq 1,\ x \neq 0$

$csch(x) \equiv \frac{1}{sinh(x)},\ x \neq 0$

$sech(x) \equiv \frac{1}{cosh(x)}$

$ctanh(x) \equiv \frac{cosh(x)}{sinh(x)}$

Referencias


[JL03] Luis Joyanes Aguilar.
Fundamentos de Programación: Algoritmos Estructuras de datos y Objetos, McGraw Hill. Tercera Edición. 2003


[SW93] William I. Salmon
Introducción a la computación con Turbo Pascal. Estructuras y Abstracciones. Addison-Wesley Iberoamericana. 1993


[KW] John Konvalina, Stanley Wileman
Programación con Pascal. Mc Graw - Hill


[GP] Peter Grogono
Programación en Pascal. Addison - Wesley