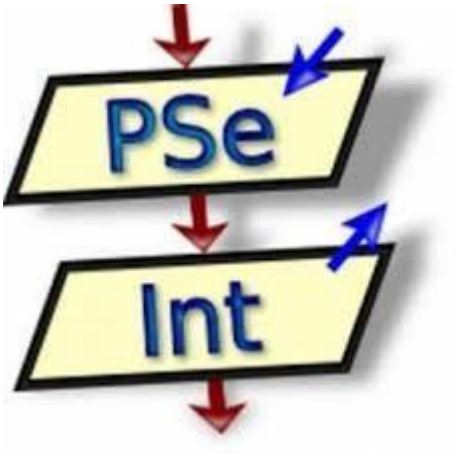
Introducción a la Programación con PSeInt



2024

Programación con PSeInt

Programa

Un programa no es más que un algoritmo escrito en algún lenguaje de programación de computadoras. Un programa es, por lo tanto, un conjunto de instrucciones —órdenes dadas a la computadora— que producirán la ejecución de una determinada tarea. En esencia, un programa es un medio para conseguir un fin. El fin será probablemente definido como la información necesaria para solucionar un problema.

Pasos para la Construcción de un Programa

El proceso de programación es un proceso de solución de problemas en el cual deben llevarse a cabo los pasos descritos a continuación.

1- Definición del problema

En este paso se determina la información inicial para la elaboración del programa. Es donde se determina que es lo que debe resolverse con el computador, el cual requiere una definición clara y precisa.
Es importante que se conozca lo que se desea que realice la computadora; mientras la definición del problema no se conozca del todo, no tiene mucho caso continuar con la siguiente etapa.

2- Análisis del problema

Una vez que se ha comprendido lo que se desea de la computadora, es necesario definir:

- ✔ Los datos de entrada.
- ✔ Los datos de salida
- ✔ Los métodos y fórmulas que se necesitan para procesar los datos.

Una recomendación muy practica es la de colocarse en el lugar de la computadora y analizar qué es lo que se necesita que se ordene y en que secuencia para producir los resultados esperados.

3- Diseño del algoritmo

Se puede utilizar algunas de las herramientas de representación de algoritmos para definir la secuencia de pasos que se deben llevar a cabo para conseguir la salida identificada en el paso anterior. La herramienta de representación de algoritmos que utilizaremos es el pseudocódigo.

El *pseudocódigo* es una herramienta de programación en la que las instrucciones se escriben en palabras similares al inglés o español, que facilitan tanto la escritura como la lectura de programas. En esencia, el pseudocódigo se puede definir como un lenguaje de especificaciones de algoritmos. El uso de tal lenguaje

hace el paso de codificación final (esto es, la traducción a un lenguaje de programación) relativamente fácil. El pseudocódigo se considera un primer borrador, dado que tiene que traducirse posteriormente a un lenguaje de programación.

4- Codificación

La codificación es la operación de escribir la solución del problema (de acuerdo a la lógica del pseudocódigo), en una serie de instrucciones detalladas, en un código reconocible por la computadora. La serie de instrucciones detalladas se conoce como *código fuente*, el cual se escribe en un lenguaje de programación o lenguaje de alto nivel.

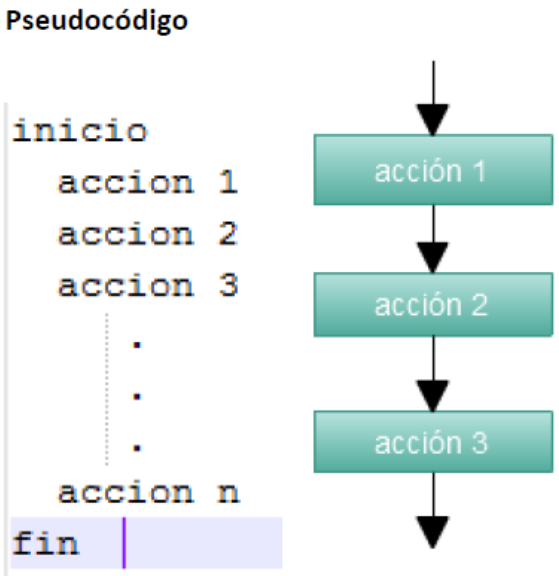
5- Prueba y Depuración

Se denomina prueba de escritorio a la comprobación que se hace de un algoritmo para saber si está bien realizado. Esta prueba consiste en tomar datos específicos como entrada y seguir la secuencia indicada en el algoritmo hasta obtener un resultado, el análisis de estos resultados indicara si el algoritmo esta correcto o si por el contrario hay necesidad de corregirlo o hacerle ajustes.

Partes Constitutivas de un Programa

Tras la decisión de desarrollar un programa, el programador debe establecer el conjunto de especificaciones que debe contener el programa: entrada, salida y algoritmos de resolución, que incluirán las técnicas para obtener las salidas a partir de las entradas.

Un programa puede ser lineal (secuencial) o no lineal. Un programa es lineal si las instrucciones (acciones) se ejecutan secuencialmente, es decir, sin bifurcaciones, decisiones ni comparaciones.



Estructura de un programa secuencial

1- Tipos de Instrucciones

Las instrucciones —acciones— básicas que se pueden implementar de modo general en un algoritmo y que esencialmente soportan todos los lenguajes son las siguientes:

✔ *Instrucciones de inicio/fin*, son utilizadas para delimitar bloques de código.

✔ *Instrucciones de asignación*, se utilizan para asignar el resultado de la evaluación de una expresión a una variable. El valor (dato) que se obtiene al evaluar la expresión es almacenado en la variable que se indique a la izquierda de la expresión de asignación:

`<nombre de la variable> ← <expresión>`

donde, expresión es igual a una expresión matemática o lógica.

✔ *Instrucciones de lectura*, se utilizan para leer datos de un dispositivo de entrada y se asignan a las variables. En PSeInt esta instrucción se expresa con la palabra `Leer`.

✔ *Instrucciones de escritura*, se utilizan para escribir o mostrar mensajes o contenidos de las variables en un dispositivo de salida. En PSeInt esta instrucción se expresa con la palabra `escribir` ó `imprimir`.

✔ *Instrucciones de flujo de control, o de bifurcación y salto*, estas instrucciones de flujo de control son aquellas instrucciones que alteran el orden secuencial de la ejecución de un programa.

2- Identificadores

Un identificador es un conjunto de caracteres alfanuméricos de cualquier longitud que sirve para identificar las entidades del programa (nombre del programa, nombres de variables, constantes, subprogramas, etc). En PseInt los identificadores deben constar sólo de letras, números y/o guión_bajo (_), comenzando siempre con una letra.

3- Variables y Constantes

Los programas de computadora contienen ciertos valores que no deben cambiar durante la ejecución del programa. Tales valores se llaman constantes. De igual forma, existen otros valores que cambiarán durante la ejecución del programa; a estos valores se les llama variables. Una variable es un objeto o tipo de datos cuyo valor puede cambiar durante el desarrollo del algoritmo o ejecución del programa.

Dependiendo del lenguaje, hay diferentes tipos de variables, tales como enteras, reales, carácter, lógicas y de cadena. Una variable que es de un cierto tipo puede tomar únicamente valores de ese tipo. Una variable de carácter, por ejemplo, puede tomar como valor sólo caracteres, mientras que una variable entera puede tomar sólo valores enteros.

4- Declaración de Variables

Normalmente los identificadores de las variables y de las constantes con nombre deben ser declaradas en los programas antes de ser utilizadas.

5- Entrada y Salida de Información

Los cálculos que realizan las computadoras requieren para ser útiles la entrada de los datos necesarios para ejecutar las operaciones que posteriormente se convertirán en resultados, es decir, salida.
Las *operaciones de entrada* permiten leer determinados valores y asignarlos a determinadas variables. Esta entrada se conoce como operación de lectura (leer). Los datos de entrada se introducen al procesador mediante dispositivos de entrada (teclado, tarjetas perforadas, unidades de disco, etc.). La salida puede aparecer en un dispositivo de salida (pantalla, impresora, etc.). La *operación de salida* se denomina escritura (escribir). En la escritura de algoritmos las acciones de lectura y escritura se representan por los siguientes formatos:

```
leer (lista de variables de entrada)
escribir (lista de variables de salida)
```

Escritura de Algoritmos/Programas

Un algoritmo constará de dos componentes: una cabecera de programa y un bloque algoritmo. La cabecera de programa es una acción simple que comienza con la palabra algoritmo. Esta palabra estará seguida por el nombre asignado al programa completo. El bloque algoritmo es el resto del programa y consta de dos componentes o secciones: las *acciones de declaración* y las *acciones ejecutables*.

Las declaraciones definen o declaran las variables que tengan nombres. Las acciones ejecutables son las acciones que posteriormente deberá realizar la computación cuando el algoritmo convertido en programa se ejecute.

```
algoritmo
cabecera del programa
sección de declaración
sección de acciones
```

1- Cabecera del programa

Todos los algoritmos y programas deben comenzar con una cabecera en la que se exprese el identificador o nombre correspondiente con la palabra reservada que señale el lenguaje. En PSeInt, la palabra reservada es Algoritmo.

```
Algoritmo sin_titulo
      <Acciones>
FinAlgoritmo
```

Donde la palabra sin_titulo debe ser reemplazada por el nombre del algoritmo.

2- Sección de declaración

En esta sección se declaran o describen todas las variables utilizadas en el algoritmo, listándose sus nombres y especificando sus tipos (ver apéndice al final de la guía). La declaración de variables en PseInt comienza con la palabra reservada *Definir* seguida del *nombre de la variable* (identificador) la palabra reservada *como* y luego el tipo de dato.

```
Definir <identificador> como <Tipo de dato>
```

3- Sección de acciones

En esta sección se describe paso a paso cada una de las instrucciones que llevará a cabo el algoritmo/programa para obtener una solución a un determinado problema.

Apéndice

Tipos de datos en PSeInt

Los tipos de datos básico soportados son los siguientes:

✔ Entero: solo números enteros.

✔ Real: números con cifras decimales. Para separar decimales se utiliza el punto. Ejemplo: 3.14

✔ Caracter: cuando queremos guardar un carácter.

✔ Logico: cuando necesitamos guardar una expresión lógica (verdadero o falso)

✔ Cadena: cuando queremos guardar cadenas de caracteres.

Notas:

✔ Cadena y Caracter son términos equivalentes, no genera error que las escribamos indistintamente.

✔ El plural de Caracter es Caracteres o Cadena

Ejemplo de declaración de variables:

Si queremos declarar una variable de tipo entero se escribe:

```
Definir varNumero Como Entero
```
-> (*varNumero* se convierte en una variable de tipo entero)

Operadores

Este pseudolenguaje dispone de un conjunto básico de operadores que pueden ser utilizados para la construcción de expresiones más o menos complejas. La siguiente tabla exhibe la totalidad de los operadores de este lenguaje reducido:

Operador	Significado	Ejemplo
Relacionales		
>	Mayor que	3>2
<	Menor que	'ABC'<'abc'
=	Igual que	4=3
<=	Menor o igual que	'a'<='b'
>=	Mayor o igual que	4>=5
<>	Distinto que	Var1<>var2
Lógicos		
& ó Y	Conjunción (y).	(7>4) & (2=1) //falso
| ó O	Disyunción (o).	(1=1 | 2=1) //verdadero
~ ó NO	Negación (no).	~(2<5) //falso
Algebraicos		
+	Suma	total <- cant1 + cant2
-	Resta	stock <- disp – venta
*	Multiplicación	area <- base * altura
/	División	porc <- 100 * parte / total
^	Potenciación	sup <- 3.41 * radio ^ 2
% ó MOD	Módulo (resto de la división entera)	resto <- num MOD div

Reglas de prioridad:

Las expresiones que tienen dos o más operandos requieren unas reglas matemáticas que permitan determinar el orden de las operaciones, se denominan reglas de prioridad o precedencia y son:

1. Las operaciones que están encerradas entre paréntesis se evalúan primero. Si existen diferentes paréntesis anidados (interiores unos a otros), las expresiones más internas se evalúan primero.
2. Las operaciones aritméticas dentro de una expresión suelen seguir el siguiente orden de prioridad:

   ✔ operador ( )

   ✔ operadores unitarios (potenciación),

   ✔ operadores *, /, % (producto, división, módulo)

   ✔ operadores +, – (suma y resta).

En caso de coincidir varios operadores de igual prioridad en una expresión o sub -expresión encerrada entre paréntesis, el orden de prioridad en este caso es de *izquierda a derecha*, y a esta propiedad se denomina asociatividad.

Funciones Matemáticas

Las funciones en el pseudocódigo se utilizan de forma similar a otros lenguajes. Se coloca su nombre seguido de los argumentos para la misma encerrados entre paréntesis, por ejemplo, trunc(x). Se pueden utilizar dentro de cualquier expresión, y cuando se evalúe la misma, se reemplazará por el resultado correspondiente. Actualmente, todas las funciones disponibles son matemáticas (es decir que devolverán un resultado de tipo numérico) y reciben un sólo parámetro de tipo numérico. A continuación, se listan las funciones integradas disponibles:

Función	Significado
RC(X)	Raíz cuadrada de x
ABS(X)	Valor absoluto de x
LN(X)	Logaritmo absoluto de x
EXP(X)	Función Exponencial de x
SEN(X)	Seno de x
COS (X)	Coseno de x
TAN(X)	Tangente de x
ASEN(X)	Arcoseno de x
ACOS (X)	Arcocoseno de x
ATAN(X)	Arcotangente de x
TRUNC(X)	Parte entera de x
REDON(X)	Entero más cercano a x
AZAR(X)	Entero aleatorio entre 0 y x-1

Notas:

✔ La función raíz cuadrada no debe recibir un argumento negativo.

✔ La función exponencial no debe recibir un argumento menor o igual a cero.

Primitivas Secuenciales (Comandos de Entrada, Proceso y Salida)

✔ Lectura (Entrada).

✔ Asignación (Proceso).

✔ Escritura (Salida).

Lectura o entrada

La *instrucción Leer* permite ingresar información desde el ambiente.

```
Leer <variable1>, <variable2>, ..., <variableN>
```

Esta instrucción lee N valores desde el ambiente (en este caso el teclado) y los asigna a las N variables mencionadas. Pueden incluirse una o más variables, por lo tanto, el comando leerá uno o más valores.

Asignación o proceso

La *instrucción de asignación* permite almacenar un valor en una variable, se puede realizar de dos maneras:

```
<variable> <- <expresión>
<variable> = <expresión>
```

Al ejecutarse la asignación, primero se evalúa la expresión de la derecha y luego se asigna el resultado a la variable de la izquierda. El tipo de la variable y el de la expresión deben coincidir.

Escritura o salida

La *instrucción Escribir* permite mostrar valores al ambiente.

```
Escribir <exprl> , <expr2> , ... , <exprN>
```

Esta instrucción imprime al ambiente (en este caso en la pantalla) los valores obtenidos de evaluar N expresiones. Dado que puede incluir una o más expresiones, mostrará uno o más valores.