INTRODUCCIÓN A LOS SISTEMAS TELEMÁTICOSII

(01B)

Métodos para resolver problemas

Ir a NIBBEL AUTOMATION

Fin del artículo

Venancio Guntiñas Rodríguez
vguntinas2@gmail.com

Índice

DEFINICIÓN DE ACCIÓN, PROCESO Y PROCESADOR

Una **acción** es un suceso producido por un sistema sobre otro sistema.
Toda acción se produce durante un intervalo de tiempo finito, lo cual, significa que, existe un instante inicial (t_o) en el cual comienza la acción y un instante final (t_f) en el cual termina la acción. **Toda acción produce un cambio en el estado del sistema en el que actúa**.
La mayoría de los sucesos no se realizan de inmediato, sino que requieren una cierta progresión hacia el fin deseado, surge así el concepto de proceso.
Un **proceso** es un conjunto de acciones organizadas en el tiempo que realizan un suceso en un sistema. De otra manera, un proceso es una acción que se puede descomponer en otras más simples.
Un **procesador** es un sistema capaz de ejecutar un determinado proceso. Para ello, necesita entender el enunciado del proceso y ejecutar las acciones correspondientes.

Inicio
fin artículo

CLASIFICACIÓN DE LOS PROCESOS

Los procesos pueden clasificarse de la siguiente manera:

1) **Paralelos**.- se ejecutan simultáneamente dos o más acciones , en general, requieren varios procesadores o un multiprocesador.

2) **Secuenciales**.- en cada instante solo puede ejecutarse una acción. Existe un orden establecido para la ejecución de las acciones que forman el proceso y una acción solo puede comenzar cuando la acción anterior esta completamente terminada.

3) **Combinacionales**.- realizan una sola acción en cada instante, la acción total no se divide en etapas. Hablando propiamente, no son procesos sino los componentes con que se construyen los procesos.

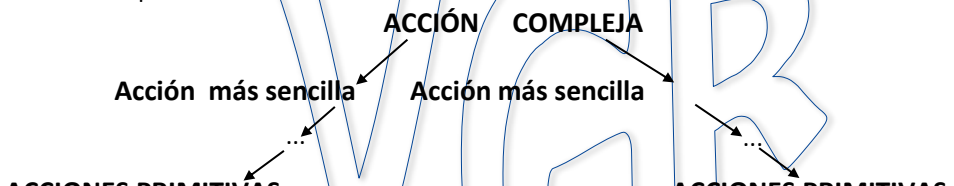
Inicio
fin_artículo

ACCIONES PRIMITIVAS

Una **acción** es **primitiva para un cierto procesador** si el enunciado de dicha acción basta para que el procesador la ejecute sin necesidad de información suplementaria.

Para que un procesador pueda ejecutar una acción no primitiva, primero hay que descomponer dicha acción en un conjunto de acciones primitivas.

Existen numerosos métodos para **descomponer una acción en acciones primitivas**. Uno de los más usados es el **análisis descendente** mediante el cual, una acción compleja se divide en otras más sencillas, cada una de éstas se divide a su vez en otras más sencillas y así sucesivamente hasta que la acción inicial queda descompuesta en un conjunto de acciones primitivas.



Inicio
fin_artículo

CONCEPTO DE PROBLEMA

Al **plantear un problema** se trata de buscar un **resultado** partiendo de unos **datos**, de tal manera que, para los mismos datos han de obtenerse siempre los mismos resultados.

Matemáticamente puede definirse un problema como una aplicación **P** de un conjunto de datos **D** en un conjunto de resultados **R**.

$$P: D \longrightarrow R$$

Para **definir bien un problema** hay que responder a las siguientes preguntas:

1)¿Qué datos o entradas se requieren?
2)¿Cuál es el resultado o salida deseada?
3)¿Qué método produce la salida deseada?

Inicio
fin_artículo

MÉTODOS PARA LA RESOLUCIÓN DE PROBLEMAS

A principios del siglo XX, los matemáticos esperaban poder construir un sistema completo y consistente, en el que, dada cualquier fórmula se pudiese decir si era verdadera o falsa. El **problema de la decisión** propuesto por **David Hilbert** trataba de descubrir un método general para decidir si una fórmula lógica es verdadera o falsa.

En los años treinta del siglo XX, **Gödel** demostró su famoso **teorema**: toda formulación axiomática consistente en la teoría de números, contiene proposiciones indecibles. Es decir, **cualquier teoría matemática es incompleta, siempre hay en ella afirmaciones que no se pueden demostrar ni negar**.

En 1937 **A.M.Turing** demostró que **existen problemas irresolubles**, es decir, ninguna máquina de Turing y por tanto, ninguna máquina, puede obtener su solución.

Para resolver problemas existen dos métodos fundamentales:

1) **Métodos heurísticos**.- Basándose en juicios e interpretaciones resuelven casos particulares. Sólo son validos para ese caso en particular.
2) **Métodos algorítmicos**.- Utilizan algoritmos para resolver el problema.
Los métodos algorítmicos se pueden implementar en computadores, los métodos heurísticos es difícil implementarlos en computadores, sólo las llamadas técnicas de inteligencia artificial hacen posible algunas de estas implementaciones.
Inicio
fin_artículo

CONCEPTO DE ALGORITMO

Dado un problema y un procesador determinado para resolverlo, un algoritmo es un método paso a paso que permite al procesador resolver el problema. Es decir, **un algoritmo es cualquier descripción del problema mediante una secuencia ordenada y finita de acciones primitivas del procesador que, a partir de los datos de entrada, permita obtener el resultado buscado**.
El algoritmo está determinado por el problema y el procesador, no obstante, para elaborar un algoritmo no es necesario conocer físicamente el procesador que va a ejecutarlo, basta con conocer sus acciones primitivas.
Inicio
fin_artículo

CARACTERÍSTICAS DE UN ALGORITMO

Un buen algoritmo ha de poseer las siguientes características:
A) Ser finito.- El número de acciones primitivas que describen al algoritmo ha de ser finito pues, de lo contrario, no podría obtenerse la solución al problema.
B) Precisión.- El algoritmo ha de indicar el orden en que se realiza cada acción, es decir, lo que ha de hacer en cada instante el procesador.
C) Estar bien definido.- Si para un mismo conjunto de datos se ejecuta el algoritmo varias veces, se ha de obtener siempre el mismo resultado. Para ello, cada orden del algoritmo ha de ser interpretada por el procesador de una forma precisa y no ambigua, cada orden ha de corresponder a una acción primitiva del procesador.
D) Ser eficaz.- Todo algoritmo ha de poder ejecutarse eficazmente. La eficacia puede definirse atendiendo a diversas características del entorno o del procesador. Algunas ***posibles definiciones*** son las siguientes:

1) Un algoritmo es más eficaz que otro si realiza el mismo trabajo en menos tiempo.
2) Cuando los recursos son limitados, un algoritmo es más eficaz que otro si utiliza menos objetos o si requiere menos espacio para almacenarlo.
3) Un algoritmo es más eficaz que otro si resulta más económico.

E) Existencia de ordenes de entrada, proceso y salida.- Puesto que ha de resolver un problema, un algoritmo poseerá órdenes que permitan entradas de datos, órdenes que permitan procesar los datos y órdenes que permitan la salida de los resultados.
Inicio
fin_artículo

TEORÍA Y ANÁLISIS DE ALGORITMOS

Para que un procesador realice un proceso que resuelva un problema es necesario:

a) Suministrarle un algoritmo adecuado.
b) Poder interpretar el algoritmo, es decir:
- Comprender las instrucciones de cada paso.
- Realizar las operaciones correspondientes.

Dado un problema, en general existen varios algoritmos que lo resuelven. Por lo tanto, habrá que determinar cual de ellos es el mejor.
El **análisis algorítmico** trata de determinar el comportamiento de un algoritmo en la resolución de un problema. La **teoría de algoritmos** trata de la existencia o no existencia de algoritmos eficaces para resolver ciertos problemas.
Inicio
fin_artículo

ALGORITMOS Y ESTRUCTURAS DE DATOS

Los algoritmos están estrechamente relacionados con las estructuras de datos que manejan.
El tipo de datos que se utiliza para resolver un problema y la frecuencia con que se realizan ciertas operaciones sobre los mismos, determina el tipo de estructura de datos a utilizar y en consecuencia el algoritmo adecuado para resolver el problema.
Normalmente habrá que encontrar un compromiso entre el tiempo de ejecución y el espacio de almacenamiento, los cuales, suelen estar en relación inversa.
Inicio
fin_artículo

COMPLEJIDAD DE UN ALGORITMO

La **complejidad de un algoritmo M** es una función ***f(n)*** que da el tiempo y/o el espacio de almacenamiento necesario para la ejecución del algoritmo, en función del tamaño ***n*** de los datos de entrada.
Con frecuencia el espacio de almacenamiento es un múltiplo de n, por lo que, *salvo que se indique lo contrario, la complejidad se referirá al tiempo de ejecución*.
La función f(n) no sólo depende de n, también puede depender de los datos particulares.
Para analizar la complejidad de un algoritmo suelen estudiarse los casos siguientes:
a) **Caso peor.-** Es el máximo valor que puede alcanzar ***f(n)*** para cualquier entrada posible.
b) **Caso medio.-** Es el valor más esperado de ***f(n)***. Para establecerlo se supone que los datos de entrada se presentan de acuerdo a una ley de probabilidad.

Si a n_1 le corresponde una probabilidad P_1
Si a n_2 le corresponde una probabilidad P_2
.....
.....
Si a n_k le corresponde una probabilidad P_k
El ***valor medio o más esperado*** será

$$E= n_1P_1+ n_2P_2+.....+ n_kP_k$$

c) **Caso mejor.-** Es el valor mínimo que puede presentar ***f(n)***.
Inicio
fin_artículo

REPRESENTACIÓN DE ALGORITMOS

Los algoritmos pueden representarse de varias maneras:
1) Matemáticamente, mediante **fórmulas**.
2) Mediante **pseudocódigo**.
3) Mediante **diagramas de flujo**.
4) Mediante **diagramas N-S**.
Veamos algunas de ellas:

REPRESENTACIÓN MATEMÁTICA DE UN ALGORITMO

Cada momento de la ejecución de un algoritmo está caracterizado por:
1) Los valores de las variables de entrada y salida en ese instante: ***i***
2) La orden que debe ejecutarse: ***o***
Los valores de ***(i, o)*** en cada instante definen el ***estado del algoritmo*** en dicho instante.
A la secuencia de acciones que se obtiene al ejecutar un algoritmo le corresponde una secuencia de estados asociados a cada momento de la ejecución.
3) Si llamamos **I** al conjunto de valores de entradas y salidas del algoritmo y llamamos **O** al conjunto de todas las ordenes del algoritmo, el producto cartesiano **IxO** será el conjunto de los estados del algoritmo:

$$Q = IxO = \{ (i,o) / i \in I \ \ y \ \ o \in O \}$$

Si q_n es un elemento de **Q** se representará $q_n = (i_n, o_n)$, es decir, si i_n es la información disponible en el instante t_n y o_n es el conjunto de órdenes a ejecutar en el instante t_n, **el par** (i_n, o_n) es el estado del algoritmo en el instante t_n .

Puesto que la estructura de un algoritmo consiste en un conjunto de órdenes cuya ejecución da lugar a una secuencia de operaciones, podemos expresar dicha estructura mediante una aplicación:

$$F: I \times O \longrightarrow I \times O$$
$$F: Q \longrightarrow Q$$

tal que, a un par (i_n, o_n) mediante **F** le corresponde un par (i_{n+1}, o_{n+1})

$$F(i_n, o_n) = (i_{n+1}, o_{n+1})$$

La ejecución de la orden o_n cuando la información tiene el valor i_n da como resultado un nuevo valor de la información i_{n+1} y a continuación se ejecutará la orden o_{n+1}. La aplicación **F** la podemos descomponer en dos aplicaciones:

$$F_1 : I \times O \longrightarrow I \qquad (i_n, o_n) \longrightarrow i_{n+1}$$

$$F_2 : E \times I \longrightarrow O \qquad (i_n, o_n) \longrightarrow o_{n+1}$$

Podemos establecer la siguiente ***definición matemática de algoritmo*** :
Un algoritmo es la cuádrupla **(I, O, F_1, F_2)**
en donde:

I=informaciones
O =órdenes
F_1=aplicación que nos da informaciones
F_2=Aplicación que nos da órdenes

Inicio
fin artículo

NOTACIÓN EN PSEUDOCÓDIGO

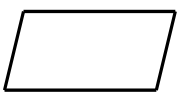
Durante el diseño de un algoritmo no es aconsejable utilizar un lenguaje de programación debido a su rigidez. Por este motivo, se utiliza una notación intermedia entre el lenguaje natural y el de programación que, sea flexible para expresar las diversas acciones a realizar y al mismo tiempo imponga algunas limitaciones.
Llamaremos **pseudocódigo** a una notación que permite describir un algoritmo para que un procesador resuelva un problema, utilizando palabras y frases del lenguaje natural sujetas a ciertas reglas.
Puede considerarse como un primer borrador, en el cual, *nos concentramos en la lógica y en las estructuras de control que permitan resolver el problema sin preocuparnos la sintaxis de un lenguaje concreto.*
No obstante, el pseudocódigo utilizado ha de describir todas las instrucciones y estructuras fundamentales utilizadas en programación.

DIAGRAMAS DE FLUJO

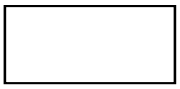
Un **flujograma** u **organigrama** es una representación gráfica de un algoritmo en la cual, las diversas ***operaciones*** que forman la secuencia del algoritmo, se representan mediante ***cajas*** de diversas formas geométricas y el **flujo de la información** o de los datos entre las diversas operaciones se indica mediante **flechas.** *En un organigrama se observa fácilmente el orden lógico o secuencia en que deben realizarse las operaciones y la naturaleza cíclica (repetitiva) o no de las diferentes operaciones lógicas o aritméticas*. No obstante, si bien lo anterior es cierto para algoritmos sencillos, no lo es *cuando el algoritmo es complicado* por lo que, en estos casos, *no son convenientes los flujogramas u organigramas.*
A continuación se representan los **principales símbolos utilizados** en la construcción de diagramas de flujo.
Símbolos:

- Caja redondeada o elipse: **Comienzo y fin del algoritmo**.

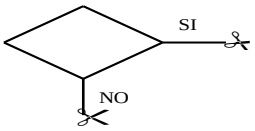
- Romboide: **Entrada/Salida de datos.**



- Caja cuadrada: **Operación o acción**.



- Rombo: **Toma de decisión**.

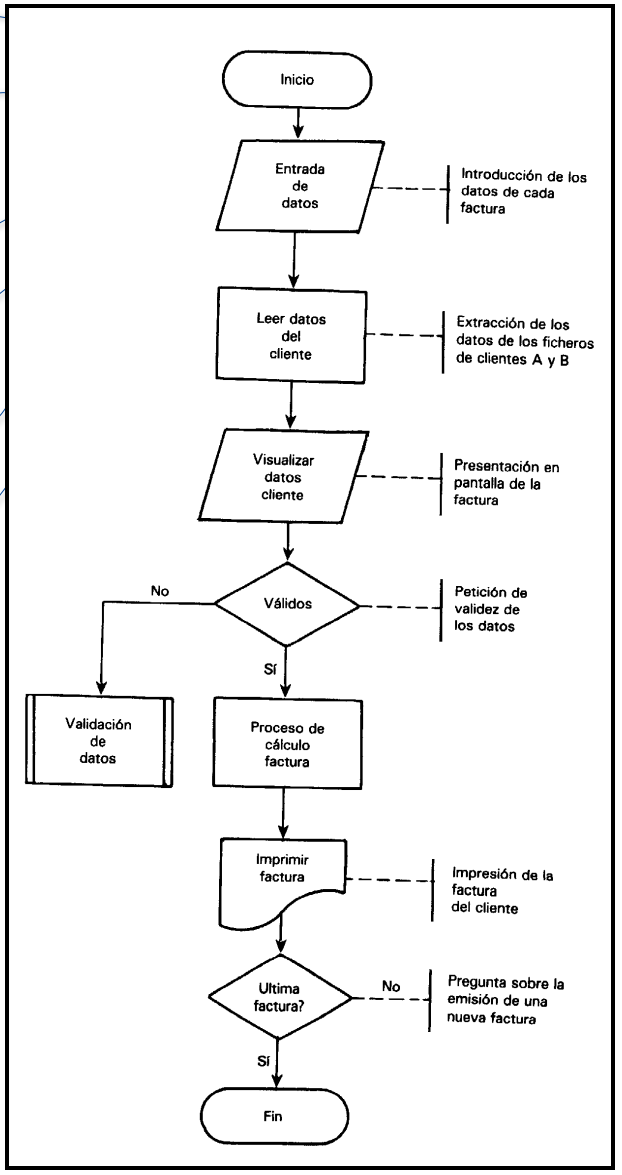


Subprogramas:

- Círculos: **Punto de conexión**.

Ejemplo:

Se quiere emitir facturas introduciendo por el teclado los datos almacenados en un fichero de disco. El modelo de la factura se presentará en la pantalla y una vez realizada se sacará por la impresora.



Inicio
fin_artículo

CONCEPTO DE MÁQUINA

De un modo general, MÁQUINA es todo artificio que permite utilizar la energía para un fin determinado. Por ejemplo, en un embalse hay almacenada una energía en forma potencial pero, no es possible utilizarla para mover un Molino si no se dispone de una rueda de paletas, de una turbina o de otro dispositivo similar.

CLASIFICACIÓN GENERAL DE LAS MÁQUINAS

Según la definición dada, puede establecerse la siguiente clasificación:

- **Transductores**.- son dispositivos cuya misión es transformar la energía de una forma en otra. Por ejemplo: una pila transforma energía química en energía eléctrica. Un altavoz transforma energía eléctrica en energía mecánica (sonido). Un motor de explosion transforma la energía interna del combustible en energía mecánica.
- **Mecanismos**.- Son dispositivos que permiten una mejor utilización de la potencia (energía/tiempo) que se dispone. Dicho de otra manera, los mecanismos permiten realizar un trabajo con menos esfuerzo o con más comodidad. Se tiene una fuerza y se trata de aplicarla convenientemente para realizar un trabajo determinado. Puesto que el trabajo es igual al desplazamiento por el valor de la fuerza en la dirección del desplazamiento, los mecanismos aprovechan en cada caso, la forma más ventajosa de ejercer la fuerza modificando el desplazamiento. Son ejemplos de mecanismos: las palancas, las poleas, los tornillos,…
- **Máquinas informáticas (procesadores de información)**.- dispositivos que facilitan y aumentan la capacidad del cerebro humano para manejar la energía externa (control de otras máquinas) y la interna (facilitan la solución de los problemas que se plantea, al facilitar el procesamiento de la información).

Inicio
fin_artículo

RENDIMIENTO DE UNA MÁQUINA

Hasta la fecha, no se ha encontrado ningún fenómeno que contradiga el **principio de conservación de la energía**, en consecuencia, **las máquinas cumplen dicho principio y no pueden actuar como fuentes de energía**. Lo que nos dan en forma de enegía útil para nuestros propósitos, hay que suministrársela en otra forma de energía.
La máxima potencia útil que puede suministrarnos una máquina es igual a la potencia de otro tipo que se le le suministre. Sin embargo, en la práctica, parte de la potencia suministrada a la máquina, se pierde en rozamientos, interferencias u otros fenómenos y, en consecuencia, no puede convertirse en potencia útil.

Potencia suministrada a la máquina= potencia útil producida + pérdidas

Una máquina será tanto más eficaz cuanto menores sean sus pérdidas. Para valorar la eficacia de una máquina se define el rendimiento (η):

$$\eta = \frac{\text{Potencia útil que suministra la máquina}}{\text{Potencia total suministrada a la máquina}}$$

Desde el punto de vista de la energía, cualquier tipo de máquina consume energía. Los transductores, se utilizan porque nos devuelven un tipo de energía que nos es útil a partir de una energía que no nos es útil. Por ejemplo, a nuestra vivienda nos llega enegía eléctrica pero, no podemos utilizarla directamente pues, incluso podría producirnos la muerte. Para poder utilizarla, necesitamos máquinas como estufas eléctricas, motores eléctricos,… . Incluso los mecanismos que no transforman la energía y suministran menos energía de la que se les suministra, son útiles. Una manera rápida de convencerse de ello, es intentar levantar un automóvil a pulso y con el gato o sacar un cubo de agua de un pozo a pulso o con una polea. Es decir, los mecanismos facilitan a los seres humanos la realización de un trabajo. Las máquinas informáticas consumen energía pero nos ahorran mucho tiempo en la realización de un trabajo, incluso pueden realizar trabajos que no sabríamos realizar directamente.

Inicio
fin_artículo

MÁQUINAS INFORMÁTICAS

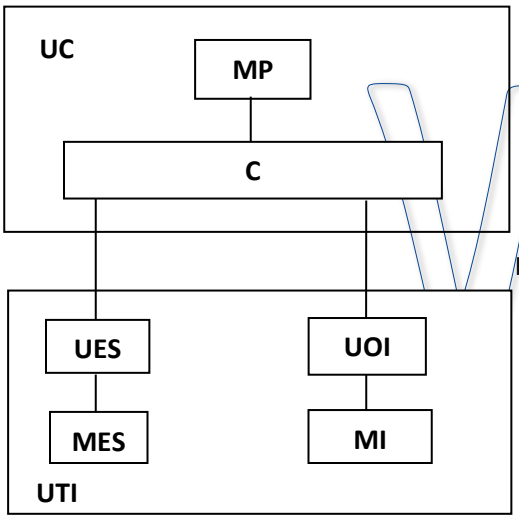
Una máquina informática es el **sistema resultante de la materialización de un algoritmo**, es decir, es un **dispositivo que nos permite resolver un problema**.
Una máquina ha de ser capaz de:

- Ejecutar órdenes que nos permitan obtener los valores de la información (datos y resultados) en el instante siguiente.
- Generar la secuencia de operaciones a realizar en el instante siguiente.

DEFINICIÓN MATEMÁTICA DE MÁQUINA:

Desde el punto de vista matemático, una definición muy general que sirve para máquinas materiales o máquinas abstractas es la siguiente:
Una máquina es el triple : J,M,P en donde:

J = Capacidad de la máquina para ejecutar un conjunto de órdenes O.
M=Capacidad de la máquina para memorizar un conjunto de estados Q = IxO.
P = Programa o conjunto de programas que la máquina es capaz de ejecutar.



El **diagrama de bloques** será el de la figura.

Podemos distinguir los siguientes elementos:

A) Una **unidad de tratamiento de la información (UTI)**, formada por:

- Subsistemas, que realizan las operaciones internas de la máquina **UOI**. Estas operaciones podrán ser en general aritméticas o lógicas por lo que llamaremos a este bloque **ALU (unidad aritmético lógica)**.
- Subsistemas, que realizan la **comunicación de la máquina con el exterior UES**. En general, los llamaremos **unidades de entrada/salida, I/O**.
- Subsistemas para **memorizar** las informaciones internas a la máquina **MI** y las informaciones de entrada y salida **MES**.

B) Una **unidad de control UC**, formada por:

- Subsistemas que sean capaces de generar la secuencia de operaciones y ordenar su ejecución a la unidad de tratamiento de la información **C**.
- Puede tener una memoria de programa **MP** que sirve para almacenar el algoritmo a ejecutar.
- **Buses** de comunicación e interconexión entre los diversos bloques.

En muchos casos es conveniente agrupar en un sólo subsistema bien una parte o bien la totalidad de la memoria (que está distribuida en el esquema anterior).

Inicio
fin_artículo

INSTRUCCIONES Y LENGUAJE MÁQUINA

El objetivo final de una máquina consiste en realizar las operaciones que indican los algoritmos de los problemas a resolver. Para que las órdenes de un algoritmo puedan ser comprendidas por la máquina, previamente deben codificarse en un lenguaje que pueda entender la máquina. A este lenguaje se le llama "**lenguaje máquina**".
Como todo procesador, una máquina posee un conjunto de **acciones primitivas** que es capaz de ejecutar. A cada una de estas acciones primitivas, le corresponderá una **palabra código** diferente del lenguaje máquina. A cada una de estas palabras código le llamaremos **instrucción máquina**. En consecuencia, **cada instrucción máquina realiza una operación (acción) diferente con uno o más operandos (datos). Para que una orden de un algoritmo pueda ser ejecutada por una máquina, ha de coincidir con una de las instrucciones máquina o ha de estar compuesta por una sucesión de instrucciones máquina**.

Inicio
fin_artículo

CONCEPTO DE PROGRAMA

Pueden considerarse tres puntos de vista:

VISIÓN ELEMENTAL

Todo procesador posee un conjunto de acciones primitivas que sabe ejecutar directamente. Cada una de estas acciones se representa mediante una palabra código diferente que llamaremos instrucción primitiva para el procesador.

El conjunto de instrucciones primitivas forma un lenguaje que entiende directamente el procesador. **Dado un algoritmo A que se desea ejecutar mediante un procesador P se llama programa del algoritmo A para el procesador P a la secuencia de instrucciones primitivas que representa al algoritmo**. Puede decirse que:

Un programa es un conjunto de instrucciones que controlan el funcionamiento de una máquina.

Pero del mismo modo que una casa es mucho más que una colección de ladrillos, un programa es más que un conjunto desordenado de instrucciones.

Un programa tambien es algo más que la solución de un problema concreto.

VISIÓN INSTRUMENTAL

Una **máquina de propósito general** puede considerarse como una herramienta que puede realizar una gran variedad de operaciones. **Un programa prepara a la máquina para que realice ciertas operaciones específicas**. Puede decirse que un programa crea un tipo especial de herramienta que se utiliza siguiendo unas normas.

Una herramienta no debe romperse con frecuencia. En consecuencia, **un programa ha de ser robusto, capaz de resistir un mal uso accidental o deliberado y debe diseñarse para funcionar repetidamente.**

Las herramientas facilitan el trabajo de usuario, por este motivo, **los programas deben ser tan fáciles de usar como sea posible** y han de estar coordinados con otros posibles trabajos que realice el usuario.

VISIÓN MULTINIVEL

Puede considerarse un programa como un **nivel o capa entre el hardware y el mundo exterior**. Esta es una de las razones por la que a los programas se les llama software (material blando).

Una máquina útil está formada por hardware recubierta por varias capas de software. Hay programas que configuran la máquina como una herramienta para ejecutar otros programas. Suelen existir varias capas o niveles, la más profunda interactúa con el hardware y la más externa interactúa con el usuario.

Todos los programas tienen varios “interfaces” o puntos de contacto con el hardware, con otros programas o con el usuario. Incluso distintos puntos de un mismo programa poseen interfaces entre ellos.

Inicio