TEMA 23

DISEÑO DE ALGORITMOS. TÉCNICAS DESCRIPTIVAS.

ÍNDICE

1. INTROCCIÓN.

La principal razón para que las personas apredan lenguajes y técnicas de programación es utilizar la computadora como herramienta para resolve problemas. La resolución de un problema exige al menos de los siguientes pasos:

definición y análisis del problema.
diseño del algoritmo.
transformación del algoritmo en un programa
ejecución y validación del programa.

El propósito del análisis del un problema es ayudar al programador a llegar a una cierta comprensión de la naturaleza del mismo. Una buena definición del problema, junto con una descripción detallada de las especificaciones de entrada/salida, son los requisitos más importantes para llegar a una solución eficaz. Paralelamente y en todas las fases se irá realizando la documentación del programa.

Se distinguen dos tipos de documentación:

Para programadores, incluye información técnica del programa, así como los algoritmos y variables y ficheros.
Para usarios, intrucciones que incluye la descripción de las tareas que realiza un programa y de las intrucciones a seguir para la intalación y arranque.

Como resultado de su trabajo, el programador proporciona un texto que describe por anticipado el conjunto de las operaciones que un ordenador debe ejecutar para resolver el problema del que tiene encomendado hallar una solución informática.
Buena parte de la dificultad del trabajo reside en que esta descripción debe tener en cuenta todas las posibilidades dque se puedan producir, con el fin de evitar que se produzcan fallos en el conjunto del tramtamineto por no haber previsto algún caso particular.

2. DEFINICION DE ALGORITMO.

Definimos un algoritmo como la descripción de un esquema de comportameinto expresado mediante un repertorio de acciones y de informaciones elementales identificadas, bien comprendidas y realizadas a priori. Este repertorio se denomina el **léxico del algoritmo.**

Un **algoritmo** a nivel informático es un procedimeitno efectivo que para todos los valores de datos dados obtiene la solución, si existe, o determina que dicha solución no existe. Esto quiere decir que el resultado de aplicar el algoritmo está definido en todos los casos

La existencia de un algoritmo para resolver un problema implica que dicho problema ha de ser decidible. Si un problema no es decidible no podrá haber ningún algoritmo que lo resuelva.

Un algoritmo debe ser preciso, indicando el orden de realización de cada paso, bien definido, si se sigue el algoritmo dos veces consecutivas con la misma entrada se debe obterner el mismo resultado y finito, es decir , debe terminar en algún momento.

3. DEFINICION DE PROGRAMA.

Definimos programa como un algortimo destinado a gobernar una máquina real. El objetivo del programador es la escritura de programas.

La realización de un programa pasa por la escritura de uno o más algoritmos que después se traducen en un lenguaje de alto nivel donde la máquina lo entenderá perfectamente.

4. TECNICAS DESCRIPTIVAS. LA PROGRAMACIÓN ESTRUCTURADA

Se denomina estructura a la manera en que diferentes partes se combinan para construir un todo.
Un programa tendrá la estructura correspondiente a la forma en que las distintas acciones individuales o cálculos parciales se combinan para construir el programa en su conjunto.

Reduciendo el problema de la estructuración de un programa a su expresión más sencilla, el planteamiento al que se llega es como combinar unas pocas operaciones de la manera más clara posible para construir otra más compleja.

5. TIPOS DE ESTRUCTURAS BÁSICAS.

5.1. Especificación formal.

Para estudiar los fundamentos de la programación estructurada vamos a utilizar principalmente el pseudocódigo. Esta es una técnica para expresar en lenguaje natural la lógica de un programa, es decir, su flujo de control. Los pseudocódigos utilizan palabras comohacer, ***si-entonces-sino, repeti-hasta, mientras….***
Unida a esta técnica s ha formalizadoun conjunto de reglas que permiten comprobar la validez de un algoritmo. La ejecución de una instrucción supone habitualmente unos cambios en su entorno. Para especificar un proceso (ejecución de programa o algoritmo), describiremos la relaciónentre los estado inicial y final del mismo. Esta relación, denominada especificación, se formaliza con la ayuda de predicados, que siempre tiene un valor de TRUE o FALSE.

En general,la especificación de un algoritmo constará de los siguientes elementos:

declaración de variables.
precondición
nombre del algoritmo
postcondición

donde la precondición y postcondición son predicados sobre las variables del proceso. Estos predicados reflejan , repectivamente la situación inicial y final pretendida de las variables de nuestro algoritmo.

5.2. Estructuras elementales

5.2.1 Asignación

La asignación consiste en asociar un valor a una variable. Su sintaxis es la siguiente: x:=E, que se lee como “x toma el valor de lo que valga E”. E debe ser una expresión válida del mismo tipo que x. Hay que tener en cuenta que primero se evalua la expresión y luego se copia en la variable el valor obtenido.

5.2.2 Composición secuencial

Consiste en la ejecución incondicional de una secuencia de sentencias.
Una operación compleja puede consistir en ejecutar una tras otra una secuencia de operaciones más sencillas.Así para indicar que éstas deben ejecutarse sucesivamente se dará la lista de dichas acciones en el orden en que deben ser ejecutadas, estas acciones se separaran por ; que juega dos papeles :

indicar el fin de una acción
indicar que el proceso no ha terminado.

5.2.3 Composición iterativa

Un operación compleja puede consistir en repetir una o más operaciones un número de veces, o hasta o mientras se cumpla una determinada condición. Para describir esta instrucción adoptamos en nuestra notación algorítmica a estructura:

Mientras B hacer S fminetras

Donde B es una expresión lógica y S una instrucción. Al principio de la ejecución, la condición B es comprobada y en caso de ser cierta, se procesa la instrucción S, conocida como cuerpo del bucle; el proceso se repite hasta que la condición B no se cumpla.

Toda iteración plantea dos tipos de problemas:

a) Determinar el efecto de la instrucción iterativa: para ello hemos de buscar una relación que se mantenga inalterada antes y después de cada iteración y que describa todos los estados por los que atraviesa el cómputo realizado por el bucle, observados justo antes de evaluar la condición B de terminación. A esta relación le llamaremos invariante.
b) Probar que la instrucción iterativa termina. Para ello es útil asociar a cada iteración una función que dependa de las variables y que actúe de limitadora o cota superior del número de iteraciones pendientes haciendo que el aserto B deje de cumplirse al dejar de ser estrictamente positiva.

EXISEN TAMBIEN OTRAS INSTRUCCIONES DE ITERCIÓN COMO SON : repetir S hasta B Y LA INSTRUCCIÓN para, QUE ES UN CASO DEL mientras.

5.3. Teorema de estructura.

Los fundamentos de la denominada programación estructurada se establecieron a principios de los años sesenta y se consolidaron con los trabajos de Dijkstra, Bohm ..El resumen de sus postulados es el siguiente: se dice que un programa es estructurado si se expresa únicamente mediante combinaciones de las estructuras básicas:

Secuencial
Alternativa
Iterativa

El teorema de estructura afirma que dado un programa P no estructurado con una única entrada y una única salida y sin interrupciones en el flujo de control, es posible construir un programa P' estructurado que obtnega los mismos resultados.

5.4. Estructuras que permiten reutilizar el código

Uno de los componentes que los lenguajes estructurados es que incorporan un tipo de secuencias algorítmicas individualizadas que pueden recibir o no valores de entrada y que también pueden devolver o no valores de salida.
Se trataría de sustituir todo un conjunto de instrucciones que puede incluir cualquier combinación de las denominadas estructuras básicas (secuencial, condicional, iterativa) por un identificador que puede incorporar la declaración de valores. Posteriormente en cualquier lugar del programa se podrá invocar al conjunto de instrucciones utilizando el identificador.

Tras la ejecución y según el tipo de estructura utilizada se pueden obtener resutados devueltos.
La diferencia entre procedimientos y funciones tiene su origen en las especificaciones de algunos lenguajes estructurados como PASCAL, aunque esta es muy poca.

6. PASOS A SEGUIR PARA DESARROLLAR PROGRAMAS DESDE ALGORITMOS.

Las recomendaciones de la programación estructurada se traducen en una metodología a seguir para desarrollar programas. Dicha metodología recibe el nombre general de **refinamiento progresivo.**

La técnica de refinamiento es del tipo descendiente (topdown).
Consiste en describir inicialmente el programa como una acción global y luego irlo decomponiendolo en partes de manera que en cada paso de descomposición se apliquen sólo las estructuras básicas recomendadas. La descomposición o refinamiento debe ir aplicándose a los datos al mismo tiempo que a las operaciones.

En programas voluminosos el proceso de refinamiento debe ir acompañado de una descomposición en módulos.

7. BIBLIOGRAFIA

Pérez Lobato, J.M.
Metodología de la programación
Alhambra-Longman, 1994

Joyanes Aguilar, L.
Fundamentos de programación
Mc Graw-Hill, 1992

Alfonso Ureña López
Fundamentos de Informática
Ra-ma, 1997