Universidad Nacional Autónoma de México

Facultad de Ingeniería

Estructura de Datos y Algoritmos I

Actividad 1

Bautista Corona Yahir

04/03/2021

Problema:
Cuestión que se plantea para hallar resultados a partir de un conjunto de datos conocidos.

Método para obtener un resultado: (estrategia)
Acciones meditadas y encaminadas hacia un fin determinado.

Análisis:
Examen detallado de algo para conocer sus características o cualidades y extraer conclusiones.
Partes:

a) Datos de entrada
b) Restricciones
c) Datos de salida

Datos de entrada:
Componentes que se emplean como inicio en un proceso.

Datos de salida:
Componente que se obtiene como resultado tras realizarse un proceso.

Pasos para resolver un problema:

a) Entender el problema
b) Analizar el problema
c) Diseñar y desarrollar la solución más eficiente
d) Probar la solución
e) Implementar la solución y probarla

Características de la mejor solución de un problema:

a) Debe ser una estrategia
b) Eficiente: cumplir con una cierta actividad
c) Eficaz: emplear recursos de la mejor manera en un tiempo adecuado
d) Basarse en el análisis
e) Resolver el problema
   ES LA MÁS INTELIGENTE

Caja negra:
Se desconoce el proceso, pero sí se sabe cuáles son las entradas y cuáles son las salidas.

Caja blanca:
Se conoce el proceso, se sabe cuáles son las entradas y como fueron procesadas para obtener las salidas.

Restricciones: (limitaciones) Ayuda a…
Encontrar una solución al problema
Solución completa o factible
Solución eficiente o eficaz

Ingeniería de software:
Es la aplicación de un enfoque sistemático para desarrollar, operar y mantener el software.

Ciclo de vida del software:
Se define en base en la norma ISO 12207
Un marco de referencia que contiene las actividades y las tareas involucradas en el desarrollo, la explotación y el mantenimiento de un producto de software, abarcando desde la definición hasta la finalización de su uso.

Definición de necesidades
Análisis
Diseño
Codificación
Pruebas
Validación
Mantenimiento y evolución

Programación:
Proceso de diseñar codificar, depurar y mantener el código fuente.

Algoritmo:
Es el conjunto de pasos ordenados (método) que permiten resolver un problema.
Es:
Finito: numero finito de pasos.
Preciso: datos de entrada, pasos, datos de salida.
Definido: pasos muy claros.

Los algoritmos se clasifican en:

a) Numéricos
b) No numéricos

Los algoritmos se representan en:

a) Gráfica: diagrama de flujo
b) No gráfica: seudocódigos

Pasos para resolver un problema:

1. Entender el problema
   Para empezar a resolver el problema debe quedar claro lo que esta planteado y a donde se quiere llegar.
2. Realizar el análisis
   Es importante la identificar la información con la que se cuenta y la que hace falta.
   Para ello se aplica un análisis.

a) Datos de entrada: deben identificarse cuantos datos se requieren para empezar a resolver el problema, de que tipo don estos datos (enteros, reales, alfanuméricos) y asignarles un identificador a cada uno para manipularlos.
b) Restricciones: deben identificarse que valores NO se pueden utilizar y cuales, si pueden usarse para resolver el problema, identificar las

unidades de medida que se deben emplear; que condiciones deben cumplirse.
Con base en los datos de entrada.

c) Datos de salida: deben identificarse cuantos datos se van a obtener al resolverse el problema, de que tipo son estos datos (enteros, reales, alfanuméricos) y asignarles un identificador a cada uno para manipularlos.

3. Diseñar la solución al problema
   La solución al problema debe ser eficiente y eficaz y debe de ser indispensable que tome en cuenta lo que se detecta en el análisis, de lo contrario, la solución será errónea.
   En este paso se describen de manera detallada y sin ambigüedades los pasos que se deben seguir para resolver el problema.
   En este mismo paso se debe comprobar el diseño de la solución que se acaba de realizar.
   Esta comprobación se realiza paso a paso sin saltarse paso alguno y sin inventar pasos inexistentes.
   A esta comprobación se le conoce como prueba de escritorio y comúnmente se realiza más de una.

   NOTAS:
   a) Las operaciones se representan:
      Identificador=operación NUNCA al revés
   b) Cuando se evalúan condiciones de la forma una tras otra o una dentro de otra.
   c) Para repetir un proceso (una introducción o varias) necesito un identificador contador, este va desde un valor inicial hasta un valor final y considera un incremento o decremento.
   d) Identificador contador nunca es el valor inicial ni el valor final ni el incremento o decremento.
   e) El identificador contador puede tomar el valor inicial para empezar a contar.
   f) Las repeticiones, las estructuras condicionales SIEMPRE deben tener instrucciones con jerarquía interna.

4. Probar la solución

Reglas para nombrar a un identificador:

1. Empezar con una letra.
2. Seguirle a esa primera letra otras letras o números.
3. No se emplean espacios en blanco.
4. No se emplean caracteres especiales.
5. Sensible a mayúsculas y minúsculas.
6. Nombre relacionado con el uso que se le da.
7. Puede usar guion bajo.
8. Las leyendas y unidades alfanuméricos van entre “joule” “El denominador no puede ser cero ”
9. Los valores se encierran entre ’+’

Codificación:

Existen dos tipos:

a) Codificación algorítmica
   1. Diagrama de flujo
      O
   2. Pseudocódigo

b) Codificación con lenguaje de programación
   Codificación algorítmica

   Está basada en estándares para llevarla a cabo

   I. Diagrama de flujo
      Es la representación del algoritmo de manera gráfica, por lo que emplea símbolos estandarizados para representar acciones.

Reglas para diseñar un diagrama de flujo:

1. Todo diagrama de flujo solo cuenta con un inicio y un fin.
2. Siempre se deben declarar TODOS los identificadores que se van a utilizar.
   NOTA: declarar significa indicar el nombre del identificador y su tipo para mencionar que EXISTE.
3. Las representaciones de los flujos deben ser RECTAS.
4. Todas las líneas que indican a los flujos deben estar conectadas a un símbolo (salen de un símbolo y llegan a otro).
5. Un diagrama se construye de arriba hacia abajo y de izquierda a derecha cuando hay una condición que necesita ser desplegada para evitar que se amontone con parte del diagrama.
6. NO se emplean notaciones como palabras, instrucciones o símbolos pertenecientes a un lenguaje de programación ya que el diagrama de flujo es INDEPENDIENTE de cualquier lenguaje.
7. A un símbolo solo puede llegarle UN solo flujo y si es necesario más de uno, estos flujos deben INCORPORARSE al ya existente.
8. Emplear conectores para indicar que pasa a otra hoja.
9. Emplear la notación camello para representar a los identificadores.
   - Lower camel: contador Uno
     energía Potencial
   - Upper camel: Contador Uno
     Energía Potencial

Simbología:

1. Proceso: colocar dentro del símbolo asignaciones.
   Identificador=valor
   Identificador=operación
   Acción
   Para las asignaciones se emplea el operador asignación (←)
   Para las operaciones se emplean los siguientes operadores:
   +, -, *, /, mod, ^, ()
2. Terminación: escribo Inicio o Fin
3. Condición simple: dentro del rombo se escriben las condiciones.
   Identificador operador Condición valor
   Operadores de tipo condición son: =, <>, <=, >=

Para tener más de una condición se emplean los operadores lógicos | (o), & (y)
Para agrupar se emplea un par de paréntesis: ()
Se debe colocar encima de cada flecha de la condición simple él **sí** y el **no**, para saber que rama se debe seguir cuando la condición se cumple y cuando no es así.

4,5,6. Entrada: dentro del símbolo se escribe únicamente el nombre del identificador se listan con una coma en el orden en el que se van a almacenar los valores.

4,7. Salida: dentro de símbolo se coloca lo que se va a mostrar.

I. Valor: coloco el identificador o los identificadores que almacenan el valor que se va a mostrar y que se listan con una coma en el orden deseado.
II. Cadenas: colocas a la cadena ENTRE comillas.

10. Conectores en la misma página.
Colocar un par del mismo número dentro del circulo por cada par de conectores. Se maneja a manera consecutiva la numeración.
11. Conectores en diferente página.
12. Colocar un par del mismo número dentro del pentágono por cada par de conectores. Se manejan de manera consecutiva la numeración, PERO es distinta a la de los conectores anteriores.
13. Subproceso o función: se escribe dentro del símbolo el nombre de la función y los argumentos de ella en caso de que existan.
14. Condición múltiple: dentro del símbolo se escribe un UNICO identificador y este identificador SOLAMENTE puede ser de tipo ENTERO o alfanumérico CARÁCTER.
Cada flujo representa una opción y puede haber mas flujos.
En cada flujo se indica una opción escribiendo al lado de la flecha el valor correspondiente y solo se evalúa si es exactamente igual a lo que contiene el identificador, NO se emplean operadores.
NOTA: la codificación algorítmica debe ser probada empleando na prueba de escritorio.

Pseudocódigo
Representación del algoritmo, pero empleando texto.
Codificación algorítmica NO GRAFICA.

Reglas:

1. Emplea etiquetas escritas en mayúsculas
NOTA: las etiquetas son las acciones que se realizan.
2. Todo pseudocódigo contiene las etiquetas INICIO y FIN para delimitar la representación de algorítmico, el resto de las instrucciones deben estar contenidas en ellas.
3. Las etiquetas también reciben el nombre de palabras reservadas.
4. Se emplean sangrías o tabulaciones para indicar jerarquía que tiene la instrucción.
5. Se deben declarar TODOS los identificadores empleados en el algoritmo.
6. Para representar la entrada/lectura de datos se utiliza la etiqueta LEER.
7. Para representar la salida/escritura de datos se utiliza la etiqueta ESCRIBIR.

8. Se usan los siguientes operadores:
   a) Operadores aritméticos: +, -, *, /, mod, ^, :=(operador asignación)
   b) Operadores lógicos: |&
   c) Operadores comparación: <>, <=, >=, < >
   d) Operadores de agrupación: ()
9. Empela la notación: camello: upper camel y lower camel
10. Emplea tres tipos de instrucciones de control de flujo: secuenciales, condicionales, repetitivas.
    a) Secuenciales: ESCRIBIR, LEER, HACER
    b) Condicionales

- Condicional simple
  Sintaxis
  Si condición ENTONCES
  Instrucciones o acciones
  FIN SI
  EN CASO CONTRARIO
  Instrucciones o acciones
  FIN EN CASO CONTRARIO
- Condición múltiple
  Sintaxis
  SELECCIONAR identificador EN
  CASO valor 1:
  Instrucciones o acciones
  CASO valor 2:
  Instrucciones o acciones
  CASO valor N:
  Instrucciones o acciones
  DEFECTO
  Instrucciones o acciones
  FIN SELECCIONAR
  NOTA: identificador es entero o alfanumérico de tipo carácter.

c) Repetitivas o iterativas:

- Mientras
  Verifica condiciones ANTES de ingresar a un ciclo, si las condiciones NO se cumplen JAMAS entra al ciclo.
  Sintaxis
  MIENTRAS condición ENTONCES
  Instrucciones o acciones
  FIN MIENTRAS
- Hacer mientras
  Realiza la acción al menos UNA VEZ y verifica las condiciones después de haber hecho esta acción para saber si se repite o no dicha acción.
  Sintaxis
  HACER
  Instrucciones o acciones
  MIENTRAS condición

Representación interna de los números
Definiciones:

a) Bit: unidad mínima de la información. Binary DigiT.
   Puede tomar como valores el 0 o el 1.
   0 → apagado
   1 → encendido
b) Byte: unidad fundamental de la información. Conjunto de 8 bits, cadenas de 8 bits.
c) Nibble: son 4 bits, es la mitad de un byte.
d) Palabra: conjunto de bits, en múltiplos de 8, conjunto de bytes.
e) Bit más significativo: es el bit que está más a la izquierda en la cadena de 1 y 0.
f) Bit menos significativo: es el bit que está más a la derecha en la cadena de 1 y 0.

Bit más significativo →01010101010101010101 ← Bit menos significativo

1. Sin signo: solo se representan los números positivos.
2. Magnitud verdadera y signo.
   El bit más significativo representa al signo:
   1→negativo, 0→ positivo