**1. IDENTIFICACIÓN DE LA GUÍA DE APRENDIZAJE**

* Denominación del Programa de Formación: **PROGRAMACIÓN DE SOFTWARE**
* Código del Programa de Formación: 228120
* Nombre del Proyecto: Implementación de Módulos en el Sistema Integral Web de Gestión de Procesos Educativos para el CEET
* Fase del Proyecto: Análisis
* Actividad de Proyecto: Determinar los requerimientos del sistema según especificaciones y lineamientos del proyecto.
* Competencia: Analizar los requisitos del cliente para construir el sistema de información
* Resultados de Aprendizaje Alcanzar: Identificar cada uno de los conceptos y principios que constituye la programación orientada a objetos para interpretar el diseño.
* Duración de la Guía: 6 Horas

**2. PRESENTACION**

TEMAS:

* Entidades Primitivas para el Desarrollo de Algoritmos
* Técnicas de Diseño,
* Técnicas para la Formulación de Algoritmos
* Estructuras Algorítmicas

**ENTIDADES PRIMITIVAS PARA EL DESARROLLO DE ALGORITMOS**

* Tipos de datos
* Expresiones
* Operadores y operandos
* Identificadores como localidades de memoria

**OBJETIVO EDUCACIONAL:**

El Aprendiz:

Conocerá las reglas para cambiar fórmulas matemáticas a expresiones válidas para la computadora, además de diferenciar constantes e identificadores y tipos de datos simples.

***Identificadores***

Los *identificadores* representan los datos de un programa (constantes, variables, tipos de datos). Un identificador es una secuencia de caracteres que sirve para identificar una posición en la memoria de la computadora, que nos permite accesar a su contenido.

Ejemplo: Nombre

Num\_hrs

Calif2

R***eglas para formar un Identificador***

Debe comenzar con una letra (A a Z, mayúsculas o minúsculas) y no deben contener espacios en blanco.

Letras, dígitos y caracteres como la subraya ( \_ ) están permitidos después del primer carácter.

La longitud de identificadores puede ser de hasta 8 caracteres.

***Constantes y Variables***

***Constante:*** Una constante es un dato numérico o alfanumérico que no cambia durante la ejecución del programa.

Ejemplo:

pi = 3.1416

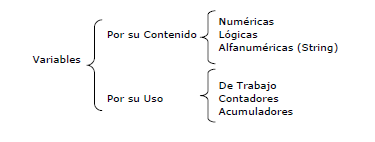
***Variable:*** Es un espacio en la memoria de la computadora que permite almacenar temporalmente un dato durante la ejecución de un proceso, su contenido puede cambia durante la ejecución del programa. Para poder reconocer una variable en la memoria de la computadora, es necesario darle un nombre con el cual podamos identificarla dentro de un algoritmo.

Ejemplo:

área = pi \* radio ^ 2

Las variables son: el radio, el área y la constate es pi

***Clasificación de las Variables***

******

***Por su Contenido***

***Variable Numéricas:*** Son aquellas en las cuales se almacenan valores numéricos, positivos o negativos, es decir almacenan números del 0 al 9, signos (+ y -) y el punto decimal. Ejemplo:

iva=0.15 pi=3.1416 costo=2500

***Variables Lógicas:*** Son aquellas que solo pueden tener dos valores (cierto o falso) estos representan el resultado de una comparación entre otros datos.

***Variables Alfanuméricas:*** Está formada por caracteres alfanuméricos (letras, números y caracteres especiales). Ejemplo:

letra=’a’ apellido=’lopez’ direccion=’Av. Libertad #190’

***Por su Uso***

***Variables de Trabajo:*** Variables que reciben el resultado de una operación matemática completa y que se usan normalmente dentro de un programa. Ejemplo:

suma=a+b/c

***Contadores:*** Se utilizan para llevar el control del número de ocasiones en que se realiza una operación o se cumple una condición. Con los incrementos generalmente de uno en uno.

***Acumuladores:*** Forma que toma una variable y que sirve para llevar la suma acumulativa de una serie de valores que se van leyendo o calculando progresivamente.

**TÉCNICAS DE DISEÑO**

o Top down

o Bottom up

**OBJETIVO EDUCACIONAL:**

El aprendiz: Conocerá las características de las técnicas de diseño más empleadas, así como su aplicación a cada tipo de problemas

***Top Down***

También conocida como de arriba-abajo y consiste en establecer una serie de niveles de mayor a menor complejidad (arriba-abajo) que den solución al problema. Consiste en efectuar una relación entre las etapas de la estructuración de forma que una etapa jerárquica y su inmediato inferior se relacionen mediante entradas y salidas de información.

Este diseño consiste en una serie de descomposiciones sucesivas del problema inicial, que recibe el refinamiento progresivo del repertorio de instrucciones que van a formar parte del programa.

La utilización de la técnica de diseño ***Top-Down*** tiene los siguientes objetivos básicos:

- Simplificación del problema y de los subprogramas de cada descomposición.

- Las diferentes partes del problema pueden ser programadas de modo independiente e incluso por diferentes personas.

- El programa final queda estructurado en forma de bloque o módulos lo que hace más sencilla su lectura y mantenimiento.

***Bottom Up***

El diseño ascendente se refiere a la identificación de aquellos procesos que necesitan computarizarse con forme vayan apareciendo, su análisis como sistema y su codificación, o bien, la adquisición de paquetes de software para satisfacer el problema inmediato.

Cuando la programación se realiza internamente y haciendo un enfoque ascendente, es difícil llegar a integrar los subsistemas al grado tal de que el desempeño global, sea fluido. Los problemas de integración entre los subsistemas son sumamente costosos y muchos de ellos no se solucionan hasta que la programación alcanza la fecha limite para la integración total del sistema. En esta fecha, ya se cuenta con muy poco tiempo, presupuesto o paciencia de los usuarios, como para corregir aquellas delicadas interfaces, que en un principio, se ignoran.

Aunque cada subsistema parece ofrecer lo que se requiere, cuando se contempla al sistema como una entidad global, adolece de ciertas limitaciones por haber tomado un enfoque ascendente. Uno de ellos es la duplicación de esfuerzos para accesar el software y mas aun al introducir los datos. Otro es, que se introducen al sistema muchos datos carentes de valor. Un tercero y tal vez el mas serio inconveniente del enfoque ascendente, es que los objetivos globales de la organización no fueron considerados y en consecuencia no se satisfacen.

**TÉCNICAS PARA LA FORMULACIÓN DE ALGORITMOS**

* Diagrama de flujo
* Pseudocodigo
* Diagrama estructurado (nassi-schneiderman)

**OBJETIVO EDUCACIONAL:**

El Aprendiz:

Será capaz de diferenciar los métodos de representación y formulación de algoritmos, así como de conocer las características más importantes de cada técnica.

Las dos herramientas utilizadas comúnmente para diseñar algoritmos son:

Diagrama de Flujo

Pseuducodigo

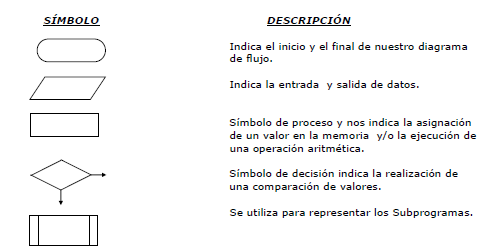
***Diagrama de Flujo***

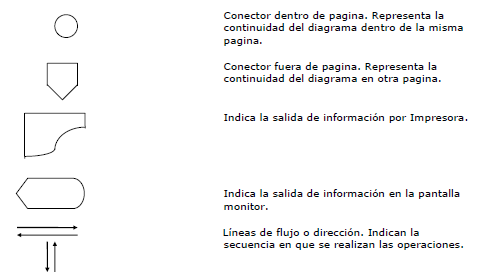
Un diagrama de flujo es la representación gráfica de un algoritmo. También se puede decir que es la representación detallada en forma gráfica de cómo deben realizarse los pasos en la computadora para producir resultados.

Esta representación gráfica se da cuando varios símbolos (que indican diferentes procesos en la computadora), se relacionan entre sí mediante líneas que indican el orden en que se deben ejecutar los procesos.

Los símbolos utilizados han sido normalizados por el instituto norteamericano de normalización (ANSI).

***SÍMBOLO DESCRIPCIÓN***





***Recomendaciones para el diseño de Diagramas de Flujo***

* Se deben se usar solamente líneas de flujo horizontal y/o vertical.
* Se debe evitar el cruce de líneas utilizando los conectores.
* Se deben usar conectores solo cuando sea necesario.
* No deben quedar líneas de flujo son conectar.
* Se deben trazar los símbolos de manera que se puedan leer de arriba hacia abajo y de izquierda a derecha.
* Todo texto escrito dentro de un símbolo deberá ser escrito claramente, evitando el uso de muchas palabras.

***Pseudocodigo***

Mezcla de lenguaje de programación y español (o inglés o cualquier otro idioma) que se emplea, dentro de la programación estructurada, para realizar el diseño de un programa. En esencial, el pseudocodigo se puede definir como un lenguaje de especificaciones de algoritmos.

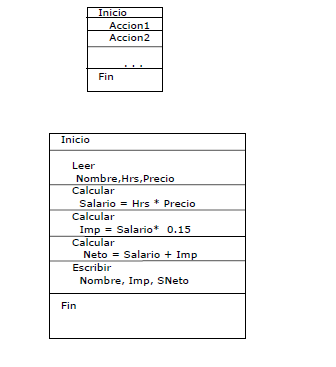
Es la representación narrativa de los pasos que debe seguir un algoritmo para dar solución a un problema determinado. El pseudocodigo utiliza palabras que indican el proceso a realizar.

***Ventajas de utilizar un Pseudocodigo a un Diagrama de Flujo***

* Ocupa menos espacio en una hoja de papel
* Permite representar en forma fácil operaciones repetitivas complejas
* Es muy fácil pasar de pseudocodigo a un programa en algún lenguaje de programación.
* Si se siguen las reglas se puede observar claramente los niveles que tiene cada operación.

***Diagramas estructurados (Nassi-Schneiderman)***

El diagrama estructurado N-S también conocido como diagrama de chapin es como un diagrama de flujo en el que se omiten las flechas de unión y las cajas son contiguas. Las acciones sucesivas se pueden escribir en cajas sucesivas y como en los diagramas de flujo, se pueden escribir diferentes acciones en una caja. Un algoritmo se represente en la sig. forma:

****

**ESTRUCTURAS ALGORITMICAS**

* Secuenciales

- Asignación

- Entrada

- Salida

* Condicionales

- Simples

- Múltiples

* Repetición fila condicional

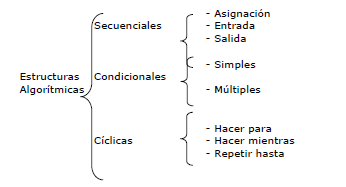
**OBJETIVO EDUCACIONAL:**

El aprendiz:

Conocerá las diferentes estructuras algorítmicas como componentes básicos de los programas y aplicara la combinación de ellas para el desarrollo de algoritmos más complejos.

**ESTRUCTURAS ALGORÍTMICAS**

Las estructuras de operación de programas son un grupo de formas de trabajo, que permiten, mediante la manipulación de variables, realizar ciertos procesos específicos que nos lleven a la solución de problemas. Estas estructuras se clasifican de acuerdo con su complejidad en:

****

***Estructuras Secuenciales***

La estructura secuencial es aquella en la que una acción (instrucción) sigue a otra en secuencia. Las tareas se suceden de tal modo que la salida de una es la entrada de la siguiente y así sucesivamente hasta el fin del proceso. Una estructura secuencial se representa de la siguiente forma:

Inicio

Accion1

Accion2

.

.

AccionN

Fin

***- Asignación:*** La asignación consiste, en el paso de valores o resultados a una zona de la memoria. Dicha zona será reconocida con el nombre de la variable que recibe el valor. La asignación se puede clasificar de la siguiente forma:

* ***Simples:*** Consiste en pasar un valor constate a una variable (a=15)
* ***Contador:*** Consiste en usarla como un verificador del número de veces que se realiza un proceso (a=a+1)
* ***Acumulador:*** Consiste en usarla como un sumador en un proceso (a=a+b)
* ***De trabajo:*** Donde puede recibir el resultado de una operación matemática que involucre muchas variables (a=c+b\*2/4).

***- Lectura:*** La lectura consiste en recibir desde un dispositivo de entrada (p.ej. el teclado) un valor. Esta operación se representa en un pseudocodigo como sigue:

Leer a, b

Donde “a” y “b” son las variables que recibirán los valores

***-Escritura:*** Consiste en mandar por un dispositivo de salida (p.ej. monitor o impresora) un resultado o mensaje. Este proceso se representa en un pseudocodigo como sigue:

Escribe “El resultado es:”, R

Donde “El resultado es:” es un mensaje que se desea aparezca y R es una variable que contiene un valor.

**ESTRATEGIAS DE ENSEÑANZA**

**ACTIVIDADES DE APRENDIZAJE**

1. Resolver cada uno de los siguientes problemas propuestos, Utilizando el programa bloc de notas, en cada problema debe hacer el análisis del problema, indicando datos de entrada, datos de salida y operaciones.

**Problemas Propuestos**

1) Dada una cantidad en pesos, obtener la equivalencia en dólares, asumiendo que la unidad cambiaría es un dato desconocido.

2) Leer un número y escribir el valor absoluto del mismo.

3) La presión, el volumen y la temperatura de una masa de aire se relacionan por la fórmula:

masa = (presión \* volumen)/(0.37 \* (temperatura + 460))

4) Calcular el número de pulsaciones que una persona debe tener por cada 10 segundos de ejercicio, si la fórmula es:

num. pulsaciones = (220 - edad)/10

5) Calcular el nuevo salario de un obrero si obtuvo un incremento del 25% sobre su salario anterior.

**CONTROL DEL DOCUMENTO**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Nombre** | **Cargo** | **Dependencia** | **Fecha** |
| **Autor (es)** | **Mauricio Estupiñan.** | **Instructor** | **CEET** | **Julio 2018** |