



**Nombre del alumno: Johanne Joaquín
Arriaga Díaz**

Nombre del profesor: Ensayo de unidad 3

Nombre del trabajo: Icel Bernardo Lepe

**Materia: Fundamentos y lógica de
programación**

PASIÓN POR EDUCAR

Grado: Tercer cuatrimestre

Grupo: ISC13SDC0119-F

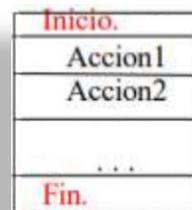
Pseudocódigo

El Pseudocódigo es una forma más simple y entendible de los lenguajes de programación y esto se debe a que mezcla la lógica y lenguaje de la programación y la mezcla con el idioma deseado un ejemplo de programa en el que se puede programar con pseudocódigo es el llamado "Pseint". Esta forma simple de los lenguajes de programación se emplea para realizar el diseño de un programa y en resumen se trata de un lenguaje para especificar los pasos y procesos a seguir para solucionar un problema (Algoritmos) y hacerlo de manera sencilla y comprensible para casi cualquier persona basándose en la lógica pues para programar solo es necesario representar de manera narrativa los pasos que se deben seguir y esto es muy simple ya que las palabras utilizadas para describir las acciones serán en el idioma que conocemos o con un estilo comprensible.

Las ventajas claves del pseudocódigo es principalmente que ocupa menos espacio en una hoja de papel, permite representar fácilmente operaciones repetitivas y complejas ya que la forma en que el programa lleva a cabo el algoritmo que el programador escribe simplifica estos procesos por ejemplo en un lenguaje formal de programación como lo es JAVA para escribir texto en pantalla se usa "System.out.println("Texto")" pero en pseudocódigo (dependiendo del programa usado) puedes tan solo escribir "Escribir: "Texto" ", es muy fácil pasar de pseudocódigo a un lenguaje de programación ya que solo bastaría reemplazar las palabras claves por las necesarias para el lenguaje de programación y acoplarlo a la lógica de este como ya vimos anteriormente en el ejemplo de la forma de escribir texto en pantalla con Java y pseudocódigo.

Diagramas estructurados (Nassi-Schneiderman)

El diagrama estructurado N-S o diagrama de Chapín es una forma gráfica de representar un algoritmo y ordenarlo, parecido a un diagrama de flujo pero sin las flechas y las cajas están unidas unas a otras las acciones relacionadas o sucesivas se escriben en cajas sucesivas unidas unas a otras y para acortar el proceso o mostrar acciones que tienen relación se pueden escribir varias acciones en una misma caja.



Tipos de datos y expresiones

Dependiendo de qué dato sea será el tipo que tiene asociado ya sea simple o estructurado, además se utilizan identificadores para cada dato en este caso los ejemplos son del lenguaje Java, los datos simples son:

Datos numéricos: Que por su naturaleza representan valores de forma numérica y debido a la cantidad de usos que se le pueden asignar podemos realizar operaciones comunes y aritméticas. Y para insertar números en Java se utiliza “int” para números enteros y “Double” para números de punto flotante o reales con puntos decimales fraccionables o de alguna manera científica

Lógicos: Se basan en el principio de cierto o falso, ya que permite comparar datos ya sean numéricos o alfanuméricos.

Datos alfanuméricos (String):

Los datos alfanuméricos están formados por números y letras y en la programación están encerrados entre comillas en ellas puede introducirse tanto nombres direcciones entre otras cosas además se pueden introducir números pero estos pierden su propiedad matemática. Algunos ejemplos de String son “Ingeniería en sistemas computacionales” y “2020”.

Expresiones

Las expresiones son una combinación de datos numéricos y alfanuméricos que como se vio anteriormente son números reales o enteros y caracteres no numéricos así como símbolos y operaciones, gracias a esto podemos plantear operaciones complejas en un programa como por ejemplo “ $a+(b+3)/c$ ” lo cual es muy útil si utilizamos dicho programa para resolver o plantear cosas de este tipo y para resolver lo anterior cada expresión tomara el valor que debe declararse antes de ejecutar el código y este realizara las operaciones de manera indicada

Operadores y operandos

Los operadores son aquellos que nos permiten relacionar y manipular los datos que ya declaramos estos se dividen en aritméticos con los que podemos realizar operaciones matemáticas con los valores ya determinados, lo cual se utiliza de manera practica por ejemplo al crear una calculadora o plantarle al programa creado realizar cálculos simples

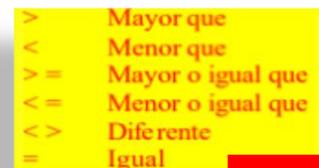
tal como si se trata de un videojuego a la cantidad de puntos que ya tenemos sumarle los que acabamos de obtener sería algo parecido a "1000 + X" donde 1000 serían los puntos que tenemos y x los puntos que obtuvimos en el caso de que ganemos 500 la x sería reemplazada por este valor y el resultado sería "1000+500= 1500" al igual que si por el contrario perdemos esa cantidad de puntos el operador aritmético sería "-" y entonces la operación sería "1000-500= 500" gracias a esto podemos resolver sumas (+), restas (-), multiplicación (*), división (/), modulo (el residuo de una división) su simbología es (Mod).

Prioridad de operadores aritméticos

Como en las matemáticas las reglas aplicables son las mismas al realizar operaciones pero con las excepciones de que los datos con paréntesis anidados se leen de dentro hacia afuera y los datos que son más internos dentro del paréntesis se evalúa primero lo cual también puede causar errores lógicos de parte del programador pues si aplicamos las leyes matemáticas tal cual son la maquina podría realizar las operaciones de manera distinta a la deseada, un ejemplo sería que leyéramos sin reglas matemáticas "4+2*5=30" pero la maquina lo interpretara como "4+(2*5)=14" ya que realizara primero lo que está en paréntesis.

Operadores racionales

Comparan datos y retornaran un valor de cierto o falso en el caso de Java se declaran con la palabra "Boolean". Estos comparan datos del mismo tipo (numéricos o de cadena) y como son comparaciones tienen el mismo nivel de importancia al momento del programa evaluar esos datos. Gracias a estas comparaciones podemos hacer que un mismo programa siga distintos caminos dependiendo de los datos que introduzcamos, podríamos crear un programa que realice encuestas con dos o más opciones como por ejemplo preguntar la edad y si se introduce una edad mayor 18 años sugerir productos para personas adultas y si es menor sugerir para personas de corta edad o de acuerdo a la edad. Los operadores que se utilizan son los de la figura 1.1.



>	Mayor que
<	Menor que
> =	Mayor o igual que
< =	Menor o igual que
< >	Diferente
=	Igual

1.1

Operadores lógicos

Estos son para relacionar valores, las expresiones que se utilizan tienen una equivalencia entendible para nosotros y de uso cotidiano pero no podemos emplearla ya que eso al igual que en otras partes del código podría causar errores de lógica algunos de estos son “and” que equivale a “Y”, “or” que equivale a “O”, “not” que equivale a “no”, y sirven para expresar por ejemplo que “1 es mayor que 0 y 6 menor que 9” o “(1 > 0) and (6 < 9)”

Estructuras secuenciales

Nos sirven para asignar a cada acción su lugar en concreto y en secuencia o que acción sigue después de la ya realizada para eso se debe asignar a cada variable un valor que se guardara, los valores simples no cambian, el contador sirve para saber la cantidad de veces que se repetirá el proceso, el acumulador sirve para introducir más procesos, los de lectura permiten que el usuario introduzca los valores hacia el programa y los de escritura muestran el resultado en un dispositivo de salida.

Estructuras condicionales

Son aquellas que pueden seguir distintos caminos dependiendo de los datos que introduzcan los usuarios o las decisiones tomadas eso se refiere a que la salida de datos del programa depende de lo que el usuario decida lo cual es muy útil al crear programas interactivos los cuales interactuaran con los usuarios, un ejemplo claro de esto son las contraseñas, solo hay una contraseña correcta si la que introduces es incorrecta el programa optara por no darte acceso pero si la contraseña es correcta te dará el acceso.