INTRODUCCIÓN A LA PROGRAMACIÓN
NOTAS DE PROFUNDIZACIÓN
DATOS ESTRUCTURADOS Y ARREGLOS

¡BIENVENIDOS!

A continuación, profundizaremos en lo relativo a los conceptos alrededor de los datos estructurados, los arreglos unidimensionales, bidimensionales, las funciones y procedimientos. El aprendizaje esperado es organizar los datos estructurados considerando el uso de los arreglos, funciones y procedimientos, para la solución algorítmica de problemas determinados.

Para alcanzarlo de manera exitosa te sugerimos realizar lo siguiente:

Revisar la presentación de la semana para conocer el aprendizaje esperado de la semana.

Estudiar con tiempo los contenidos de la semana y revisar los recursos adicionales, lo que te preparará para realizar las actividades evaluativas de manera exitosa.

Revisar las instrucciones de la evaluación final para que puedas realizar tus consultas a tiempo.

Publicar tus dudas y comentarios en el foro de interacción semanal.

¡Que disfrutes aprendiendo y compartiendo!

COMENCEMOS

Antecedentes relevantes

En la programación para realizar programas eficientes y recomendables, es muy importante diseñar algoritmos óptimos, para lo cual, generalmente se precisan estructuras de datos eficientes. Los datos estructurados son conjuntos de datos organizados de manera lógica y coherente. En la informática, los datos estructurados se utilizan para facilitar su almacenamiento y procesamiento. En este sentido, estructurar datos se refiere a organizarlos dentro del computador, lo que permite realizar operaciones y manejar gran cantidad de información de la mejor manera posible. Dentro de la programación computacional existen varias estructuras de datos como: arreglos, listas enlazadas, pilas, colas y árboles binarios. La programación modular (las funciones y los procedimientos) permiten la resolución de un problema descomponiéndolo en varios subproblemas, independientes uno del otro, de forma tal que pueden ser tratados por separado. Esto permite probar módulos o subprogramas de forma individual, corrigiendo errores antes de incluirlo en el programa principal.

En relación a lo anterior revisa la siguiente información:

En el campo de la tecnología de la información, las estructuras de información son aquellas que permiten a los programadores poder organizar la información de manera eficiente y, en última instancia, diseñar la solución adecuada para un problema específico.
Un arreglo es una estructura de datos que consiste en un conjunto de elementos del mismo tipo, que se acceden a través de un índice numérico.
Dentro de la programación existe un paradigma, llamado programación por procedimientos y funciones, lo que constituye la base de la programación modular y permite realizar tareas concretas varias veces dentro de una página, sistema o aplicación.

Tabla 1: Datos Estructurados.

Fuente: Elaboración propia

DATOS ESTRUCTURADOS ARREGLOS, FUNCIONES Y PROCEDIMIENTOS

Al resolver un problema, el primer paso para solucionar el problema es percibir la información que compone el universo de los datos en estudio. De modo que, la elección de los datos y su posterior organización es esencial para concretar y solucionar un problema. Las estructuras de datos son una forma de organizar y almacenar datos de manera eficiente y útil. Los arreglos son un tipo de estructura de datos que permiten almacenar una colección de elementos del mismo tipo de datos. Son muy útiles para almacenar y procesar conjuntos de datos, ya que permiten acceder a cada elemento de manera sencilla y rápida mediante un índice.

En la programación, es importante seleccionar la estructura de datos adecuada para cada problema, ya que cada una tiene sus propias ventajas y desventajas. Por ejemplo, los arreglos son muy rápidos para acceder a elementos individuales, pero son ineficientes para insertar o eliminar elementos en el medio del arreglo. Por otro lado, las listas enlazadas son más lentas para acceder a elementos individuales, pero son más eficientes para insertar y eliminar elementos en cualquier lugar de la lista.

Las estructuras de datos y los arreglos son importantes en la programación porque nos permiten organizar y almacenar datos de manera eficiente y útil, y nos facilitan el procesamiento de estos datos de manera rápida y sencilla.

DATOS ESTRUCTURADOS ARREGLOS, FUNCIONES Y PROCEDIMIENTOS

Cuando se menciona la frase datos estructurados, se hace referencia a los datos que existen en cualquier base de datos, su mejor característica es que siempre están organizados y mediante el lenguaje máquina se comprende fácilmente. Esto permite a los programadores

que aplican bases de datos relacionales, ingresar, manipular y buscar datos estructurados con mayor inmediatez, tales como direcciones, nombres, fechas, números de cuentas bancarias, números de tarjeta de crédito o débito, entre otras.

Las Estructuras de datos permiten organizar y almacenar información de manera eficiente y útil. Algunos ejemplos de estructuras de datos estructuradas comunes incluyen arreglos, listas, árboles y tablas hash. Cada una de estas estructuras tiene sus propias ventajas y desventajas y se utiliza en diferentes situaciones según las necesidades del problema.

Para conocer más:

- Las estructuras de datos son una parte esencial de la programación. Son la forma en que organizamos y almacenamos los datos en nuestros programas. Dependiendo de la estructura de datos que elijamos podremos acceder a los datos de manera más rápida o más lenta, y podremos realizar ciertas operaciones más fácilmente que con otras estructuras de datos. Por lo tanto, elegir la estructura de datos adecuada puede tener un gran impacto en la eficiencia y el rendimiento de nuestro programa.
- Las estructuras de datos también nos permiten modelar y representar los datos de una manera más natural y adecuada al problema que estamos tratando de resolver
- Las estructuras de datos son una herramienta esencial en la programación y su elección y uso adecuado pueden tener un gran impacto en la eficiencia y el rendimiento de nuestro programa, así como en la capacidad de modelar y representar de manera adecuada los datos con los que estamos trabajando.

ARREGLOS

En algoritmos y programación se le conoce a un arreglo como un conjunto de datos agrupados en una misma variable. También, otros autores definen a los arreglos como vectores. Como lo indica el Duque (2017), “se debe entender como arreglo a una estructura en la que se almacena una colección de datos del mismo tipo (ejemplo: las calificaciones de los alumnos de un grupo, sus edades, sus estaturas, etcétera)” (p. 143).

Estos arreglos se caracterizan por tener el mismo nombre de variable, tener un índice que indica la posición de uno de los datos almacenados en él, tener un tamaño en posiciones y un tipo definidos de datos, bien sea de tipo texto, número, booleano, entre otros. Se tienen dos tipos de arreglos para dar solución a distintos problemas que podamos enfrentar, por un lado, se tienen los arreglos unidimensionales, y por el otro los arreglos bidimensionales.

UNIDIMENSIONALES (VECTORES)

Los vectores, son arreglos unidimensionales o lineales que guardan datos en posiciones que se incrementan de manera secuencial y utilizan un índice para ubicar estos datos. De acuerdo con la definición de Duque (2017), “Los vectores son arreglos que contienen un solo índice que indica la posición que guarda el dato dentro del arreglo” (p. 144). La estructura de un vector unidimensional es como sigue:

Nombres[5]={“Juan”,”Maria”,”Pedro”,”Rosa”,”Elena”};

Este ejemplo es un arreglo de tipo texto que almacena en la misma variable “Nombres” cinco (5) datos, de los cuales el primero se encuentra en la posición cero (0) y el último en la posición cuatro (4). Dado lo anterior, si se quiere acceder a algún dato en particular dentro del arreglo, se debe indicar el nombre del arreglo y su índice de la siguiente forma: **Nombres[3];**

En este caso, nos devolverá el dato que contiene el índice en la posición 3, y para este ejemplo en particular es el dato “Rosa”. Es importante recordar que a pesar de que el índice es el número 3, no es la tercera posición sino la cuarta, puesto que el indexado comienza en el número 0.

```
1  Algoritmo Arreglo_Unidimensional
2      Definir Acum,i,Nota Como Entero
3      Definir Prom Como Real
4      Dimension Nota[10]; // Declaración del Arreglo
5      Para i<-1 Hasta 3 Con Paso 1 Hacer
6          Escribir "Ingrese La Nota[",i,"]"
7          Leer Nota[i]
8          Acum=Acum+Nota[i]
9      Fin Para
10     Prom=Acum/3
11     Escribir"El promedio es: ",Prom
12 FinAlgoritmo
```

Imagen 1: Pseudocódigo Unidimensional

Fuente: Elaboración propia

Para conocer más:

- Se requiere realizar un algoritmo que permita leer tres (3) notas y devolver el promedio mediante un mensaje en pantalla. En este caso, se usan arreglos únicamente para mostrar cómo el almacenamiento en cada uno y cómo se realiza una operación simple con los datos.
- En este ejercicio se aplicó una estructura repetitiva para acortar el código, es decir el ciclo “Para”. Es importante notar que se utilizó una variable que permitiera ir acumulando las notas (la variable “Acum”), y finalmente realizar la operación. También se comienza a utilizar una variable que maneja las posiciones con la variable (i).

- Como se puede mostrar en este ejemplo se realiza un ciclo Para, se declara un arreglo que se llama Nota[10] y luego se utiliza la variable para realizar el cálculo del promedio que es Prom.

BIDIMENSIONALES (MATRICES)

Los arreglos bidimensionales, también conocidos como matrices, guardan datos en dos posiciones que se pueden llamar temporalmente "X" y "Y", y utilizan dos índices para ubicar estos datos. De acuerdo con la definición de Duque (2017) "por tal motivo se deben especificar dos posiciones (dos subíndices), uno para la fila y otro para la columna, a este tipo de arreglos indistintamente se les llama tablas o matrices." (p. 157).

La estructura de un vector bidimensional es como sigue:

Edades [3][3] = {"55","76","42"};

{"37","22","15"};

{"18","14","20"};

El ejemplo anterior es un arreglo de tipo texto que almacena en la misma variable "Edades" tres datos, en dos subíndices. Dado lo anterior, si se quiere acceder a algún dato en particular dentro del arreglo, se debe indicar el nombre del arreglo y su índice de la siguiente forma: **Edades [3][2];**

En este caso**, Edades [3][2]** devolverá el dato que contiene la posición del índice 3 para la fila, y 2 para la columna, para este **ejemplo el dato "14**".

```
Algoritmo Arreglo_Bidimensional
Definir Datos,Nombre1,Nombre2,Nombre3 Como Caracter
Dimension Datos[2,3]; Dimension Edad[2,3]
Definir indice_f, indice_c, Edad, Ed1,Ed2,Ed3 como número;
Definir indicador Como Logico
Indicador = falso;
Para indice_f<-1 Hasta 2 Con Paso 1 Hacer
 Para indice_c<-1 Hasta 3 Con Paso 1 Hacer
  Si indicador=falso Entonces
    Escribir"Ingrese el nombre [",indice_f,",",indice_c,"]";
     Leer Datos[indice_f,indice_c];
     Nombre1=Datos[1,1];Nombre2=Datos[1,2];Nombre3=Datos[1,3];
  SiNo
     Escribir "Ingrese la edad [",indice_f,",",indice_c,"]";
      Leer Edad[indice_f,indice_c];
       Ed1=Edad[2,1];Ed2=Edad[2,2];Ed3=Edad[2,3];
  Fin Si
 Fin Para
    indicador = verdadero;
 Fin Para
Escribir"-------------------------------------------------------------------------------"
Escribir"- Nombre No.1:-> ",Nombre1,"- Nombre No.2:-> ",Nombre2,"-  Nombre No.3:->
",Nombre3
Escribir"- Edad No.1----> ",Ed1,"   - Edad No.2---> ",Ed2,"   - Edad No.3---> ",Ed3
Escribir"-------------------------------------------------------------------------------"
FinAlgoritmo
```

Imagen 2: Pseudocódigo Unidimensional

Fuente: Elaboración propia

Para conocer más:

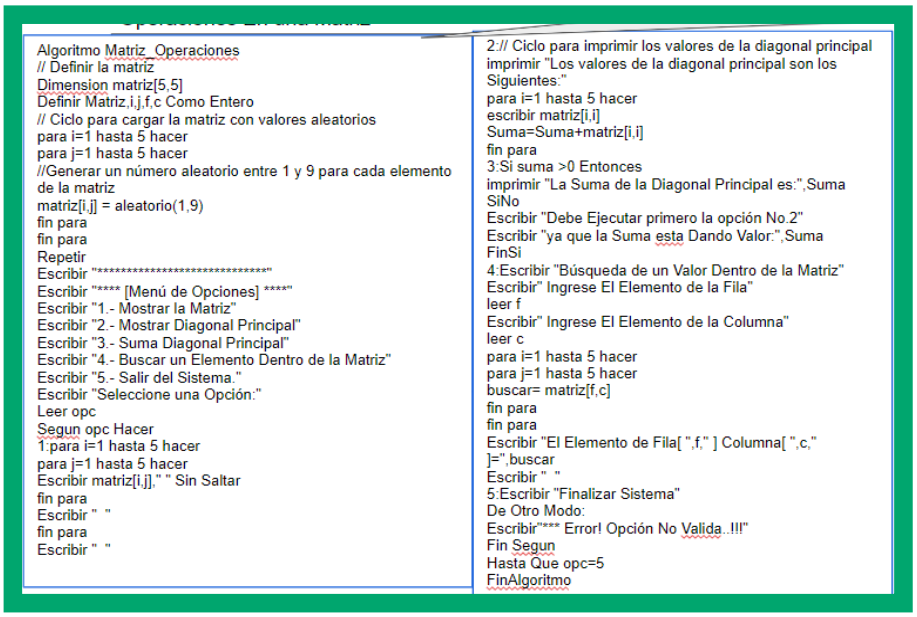
- En este ejemplo se realizará la carga por medio del teclado de 3 nombres y 3 edades.
- Se realizará la declaración del arreglo bidimensional con (Dimension Datos[2,3] y Edad[2,3].
- En este ejercicio se ha utilizado una variable de tipo booleana que ayudará a identificar cuándo el primer ciclo “para” haya dado la primera iteración y saber que ya no es necesario solicitar al usuario ingresar el nombre de la persona, sino la edad.
- En este ejemplo se les asignan a 3 variables los nombres y las edades, como se puede mostrar donde se le asignan las variables del arreglo y luego en la salida, los datos.

A continuación, mostraremos un ejemplo:

Realizar un algoritmo que permita crear una matriz 5x5 y cargar los valores de la matriz de forma aleatoria.

El programa debe ofrecer un menú de opciones utilizando la estructura de control "switch case", (en Pseint se conoce como “**Según opción Hacer”**) donde se presenten las siguientes operaciones:

- Mostrar la matriz: debe realizar un ciclo que recorra la matriz y mostrar cada uno de los valores por pantalla.
- Indicar los valores de la diagonal principal: debe realizar un ciclo que recorra la diagonal principal de la matriz y mostrar los valores que la conforman.
- Sumar la diagonal principal: debe realizar un ciclo que recorra la diagonal principal de la matriz y mostrar la suma de sus valores por pantalla.
- Buscar un valor en la matriz: dada una fila y una columna ingresadas por el usuario, el programa debe mostrar el valor que se encuentra en esa posición de la matriz.
- Cada una de estas opciones debe ser presentada al usuario a través del menú, permitiendo que el usuario seleccione la operación que desee realizar.



```
Algoritmo Matriz_Operaciones
// Definir la matriz
Dimension matriz[5,5]
Definir Matriz,i,j,f,c Como Entero
// Ciclo para cargar la matriz con valores aleatorios
para i=1 hasta 5 hacer
para j=1 hasta 5 hacer
//Generar un número aleatorio entre 1 y 9 para cada elemento
de la matriz
matriz[i,j] = aleatorio(1,9)
fin para
fin para
Repetir
Escribir "*******************************"
Escribir "**** [Menú de Opciones] ****"
Escribir "1.- Mostrar la Matriz"
Escribir "2.- Mostrar Diagonal Principal"
Escribir "3.- Suma Diagonal Principal"
Escribir "4.- Buscar un Elemento Dentro de la Matriz"
Escribir "5.- Salir del Sistema."
Escribir "Seleccione una Opción:"
Leer opc
Segun opc Hacer
1:para i=1 hasta 5 hacer
para j=1 hasta 5 hacer
Escribir matriz[i,j]," " Sin Saltar
fin para
Escribir " "
fin para
Escribir " "
```

```
2:// Ciclo para imprimir los valores de la diagonal principal
imprimir "Los valores de la diagonal principal son los
Siguientes:"
para i=1 hasta 5 hacer
escribir matriz[i,i]
Suma=Suma+matriz[i,i]
fin para
3:Si suma >0 Entonces
imprimir "La Suma de la Diagonal Principal es:",Suma
SiNo
Escribir "Debe Ejecutar primero la opción No.2"
Escribir "ya que la Suma esta Dando Valor:",Suma
FinSi
4:Escribir "Búsqueda de un Valor Dentro de la Matriz"
Escribir" Ingrese El Elemento de la Fila"
leer f
Escribir" Ingrese El Elemento de la Columna"
leer c
para i=1 hasta 5 hacer
para j=1 hasta 5 hacer
buscar= matriz[f,c]
fin para
fin para
Escribir "El Elemento de Fila[ ",f," ] Columna[ ",c,"
]=",buscar
Escribir " "
5:Escribir "Finalizar Sistema"
De Otro Modo:
Escribir"**** Error! Opción No Valida..!!!"
Fin Segun
Hasta Que opc=5
FinAlgoritmo
```

Imagen 3: Pseudocódigo Unidimensional

Fuente: Elaboración propia

Para conocer más:

- Esta **primera parte del ejemplo** muestra la parte inicial donde se le da el nombre al Algoritmo y se definen las variables y se declara la Matriz o Arreglo Bidimensional.
- En la segunda parte del ejemplo, se muestra un conjunto de operaciones que se le realizan a la matriz y están agrupadas por opciones.
- Como se puede mostrar en este ejemplo se utilizan Arreglos Bidimensionales, estructuras cíclicas **Para, para crear la matriz, Condicional simple (SI), la** Estructura Repetitiva **Repetir hasta, para hacer uso de un menú de opciones.**

AVERIGUA MÁS

TE RECOMENDAMOS

Con los avances en la informática que hoy en día existen en las organizaciones, es posible estar a la vanguardia tecnológica mejorando sus procesos, permitiendo diseñar algoritmos óptimos para una mejor interacción de sus sistemas. Te invitamos a reflexionar un poco más al respecto de la importancia de los Datos Estructurados y Arreglos:

Duque, D., Saint-Priest, Y., Segovia, P., & Loaiza, D. (2017). *Algoritmos y programación en pseudocódigo.*

Joyanes, L. (2003). *Fundamentos de programación: algoritmos y estructura de datos y objetos.*