Universidad Nacional Autónoma de México

Facultad de Ingeniería

Estructura de Datos y Algoritmos I

Actividad 1

Bautista Corona Yahir

08/06/2021

Declaraciones globales: variables o funciones que deben reconocerse en todo el programa. Van antes del main()

Directivas de pre-procesamiento (aka bibliotecas o librerías): #include

Archivo: se sustituye por el archivo que contiene las indicaciones necesarias que permiten que una estructura funcione correctamente.

Lectura de datos (almacenamiento de datos cuando estos son pedidos) Scanf("Especificación de formato", &identificador);

Escritura de texto o datos (impresiones de texto o datos) puts("Texto"); o printf("Texto");

Impresión de un dato almacenado en un identificador printf("especificación de formato",identificador);

Impresión de dos o más datos almacenados en un identificador printf("especificaciones de formato",identif1,identifN);

Ejemplo: printf("%lf %lf %lf",ab,c); <-- Especif. de formato se separan de la forma en la que se quiera que se impriman Impresión de datos almacenados en un identificador y texto printf("Texto especificación de formato", identificador); Ejemplo: printf("Los valores son: %i,%i,%i",x,y,g);

Gestión dinámica de memoria-Arreglos dinámicos

Permite gestionar la memoria durante la ejecución del programa, depende de si hay memoria disponible o no • Se usan las librerías stdio.h y stdlib.h

Se usa la función malloc() • Los punteros se "convertirán" en arreglos al momento de ejecutar el programa, por lo que el tamaño del arreglo dependerá de la información que se proporcione mientras se ejecuta el programa

Los arreglos dinámicos se representan en diagrama de flujo igual que los arreglos estáticos, pero en este caso también se pregunta el tamaño y se almacena en algún identificador para ser utilizado posteriormente para crear el arreglo

a) Arreglos dinámicos unidimensionales

1. Incluir #incluide

2. Declarar un apuntador del tipo que será el arreglo tipo *nombreApuntador;

3. Obtener el tamaño del arreglo: ya sea pidiéndoselo al usuario o generando un número aleatorio

4. Cuando se sepa el tamaño se "convertirá" al puntero en un arreglo de ese tamaño de la siguiente forma:

nombrePuntero=(tipo*)malloc(tamaño*sizeof(tipo));

tipo: tipo del puntero declarado que se "convertirá" en el arreglo tamaño: variable entera donde se almacenó el número que indica el tamaño del arreglo

5. Emplear el arreglo dinámico unidimensional usando las mismas reglas que con los estáticos unidimensionales

6. Cuando ya NO se use más el arreglo, se debe liberar la memoria “convirtiendo” al arreglo otra vez en apuntador: free(nombreApuntador);

b) Arreglos dinámicos multidimensionales

1. Incluir #incluide

2. Declarar un apuntador del tipo que será el arreglo tipo **nombreApuntador;

3. Obtener el número de renglones y columnas del arreglo ya sea pidiéndoselo al usuario o generando números aleatorios

4. Cuando se sepa el tamaño se “convertirá” al puntero en un arreglo de ese tamaño de la siguiente forma: Primero se crean los

renglones: nombrePuntero=(tipo**)malloc(númRenglones*sizeof(tipo*));

tipo: tipo del apuntador que se “convertirá” en el arreglo Segundo se crean las columnas for(identif=0;identif