

Arreglos dinámicos

martes, 30 de noviembre de 2021 07:12 a. m.

`void free(void* ptr)`: Esta función requiere un apuntador devuelto por `malloc`, `calloc` o `realloc` debido a que esta función libera la memoria solicitada.

Definir un arreglo dinámico de tam NxM ---> Matriz --> `crearArregloNxM`

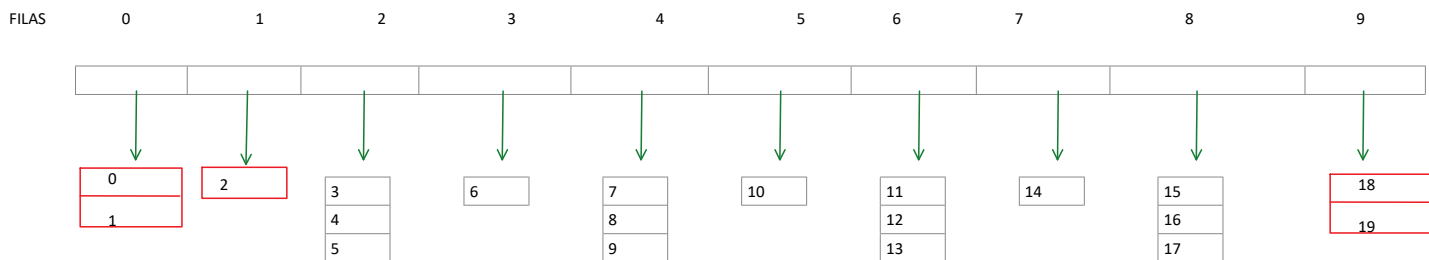
Columnas (M)

| | | | | |
|-----------|-----|-----|-----|-----|
| 0,0 int * | 0,1 | 0,2 | ... | 0,M |
| 1,0 int* | | | | |
| 2,0 int* | | | | |
| ... int* | | | | |
| N,0 int* | | | | N,M |

Filas (N)

`int arregloMatriz[N][M];`

Generar la siguiente estructura para el almacenamiento de 20 valores enteros



Ejemplo:

archivo.h

```
#include <stdio.h>
#include <stdlib.h>
void crearArregloNxM(int tamPrimeraDim, int tamSegundaDim);
int** crearArregloDin(int tamFilas);
void iniciaOperacion(int argc, char** argv);
```

archivo.c

```
#include "arregloD.h"
/*
Esta función genera una matriz de N filas (tamPrimeraDim) con M columnas cada
fila (tamSegundaDim)
@param tamPrimeraDim Valor del tamaño de la primera dimensión de la matriz (N
filas)
@param tamSegundaDim Valor del tamaño de la segunda dimensión de la matriz (M
columnas)
*/
void crearArregloNxM(int tamPrimeraDim, int tamSegundaDim){
//Se reserva memoria dinámica para la primera dimensión
int** arregloDosDim=(int**)malloc(sizeof(int)*tamPrimeraDim);
int indice, fila, columna;
if(arregloDosDim==NULL){
puts("Se termino la memoria, no se pueden reservar filas para la
matriz");
exit(1);
}else{
//Se reserva memoria para cada fila, se le asignan M
columnas(tamSegundaDim)
for(indice=0; indice<tamPrimeraDim; indice++){
arregloDosDim[indice]=(int*)malloc(sizeof(int)*tamSegundaDim);
if(arregloDosDim[indice]==NULL){
puts("Se termino la memoria, no se pueden reservar las columnas
para la matriz");
exit(1);
}
}
}
//Llenando la matriz
for(fila=0; fila<tamPrimeraDim; fila++){ //Recorre filas
for(columna=0; columna <tamSegundaDim; columna++){ //Recorre cada columna
arregloDosDim[fila][columna]=fila+columna;
}
}
//Imprimiendo la matriz
for(fila=0; fila<tamPrimeraDim; fila++){ //Recorre filas
for(columna=0; columna <tamSegundaDim; columna++){ //Recorre cada
columna
printf("El valor en la fila[%d], columna[%d] es %d.
\n",fila,columna,arregloDosDim[fila][columna]);
}
}
}

//Liberación de memoria dinámica: Liberando columnas
```

```

        for(indice=0; indice<tamPrimeraDim; indice++){
            free(arregloDosDim[indice]);
        }
        //Liberando la memoria de las filas
        free(arregloDosDim);
    }
    int** crearArregloDin(int tamFilas){
        int i;
        int** arregloEstructura=(int**) calloc(tamFilas,sizeof(int*));
        if(arregloEstructura==NULL){
            puts("Se termino la memoria, no se pueden reservar filas para la
matriz");
            exit(1);
        }
        for(i=0; i<tamFilas; i++){
            if(i==0 || i==9){
                arregloEstructura[i]= (int*) calloc(2, sizeof(int));
            } else if(){
            } else if(){

            }
        }
    }
}
void iniciaOperacion(int argc, char** argv){
    puts("Iniciando el programa");
    //Operaciones de la función main.
    crearArregloNxM(40000,40000);
    puts("Fin del programa");
}

```