PRACTICA DIRIGIDA 7

EJERCICIO 1

Escriba un programa en C++ a fin de crear un vector dinámico el cual debe llenarse ingresando enteros mayores a 0. Cada vez que se llene se debe redimensionar su tamaño por dos, y continuar ingresando datos hasta que el usuario crea conveniente, el vector inicial tiene tamaño 1. Finalmente imprimir todos los valores del tamaño del vector actual con el formato indicado líneas abajo.

Ejemplo 1:

```
Ingrese enteros, termina con -1:  5
Ingrese enteros, termina con -1:  6
Ingrese enteros, termina con -1: -1
val 0: 5
val 1: 6
```

Ejemplo 2

```
Ingrese enteros, termina con -1:  8
Ingrese enteros, termina con -1:  5
Ingrese enteros, termina con -1:  6
Ingrese enteros, termina con -1:  9
Ingrese enteros, termina con -1:  6
Ingrese enteros, termina con -1: -1
val 0: 8
val 1: 5
val 2: 6
val 3: 9
val 4: 6
```

EJERCICIO 2

Se tiene una matriz de tamaño de 4 filas y 3 columnas con las notas de los cuatro cursos que está cursando este semestre. Usando asignación dinámica de memoria en C++, se pide redimensionar la matriz para agregar una columna donde se pondrá el promedio redondeado y otra columna donde se ponga la mayor de las tres notas.

Para la siguiente matriz:

```
12 20 15
15 11 13
17 19 10
15 16 19
```

La salida sería:

```
12 20 15 16 20
15 11 13 13 15
17 19 10 15 19
15 16 19 17 19
```

EJERCICIO 3

Escriba un programa que lea un entero positivo n, aloje un rango dinámico de n*n enteros, y asigne un valor aleatorio: 0, 1, o 3 a cada elemento, luego recorra el bloque y dibuje un cielo estrellado de tamaño n x n con 3 astros: "* ", "+ ", " ". Su salida puede ser:

Mi cielo

Ingrese un entero positivo: 6

```
*     +     +
+     * * * +
* + * *  +
+  *     + +  *
      *  +  *
     * +  *   *
```

EJERCICIO 4

Escriba una función:

char *leer(void)

la cual lee del teclado una cadena de caracteres, la aloja en un bloque dinámico y retorna el apuntador al bloque.

Desde la main( ) llame a leer( ), escriba la cadena leída y su longitud. La salida puede ser:

Ingrese una cadena de caracteres: hola
hola, longitud: 4

Si va a usar C, debe usar la función realloc(...) cada vez que lee un carácter para aumentar en 1 char el tamaño del bloque.

EJERCICIO 5

Cree un arreglo dinámico para los 100 números enteros positivos.

EJERCICIO 6

Lea un cadena y usando asignación dinámica imprimir la cadena y su longitud.

EJERCICIO 7

Lea el código y puntaje del alumnado y usando asignación dinámica de memoria imprímalos.

EJERCICIO 8

Escribir un programa que utilice una matriz bidimensional para demostrar asignación de memoria dinámica en C ++. Asimismo muestre la suma por columna y fila de dicha matriz.

EJERCICIO 9

Escribir un programa que simule una matriz mat[2][3] en memoria dinámica. La función main define el bloque de memoria y llama a las funciones:

asignar() asigna valores a mat[2][3] = {0, 2, 4, 6, 8, 10}
escribirTranspuesta() imprime la transpuesta de mat.

Atento: solo imprime, no transpone a mat.

Ejemplo de salida:

Matriz transpuesta 3x2

0 6
2 8
4 10

EJERCICIO 10

Implemente en C++ las funciones:

a) *void mostrar(int filas, int columnas, int **p);*
que al pasarle un puntero a un arreglo (de longitud 'filas') de punteros a vectores de 'columnas' enteros, lo imprima como una matriz de 'filas' filas y 'columnas' columnas.
b) *int **generar(int filas, int columnas);*
que genere un arreglo (de longitud 'filas') de punteros a vectores de 'columnas' enteros formado por números aleatorios del 0 al 9.
c) *int **transponer(int filas, int columnas, int **p);*
que al pasarle un puntero a un arreglo (de longitud 'filas') de punteros a vectores de 'columnas' enteros, genere un arreglo (de longitud 'columnas') de punteros a vectores de 'filas' enteros donde esta nueva 'matriz' sea la transpuesta de la 'matriz' p
d) *void liberar(int filas, int **p);*
libere la memoria asignada para el arreglo (de longitud 'filas') de punteros a vectores de enteros y tb libere la memoria asignada a estos últimos vectores de enteros.

Finalmente en la función principal ejecute:

```
int **matriz, **transpuesta; srand(17);
matriz = generar(2,3); mostrar(2,3,matriz);
transpuesta = transponer(2,3,matriz); mostrar(3,2,transpuesta);
```

y libere toda la memoria asignada dinámicamente.

EJERCICIO 11

Defina las siguientes constantes simbólicas:

```
#define COLUMNAS 100
#define ARTICULOS 3
```

y la siguiente matriz:

```
char derechos[ARTICULOS][COLUMNAS] = {
"Todos los seres humanos nacen libres e iguales en dignidad y derechos.",
"Todo individuo tiene derecho a la vida, a la libertad y a la seguridad de su persona.",
"Nadie podra ser arbitrariamente detenido, preso ni desterrado."
};
```

Implemente las funciones:

a) char **generar(int items, char (*origen)[COLUMNAS]);

que al pasarle una copia de la constante 'derechos' genere un arreglo (de longitud 'items') de punteros a caracteres donde cada uno de estos punteros apunte a una cadena que sea la copia de una (de forma aleatoria) de las tres cadenas de 'derechos'.

b) void mostrar(int items, char **p);

que muestre cada una de las cadenas que son apuntadas por los punteros del arreglo de punteros (de longitud 'items').

c) void liberar(int items, char **p);

libere la memoria asignada para el arreglo (de longitud 'items') de punteros a cadenas y tb libere la memoria asignada a estas últimas cadenas.

Finalmente en la función principal ejecute:

```
  char **recital;
srand(17);
recital = generar(5,derechos);
mostrar(5,recital);
```

y libere toda la memoria asignada dinámicamente.

EJERCICIO 12

Escriba un programa que solicite ingresar temperaturas diarias una por una, termina cuando temperatura = 0, luego de ingresar una temperatura se muestran todas las temperaturas ingresadas anteriormente ordenadas ascendentemente.

EJERCICIO 13

Utilice alojamiento dinámico para una matriz A[m][n][k], lea datos e imprímalos.

EJERCICIO 14

Utilice alojamiento dinámico y un puntero a puntero a puntero (***p) para leer una matriz p[m][n][k] y luego imprímala.

EJERCICIO 15

Copie el programa del problema anterior y recorra la matriz con apuntadores para cada dimensión.