**ÚLTIMA ACTUALIZACIÓN: 19/06/2023 18:00**

[**EJEMPLO\_01**](https://pythontutor.com/render.html#code=%23include%3Cstdio.h%3E%0A%23include%3Cstdlib.h%3E%0A/*-----%20----%20A%20R%20B%20O%20L%20---%20%20-----------------*/%0Astruct%20sNodo%20%7B%0A%20%20%20%20int%20valor%3B%0A%20%20%20%20struct%20sNodo%20*sig%3B%0A%7D%3B%0Atypedef%20struct%20sNodo%20*tNodo%3B%0A%0Astruct%20sNodoBin%7B%20//%20ARBOL%0A%20%20%20%20int%20valor%3B%0A%20%20%20%20struct%20sNodoBin*%20izq%3B%0A%20%20%20%20struct%20sNodoBin*%20der%3B%0A%7D%3B%0Atypedef%20struct%20sNodoBin%20*tNodoBin%3B%0A%0A%0Avoid%20insertarEnArbol%28tNodoBin*%20arbol,%20int%20valor%29%7B%0A%20%20%20%20if%20%28*arbol%20%3D%3D%20NULL%29%7B%0A%20%20%20%20%20%20%20%20*arbol%20%3D%20%28tNodoBin%29malloc%28sizeof%28struct%20sNodoBin%29%29%3B%0A%20%20%20%20%20%20%20%20%28*arbol%29-%3Evalor%20%3D%20valor%3B%0A%20%20%20%20%20%20%20%20%28*arbol%29-%3Eizq%20%3D%20NULL%3B%0A%20%20%20%20%20%20%20%20%28*arbol%29-%3Eder%20%3D%20NULL%3B%0A%20%20%20%20%7D%0A%20%20%20%20else%20%7B//RECURSIVA%0A%20%20%20%20%20%20%20%20if%20%28valor%20%3C%20%28*arbol%29-%3Evalor%29%0A%20%20%20%20%20%20%20%20%20%20%20%20insertarEnArbol%28%26%20%28*arbol%29-%3Eizq,%20valor%29%3B%0A%20%20%20%20%20%20%20%20else%0A%20%20%20%20%20%20%20%20%20%20%20%20insertarEnArbol%28%26%20%28*arbol%29-%3Eder,%20valor%29%3B%0A%20%20%20%20%7D%0A%7D%0A%20%20%20%20%0Avoid%20inorderImprimir%28tNodoBin%20arbol%29%7B%0A%20%20%20%20if%20%28arbol%20!%3D%20NULL%29%7B%0A%20%20%20%20%20%20%20%20inorderImprimir%28arbol-%3Eizq%29%3B%20%20%20%20//%20IZQ%0A%20%20%20%20%20%20%20%20printf%28%22%25d%20-%20%22,%20arbol-%3Evalor%29%3B%20%20//%20PROCESO%20EL%20NODO%0A%20%20%20%20%20%20%20%20inorderImprimir%28arbol-%3Eder%29%3B%20%20%20%20//%20DER%0A%20%20%20%20%7D%0A%7D%0A%0Aint%20main%28%20%29%20%7B%0A%20%20%20%20tNodoBin%20arbol%3DNULL%3B%0A%20%20%20%20%0A%20%20%20%20printf%28%22%5Cn-------INSERTANDO%20...%20-------------%22%29%3B%0A%20%20%20%20insertarEnArbol%28%26arbol,35%29%3B%0A%20%20%20%20insertarEnArbol%28%26arbol,45%29%3B%0A%20%20%20%20insertarEnArbol%28%26arbol,30%29%3B%0A%20%20%20%20insertarEnArbol%28%26arbol,20%29%3B%0A%20%20%20%20insertarEnArbol%28%26arbol,40%29%3B%0A%20%20%20%20insertarEnArbol%28%26arbol,38%29%3B%0A%20%20%20%20insertarEnArbol%28%26arbol,41%29%3B%0A%20%20%20%20%0A%20%20%20%20printf%28%22%5Cn--------inorderImprimir------------%5Cn%22%29%3B%0A%20%20%20%20inorderImprimir%28arbol%29%3B%0A%20%20%20%20%0A%20%20%20%20return%200%3B%0A%7D&cumulative=false&curInstr=222&heapPrimitives=nevernest&mode=display&origin=opt-frontend.js&py=c_gcc9.3.0&rawInputLstJSON=%5B%5D&textReferences=false)

| **#include <stdio.h>** **#include <stdlib.h>** */\*\*  A R B O L \*/*  */\*----- ---- A R B O L --- -----------------\*/* **struct** **sNodo** {  **int** valor;  **struct** **sNodo** \***sig**; }; **typedef** **struct** **sNodo** \***tNodo**;  **struct** **sNodoBin**{ *// ARBOL*  **int** valor;  **struct** **sNodoBin**\* **izq**;  **struct** **sNodoBin**\* **der**; }; **typedef** **struct** **sNodoBin** \***tNodoBin**;  */\*----- ---- C O L A --- -----------------\*/* **struct** **s\_nodo** {  tNodoBin valor;  **struct** **s\_nodo**\* **sig**; }; **typedef** **struct** **s\_nodo**\* **t\_nodo**;  */\*----- ---- C O L A --- -----------------\*/* */\* \*\* con dos puntero: Frente y Final \*\* \*/* **struct** **s\_cola**{  t\_nodo frente;  t\_nodo final; }; **typedef** **struct** **s\_cola** **t\_cola**;   *//ARBOL MODIFICACI'N DEL ARBOL* **void** **insertarEnArbol**(tNodoBin\* arbol, **int** valor); **void** **eliminarArbol**(tNodoBin \*arbol); *// ELIMINA ARBOL COMPLETO* **void** **buscarYBorrarEnArbol**(tNodoBin \*arbol, **int** valor); *// ELIMINA UN SUBARBOL*  *//ARBOL LECTURA - RECORRIDOS* **void** **preorderImprimir**(tNodoBin arbol); *//(\*)* **void** **inorderImprimir**(tNodoBin arbol); *//(\*)* **void** **postorderImprimir**(tNodoBin arbol); *//(\*)* **void** **postorderImprimir2**(tNodoBin arbol); **void** **porNivelesImprimir**(tNodoBin arbol);  *//ARBOL BUSQUEDA* **int** **estaValorEnArbol**(tNodoBin arbol, **int** valor); tNodoBin **direccionValorEnArbol**(tNodoBin arbol, **int** valor);  *//ARBOL* **int** **sumar**(tNodoBin arbol, **int** min, **int** max); **int** **sumarUnNivel**(tNodoBin arbol,**int** nivel);   *//COLA* **int** **desencolar**(tNodo\* nodo); **void** **encolar**(tNodo\*, **unsigned** **int**);  **int** **main**(){  tNodoBin arbol=NULL;   printf("\n-------INSERTANDO ... -------------");  insertarEnArbol(&arbol,35);  insertarEnArbol(&arbol,45);  insertarEnArbol(&arbol,30);  insertarEnArbol(&arbol,20);  insertarEnArbol(&arbol,15);  insertarEnArbol(&arbol,40);  insertarEnArbol(&arbol,38);  insertarEnArbol(&arbol,41);  insertarEnArbol(&arbol,55);  insertarEnArbol(&arbol,21);    printf("\n--------preorderImprimir------------\n");  preorderImprimir(arbol);  printf("\n--------inorderImprimir------------\n");  inorderImprimir(arbol);  printf("\n--------postorderImprimir------------\n");  postorderImprimir(arbol);  printf("\n--------postorderImprimir (con direcciones de memoria)-------\n");  postorderImprimir2(arbol);    printf("\n--------POr Niveles Imrimir-------------------\n");  porNivelesImprimir(arbol);    printf("\n--------Sumatoria de un Nivel -------------------\n");  printf("%d\n\n",sumarUnNivel(arbol, 3));   printf("\n---valor 20 ------Esta =1, NO esta=0------------\n");  printf("\n\_\_%d\_\_\n",estaValorEnArbol(arbol, 20));   printf("\n---direccion del valor 20 ------- - - - - - - - - - -------\n");  printf("\n\_\_%p\_\_\n",direccionValorEnArbol(arbol, 20));   printf("\n-----------------------------------------------------------------------------------\n");  printf("\nBusco el valor 20 y lo elimino del arbol y el subarbol descendiente del nodo 30 -\n");  printf("Imprimo inorder para ver la eliminacion-\n");  buscarYBorrarEnArbol(&arbol,20);  inorderImprimir(arbol);   **return** 0; }  **void** **insertarEnArbol**(tNodoBin\* arbol, **int** valor){  **if** (\*arbol == NULL){  \*arbol = (tNodoBin)malloc(**sizeof**(struct sNodoBin));  (\*arbol)->valor = valor;  (\*arbol)->izq = NULL;  (\*arbol)->der = NULL;  }  **else** {*//RECURSIVA*  **if** (valor < (\*arbol)->valor)  insertarEnArbol(& (\*arbol)->izq, valor);  **else**  insertarEnArbol(& (\*arbol)->der, valor);  } }   **void** **preorderImprimir**(tNodoBin arbol){  **if** (arbol != NULL){  printf("%d - ", arbol->valor); *// PROCESO EL NODO*  preorderImprimir(arbol->izq); *// IZQ*  preorderImprimir(arbol->der); *// DER*  } }  **void** **inorderImprimir**(tNodoBin arbol){  **if** (arbol != NULL){  inorderImprimir(arbol->izq); *// IZQ*  printf("%d - ", arbol->valor); *// PROCESO EL NODO*  inorderImprimir(arbol->der); *// DER*  } }  **void** **postorderImprimir**(tNodoBin arbol){  **if** (arbol != NULL){  postorderImprimir(arbol->izq); *// IZQ*  postorderImprimir(arbol->der); *// DER*  printf("%d - ", arbol->valor); *// PROCESO EL NODO*  } }   **void** **postorderImprimir2**(tNodoBin arbol){  **if** (arbol != NULL){  postorderImprimir2(arbol->izq);  postorderImprimir2(arbol->der);  printf("(%p)<-(%p)%d->(%p) \n",arbol->izq ,arbol,arbol->valor,arbol->der);  } }   **void** **queue**(t\_cola\* cola, tNodoBin valor){  t\_nodo aux = (t\_nodo) malloc(**sizeof**(struct s\_nodo));  aux->valor = valor;  aux->sig = NULL;  **if** (cola->frente == NULL && cola->final == NULL){  cola->frente = aux;  cola->final = aux;  }  **else** {  cola->final->sig = aux;  cola->final = aux;  } }  tNodoBin **dequeue**(t\_cola\* cola){  tNodoBin valor;  t\_nodo aux = cola->frente;  cola->frente = cola->frente->sig;  valor = aux->valor;  free(aux);  **if** (cola->frente == NULL) {  cola->final = NULL;  }  **return** valor; }  **void** **porNivelesImprimir**(tNodoBin arbol){  *// Imprime los valores de cada nodo*  *// ordenado por niveles*    t\_cola cola = {NULL, NULL};  tNodoBin nodo\_aux = NULL;    queue(&cola, arbol);  **while** (cola.frente != NULL)  {  nodo\_aux = dequeue(&cola);  **if** (nodo\_aux->izq != NULL){  queue(&cola, nodo\_aux->izq);   }   **if** (nodo\_aux->der != NULL){  queue(&cola, nodo\_aux->der);   }  printf("%d -", nodo\_aux->valor);  }  printf("\n"); }   **int** **sumarUnNivel**(tNodoBin arbol,**int** nivel){  **int** res=0;  **if** (arbol != NULL){  **if** (nivel==0){  res= arbol->valor;  *//printf("%d - ", arbol->valor);*   }  res=res+ sumarUnNivel(arbol->izq,nivel-1);   res=res+ sumarUnNivel(arbol->der,nivel-1);   }   **return** res; }  **int** **estaValorEnArbol**(tNodoBin arbol, **int** valor){  *// Retorna 1 si valor se encuentra en arbol*  *// Retorna 0 si valor se NO encuentra en arbol*  **if** (arbol == NULL)  **return** 0;  **else**  {  **if** (valor == (arbol-> valor))  **return** 1;  **else**  **if** (valor < arbol->valor)  **return** estaValorEnArbol(arbol->izq, valor);  **else**  **return** estaValorEnArbol(arbol->der, valor);  } }   tNodoBin **direccionValorEnArbol**(tNodoBin arbol, **int** valor){  *// Retorna 'la direcci'n del nodo' donde se encuentra valor dentro del arbol*  *// Retorna 'NULL' si valor se NO encuentra en arbol*  **if** (arbol == NULL)  **return** NULL;  **else**  {  **if** (valor == arbol-> valor)  **return** arbol;  **else**  **if** (valor < arbol->valor)  **return** direccionValorEnArbol(arbol->izq, valor);  **else**  **return** direccionValorEnArbol(arbol->der, valor);  } }    **void** **buscarYBorrarEnArbol**(tNodoBin \*arbol, **int** valor){  *// Si valor se encuentra en arbol, entoces borra*  *// todo el sub arbol que cuelga del padre del nodo*  *// donde se encuentra valor.*  **if** (\*arbol != NULL)  {  **if** (valor < (\*arbol)->valor){  buscarYBorrarEnArbol(& (\*arbol)->izq, valor);  }  **else** **if** (valor > (\*arbol)->valor){  buscarYBorrarEnArbol(& (\*arbol)->der, valor);  }  **else**{  eliminarArbol(arbol);  }  } }   **void** **eliminarArbol**(tNodoBin \*arbol){  */\*elimina el arbol por completo\*/*  **if** ( (\*arbol) != NULL){  eliminarArbol(&((\*arbol)->izq)); *// RECORRIDO A*   eliminarArbol(&((\*arbol)->der)); *// RECORRIDO A DERECHA*   free(\*arbol); *// PROCESO*  \*arbol=NULL; *// PROCESO*  } }   **int** **sumar**(tNodoBin arbol, **int** min, **int** max){  *// Retorna la sumatoria de todos los nodos cuyo valores*   *// se encuentren entre min y max*  **int** sum=0;  **if** (arbol == NULL) {  **return** 0;  }  **else** {  *//sum= sum+arbol->valor+sumar(arbol->izq,min,max)+sumar(arbol->der,min,max);*  **if**((min<=arbol->valor)&&(max>=arbol->valor)){  sum= sum+arbol->valor+sumar(arbol->izq,min,max)+sumar(arbol->der,min,max);  }  **else**{  sum= sum+sumar(arbol->izq,min,max)+sumar(arbol->der,min,max);  }  **return** sum;  } } |
| --- |

**SALIDA**

| **P I L A S**  -------INSERTANDO ... -------------  --------preorderImprimir------------  35 - 30 - 20 - 15 - 21 - 45 - 40 - 38 - 41 - 55 -  --------inorderImprimir------------  15 - 20 - 21 - 30 - 35 - 38 - 40 - 41 - 45 - 55 -  --------postorderImprimir------------  15 - 21 - 20 - 30 - 38 - 41 - 40 - 55 - 45 - 35 -  --------postorderImprimir (con direcciones de memoria)------------  (00000000)<-(00AF1468)15->(00000000)  (00000000)<-(00AF14E0)21->(00000000)  (00AF1468)<-(00AF19E8)20->(00AF14E0)  (00AF19E8)<-(00AF19D0)30->(00000000)  (00000000)<-(00AF1498)38->(00000000)  (00000000)<-(00AF14B0)41->(00000000)  (00AF1498)<-(00AF1480)40->(00AF14B0)  (00000000)<-(00AF14C8)55->(00000000)  (00AF1480)<-(00AF19B8)45->(00AF14C8)  (00AF19D0)<-(00AF19A0)35->(00AF19B8)  --------POr Niveles Imrimir-------------------  35 -30 -45 -20 -40 -55 -15 -21 -38 -41 -  --------Sumatoria de un Nivel -------------------  115  ---valor 20 ------Esta =1, NO esta=0------------  \_\_1\_\_  ---direccion del valor 20 ------- - - - - - - - - - ------------  \_\_00AF19E8\_\_  -----------------------------------------------------------------------------------  Busco el valor 20 y lo elimino del arbol y el subarbol descendiente del nodo 30 -  Imprimo inorder para ver la eliminacion-  30 - 35 - 38 - 40 - 41 - 45 - 55 -  << El programa ha finalizado: código de salida: 0 >> << Presione enter para cerrar esta ventana >> |
| --- |