#

include **<stdio.h>**#include **<stdlib.h>***//-----------------------Lista--------------------*Lista \***crearLista**(){*//lista vacia* Lista\*lis;  
 **if**(lis = (Lista \*)malloc(**sizeof**(Lista))){  
 lis -> ***ini*** = **NULL**;lis ->***fin*** = **NULL**;lis -> ***tam*** =0;  
 }**else**{printf(**"no se pudo asignar memoria a la lista"**);}**return** lis;};  
**void insertarLista**(Lista \*plis, **int** pval){  
 Nodo \*newNodo;**if**(newNodo = (Nodo\*)malloc(**sizeof**(Nodo))){  
 newNodo -> ***valor*** =pval;newNodo -> ***sig*** = **NULL**;  
 **if** (plis->***tam*** =0){plis->***ini*** = newNodo;plis->***fin*** =newNodo;  
 }**else**{newNodo ->***sig***=plis->***ini***;plis->***ini*** =newNodo;}plis->***tam***++;  
 }**else**{printf(**"memoria no se pudo asignar"**);}}}  
**void recorerL**(Lista \*Plista){Nodo \* aux = Plista->***ini***;  
 aux =Plista->***ini***;printf(**"los valores de la lista son: \n"**)  
 **while** (aux!=**NULL**){printf(**"%d\n"**,aux->***valor***);aux =aux->***sig***;}  
**int eliminarNodo**(Lista \*plis, **int** pval){  
 Nodo \*aux = plis ->***ini***;Nodo \*auxe;**int** eliminado =0;  
 **if** (pval ==aux->***valor***){auxe =aux;plis ->***ini*** =aux->***sig***;plis ->***tam***--;  
 auxe ->***sig*** =**NULL**;eliminado =1;free(auxe);}  
 **else**{**while** (aux->***sig***!=**NULL** && eliminado !=1)  
 **if**(aux->***sig***->***valor***!=pval){aux=aux->***sig***;}**else**{  
 aux =aux->***sig***;aux ->***sig***= aux->***sig***->***sig***;plis ->***tam***--;  
 aux->***sig***=**NULL**;eliminado=1;free(auxe);}}**return** eliminado;}  
**void destruir**L(Lista \*lis){Nodo \*aux; **while** (lis-> ***ini***!=**NULL**){  
 aux = lis->***ini***;lis ->***ini*** =lis->***ini***->***sig***;free(lis);free(lis);}}  
*//-----------------------Pila--------------------*Info **\*crerInf**(**int** pval){ Info \*newInfo;**if**(newInfo =(Info\*)malloc(**sizeof**(Info))){newInfo -> ***valor*** =pval;}**else**{printf(**"no se puedo asignar memiroa"**);}  
 }**return** newInfo;}

Pila**\*crearInf**(){Pila\*p;**if**(p=(Pila\*)malloc(**sizeof**(Pila))){p->***tope*** =**NULL**;  
 p-> ***tam***=0;}**else**{printf(**"no se puedo asignar memoria"**);}**return** p;}  
**void Apilar**(Pila \*p,**int** pval){info\*pinfo =crearInf(paval);Nodo\*newNodo;**if**(newNodo=(Nodo\*)malloc(**sizeof**(Nodo))){newNodo->***elem***=pinfo;  
 newNodo->***sig*** =p->***tope***;p->***tope***=newNodo;p->***elemen***++;}**else** {printf(**"no se pudo asignar memoria"**);}free(newNodo);}  
Info \*desapil(Pila\*p){Nodo \*aux;Info\*inf;

**if**(esVacia(p)){ printf(**"la pila no se puede desapilar esta vacia"**);}**else**{aux =p->***tope***;inf =aux->***elem***;p->***tope*** =aux->***tope***;p->***tam***--;}**return** inf;}  
**void** destruir(Pila \*p){Nodo\*aux;**while**(esVacia(p)){  
 p->***tope***=p->***sig***;free(aux);}free(aux);}  
Nodo \*nodoTope(Pila \*p){ **return** p->***tope***;}  
bool esVacia(Pila \*p){**if**(p->***tope***=**NULL**){**return** true;}**else**{ **return** false;};}  
*//-----------------------Colas--------------------*Cola **\*crearCola**(){Cola \*tre;**if** (tre = (Cola\*)malloc(**sizeof**(Cola))){  
 tre -> ***ini*** = tre->***fin*** = **NULL**;tre ->***tamano*** =0;  
 }**else**{printf(**"no se pudo asignar memoria"**);}**return** tre;}Info \*crearInfo(**int** pinfo){Info\*newInfo;**if**(newInfo = (Info \*)malloc(**sizeof**(Info))){  
 newInfo->***elemen*** =pinfo;}**else**{printf(**"no se pudo asignar memoria"**);} **return** newInfo;}  
**void encolar(**Cola \*ptam,**int** pval){Info \*mac =rearInfo(pval);Nodo\*newNodo;**if**(newNodo=(Nodo\*)malloc(**sizeof**(Nodo))){**if**(estaVacia(ptam)){ ptam ->***ini*** =newNodo;} **else**{ptam->***fin***->***sig***=newNodo;} ptam->***fin***=newNodo;newNodo ->***sig*** =**NULL**;newNodo ->***elemen*** =mac;  
 ptam ->***tamano***++;}**else**{printf(**"no se pudo asignar memoria"**);}  
 free(newNodo);}  
**int tamCola**(Cola \*tam){**if** (tam ->***tamano***!=**NULL**){  
 printf(**"tamaño de la cola es: %d"**,tam->***tamano***);}  
 **else**{printf(**"la cola esta vacia"**);}**return** tam->***tamano***;}  
bool **estaVacia**(Cola \*ptam){**if**(ptam -> ==**NULL**){**return** true;}**else**{**return** false;}