typedef int TipoDato;

typedef struct unnodo {

TipoDato el;

struct unnodo \*sig;

} Nodo;

typedef Nodo Pila;

Nodo\* NuevoNodo(TipoDato elemento);

void VaciaP(Pila\*\* P);

void AnadeP(Pila\*\* P,TipoDato elemento);

void BorrarP(Pila\*\* P);

TipoDato PrimeroP(Pila \*P);

int EsVaciaP(Pila \*P);

void Pop(Pila\*\* P,TipoDato elemento);

TipoDato Push(Pila \*\*P);

Nodo\* NuevoNodo(TipoDato elemento)

{

Nodo \*a ;

a = (Nodo\*)malloc(sizeof(Nodo));

a -> el = elemento;

a -> sig = NULL;

return a;

}

void VaciaP(Pila\*\* P)

{

\*P = NULL;

}

void AnadeP(Pila\*\* P, TipoDato elemento)

{

Nodo \* nn;

nn = NuevoNodo(elemento);

nn->sig = (\*P);

\*P = nn;

}

void Push(Pila\*\* P,TipoDato elemento)

{

AnadeP(P, elemento);

}

TipoDato Pop(Pila \*\*P)

{

TipoDato Aux;

Pila \*nn;

if (EsVaciaP(\*P))

{

puts(“Se intenta sacar un elemento en pila vacía”);

exit (1);

}

Aux = (\*P)->el;

nn = \*P;

\*P = nn->sig;

free(nn);

return Aux;

}

TipoDato PrimeroP(Pila \*P)

{

TipoDato Aux;

if (EsVaciaP(P))

{

puts(“Se intenta sacar un elemento en pila vacía”);

exit (1);

}

Aux = P->el;

return Aux;

}

void BorrarP(Pila\*\* P)

{

Pila \*nn;

if (EsVaciaP(\*P))

{

puts(“Se intenta sacar un elemento en pila vacía”);

exit (1);

}

nn =(\*P);

(\*P)= nn->sig;

free(nn);

}

int EsVaciaP(Pila \*P)

{

return P == NULL;

}