TAD PILA

1.- ESPECIFICACIÓN:

tad Pila(telemento) { Pila está formada por elementos de tipo telemento}

Especificación de las operaciones:

```
accion iniciarPila( sal  P : Pila )
{ Inicia P como una pila vacía }

accion apilar( e/s  P : Pila, ent  d : telemento )
{ Apila en P el elemento d }

funcion pilaVacia( P : Pila ) devuelve booleano
{ Si P está vacía devuelve VERDAD y FALSO en otro caso }

funcion cima( P : Pila ) devuelve telemento
{ Devuelve el elemento más reciente de la pila P y no
modifica la pila }

accion desapilar( e/s  P : Pila )
{ Modifica la pila P eliminando el último elemento
apilado }
```

2a.- IMPLEMENTACIÓN ESTÁTICA:

- ***REPRESENTACIÓN:***

```
constante
     MAXPILA=....;  {representa el tamaño máximo de la pila}

tipo
     tvector = vector[1..MAXPILA]  de  telemento
     Pila = registro
          datos : tvector
          indCima : entero
     freg;
```

• ***INTERPRETACIÓN:***

`indCima` representa el número de elementos de la pila, los cuales se encuentran almacenados en las `indCima` primeras componentes. La cima se encuentra en la componente `indCima`.

• ***IMPLEMENTACIÓN DE LAS OPERACIONES:***

```
accion iniciarPila( sal  P : Pila )
{ Inicia P como una pila vacía }

principio
     P.indCima ← 0
fin


accion apilar( e/s  P : Pila, ent  d : telemento )
{ Apila en P el elemento d }

principio
     si  P.indCima<MAXPILA  entonces
          P.indCima ← P.indCima+1
          P.datos[P.indCima] ← d
     fsi
fin



funcion pilaVacia( P : Pila ) devuelve booleano
{Si P está vacía devuelve VERDAD y FALSO en otro caso}

principio
     devuelve( P.indCima = 0 )
fin


funcion cima( P : Pila ) devuelve telemento
{ Devuelve el elemento más reciente de la pila P y no
  modifica la pila }

principio
     devuelve( P.datos[P.indCima] )
fin


accion desapilar( e/s  P : Pila )
{ Modifica la pila P eliminando el último elemento
  apilado }

principio
     P.indCima ← P.indCima -1
Fin
```

Nota: esta implementación es válida para pilas de cómo mucho MAXPILA elementos

2b.- IMPLEMENTACIÓN DINÁMICA:

- ***REPRESENTACIÓN:***

```
tipo
     Nodo = registro
          dato : telemento
          sig : puntero a Nodo
     freg;

     Pila = puntero a Nodo
```

- ***INTERPRETACIÓN:***

Pila es un puntero a un nodo que contiene la cima y un puntero que apunta al nodo que contiene el elemento anterior. El puntero del nodo correspondiente al primer elemento apilado apunta a `NULL`. Si la pila está vacía el puntero apunta a `NULL`.

- ***IMPLEMENTACIÓN DE LAS OPERACIONES:***

```
accion iniciarPila( sal  P : Pila )
{ Inicia P como una pila vacía }

principio
 P ← NULL
fin



accion apilar( e/s  P : Pila, ent  d : telemento )
{ Apila en P el elemento d }

variables
 nuevo : puntero a Nodo
principio
 reservar(nuevo)
 si nuevo ≠ NULL entonces
      dest(nuevo).dato ← d
      dest(nuevo).sig ← P
      P ← nuevo
 fsi
fin
```

```
funcion pilaVacia( P : Pila ) devuelve booleano
{Si P está vacía devuelve VERDAD y FALSO en otro caso }

principio
 devuelve( P = NULL )
fin
```

```
funcion cima( P : Pila ) devuelve telemento
{ Devuelve el elemento más reciente de la pila P y no
 modifica la pila }

principio
 devuelve( dest(P).dato )
fin
```

```
accion desapilar( e/s  P : Pila )
{Modifica la pila P eliminando el último elemento
 apilado }

variables
 aux : puntero a Nodo
principio
 aux ← P
 P ← dest(aux).sig
 liberar(aux)
fin
```

TAD COLA

1.- ESPECIFICACIÓN:

tad Cola(telemento) { Cola está formada por elementos de tipo telemento}

Especificación de las operaciones:

```
accion iniciarCola( sal  C : Cola )
{ Inicia C como una cola vacía }

accion añadir( e/s  C : Cola, ent  d : telemento )
{ Añade a C el elemento d }

funcion colaVacia( C : Cola ) devuelve booleano
{Si C está vacía devuelve VERDAD y FALSO en otro caso}

funcion primero( C : Cola ) devuelve telemento
{ Devuelve el elemento más antiguo de la cola C y no
  modifica la cola }

accion eliminar( e/s  C : Cola )
{ Modifica la cola C eliminando el elemento más
  antiguo}
```

2a.- IMPLEMENTACIÓN ESTÁTICA:

- ***REPRESENTACIÓN:***

```
constante
     MAXCOLA=....;  {representa el tamaño máximo de la cola}

tipo
     tvector = vector[1..MAXCOLA]  de  telemento
     Cola = registro
          datos : tvector
          primero, ultimo  : entero
          num : entero
     freg
```

• ***INTERPRETACIÓN:***

`num` representa el número de elementos de la cola, los cuales se encuentran almacenados entre las componentes `primero` y `ultimo`, estando en la componente `primero` el elemento más antiguo de la cola y en último el más reciente.

• ***IMPLEMENTACIÓN DE LAS OPERACIONES:***

```
accion iniciarCola( sal  C : Cola )
{ Inicia C como una cola vacía }
principio
     C.num ← 0
     C.primero ← 1
     C.ultimo ← 0
fin


accion añadir( e/s  C : Cola, ent  d : telemento )
{ Añade a C el elemento d }

principio
     si  C.num<MAXCOLA  entonces
          C.num ← C.num + 1
          C.ultimo ← suma_uno(C.ultimo)
          C.datos[C.ultimo] ← d
     fsi
fin


funcion colaVacia( C : Cola ) devuelve booleano
{ Si C está vacía devuelve VERDAD y FALSO en otro caso }

principio
     devuelve( C.num = 0 )
fin


funcion primero( C : Cola ) devuelve telemento
{ Devuelve el elemento más antiguo de la cola C y no
  modifica la cola }

principio
     devuelve( C.datos[C.primero] )
fin


accion eliminar( e/s  C : Cola )
{ Modifica la cola C eliminando el elemento más antiguo }

principio
     C.num ← C.num - 1
     C.primero ← suma_uno(C.primero)
Fin
```

```
funcion suma_uno( n : entero ) devuelve entero
{ Suma uno a la posición n en el sentido circular de las
  agujas del reloj }

principio
     devuelve( (n MOD MAXCOLA) + 1 )
fin
```

Nota: esta implementación es válida para colas de cómo mucho MAXCOLA elementos

2b.- IMPLEMENTACIÓN DINÁMICA:

- ***REPRESENTACIÓN:***

```
tipo
     Nodo = registro
          dato : telemento
          sig : puntero a Nodo
     freg;

     Cola = registro
          primero : puntero a Nodo
          ultimo : puntero a Nodo
     freg;
```

- ***INTERPRETACIÓN:***

Cola es un registro con dos punteros de forma que el puntero `primero` apunta a un nodo que contiene el elemento más antiguo de la cola y un puntero que apunta al nodo que contiene el elemento que llegó a la cola después. El puntero del nodo correspondiente al elemento más reciente de la cola apunta a `NULL`. El puntero `ultimo` apunta al nodo en el que está el elemento más reciente de la cola.

- ***IMPLEMENTACIÓN DE LAS OPERACIONES:***

```
accion iniciarCola( sal  C : Cola )
{ Inicia C como una cola vacía }

principio
     C.primero ← NULL
     C.ultimo ← NULL
Fin
```

```
accion añadir( e/s  C : Cola, ent  d : telemento )
{ Añade a C el elemento d }

variables
     nuevo : puntero a Nodo
principio
     nuevo ← reservar(Nodo)
     si nuevo ≠ NULL entonces
          dest(nuevo).dato ← d
          dest(nuevo).sig ← NULL
          si   C.primero = NULL  entonces
               C.primero ← nuevo
          sino
               dest(C.ultimo).sig ← nuevo
          fsi
          C.ultimo ← nuevo
     fsi
fin



funcion colaVacia( C : Cola ) devuelve booleano
{Si C está vacía devuelve VERDAD y FALSO en otro caso}

principio
     devuelve( C.primero = NULL )
fin



funcion primero( C : Cola ) devuelve telemento
{ Devuelve el elemento más antiguo de la cola C y no
  modifica la cola }

principio
     devuelve( dest(C.primero).dato )
fin




accion eliminar( e/s  C : Cola )
{ Modifica la cola C eliminando el elemento más antiguo }

variables
     aux : puntero a Nodo
principio
     aux ← C.primero
     C.primero ← dest(C.primero).sig
     liberar(aux)
     si C.primero = NULL entonces
          C.ultimo ← NULL
     fsi

fin
```