Tema 8. El TDA Cola

8.1. Definición y operaciones del TDA Cola
8.2. Implementación con listas enlazadas
8.3. Implementación con vectores

Bibliografía:

- Weiss
- Aho, Hopcroft y Ullman

8.1. Definición y operaciones del TDA Cola

Tipo de dato abstracto en el que:

- Las inserciones se hacen por un extremo y
- Los borrados se hacen por el otro extremo.

Ejemplos: cola del cine, de un supermercado, de una impresora

Son ampliamente utilizados en el campo de la informática: para gestionar recursos.

Ejemplo: Simular el funcionamiento de una cola para saber cuántas cajas son necesarias en un supermercado con distintas condiciones de número de clientes y tiempos medios de clientes.

Se conocen como estructuras FIFO (*First In First Out*).

Las acciones básicas son Encolar (*Queue*) y Desencolar (*Dequeue*).

OPERACIONES TDA COLA

- **Crear_cola (C: cola, ok:lógico)**
- **Borrar_cola (C: cola, ok:lógico)**
- **Vacía? (C: cola, resp:lógico)**
- **Llena? (C: cola, resp:lógico)**
- **Queue (C: cola, E: Elto, resp: lógico)**
- **Dequeue (C: cola, E: Elto, resp: lógico)**
- **Tamaño (C: cola, n: numérico)**

8.2. Implementación con listas enlazadas

COLA = LISTA;

Las operaciones *Crear_cola* (C, ok), *Borrar_cola* (C, ok), *Tamaño* (C, tam), *Llena?*(C, resp) y *Vacía?* (C, resp) serán iguales que en las listas.

Algoritmo *Queue* (C: Cola, E: Elemento, resp: lógico) es
Tmp: posición;
INICIO
Llena? (C, resp);
Si resp = cierto entonces
Error ("Cola llena");
resp := falso;
Sino {insertar tras el último, que puede ser el primero también}
Obtener (tmp);
Tmp→.info := E;
Tmp→.sgte := nil; {porque va a ser el último}
Vacía? (C, resp);
Si resp = cierto entonces
C.prim := tmp; {será el primero}
Sino
C.ult→.sgte := tmp; {irá tras el siguiente al último}
Fin si;
C.ult := tmp; {será el nuevo último elemento}
C.longitud := C.longitud + 1;
Fin si;
FIN

Algoritmo *Dequeue* (C: Cola, E: Elemento, Resp: lógico) es
Tmp: posición;
Tam: numérico;
INICIO
resp := cierto;
Vacía? (C, resp);
si resp = cierto entonces
Error ("Cola vacía");
resp := falso;
sino {al menos hay un elemento}
Tmp := C.prim;
E := tmp→.info;
Tamaño (C, tam);
si tam = 1 entonces
C.ult := nil;
fin si
C.prim := tmp→.sgte; {puede ser nil, si sólo había uno}
Liberar(tmp);
C.longitud := C.longitud – 1;
Fin si
FIN

8.3.Implementación mediante vectores (estructura circular)

```
/* Definición del tipo cola */
#define TAM 1000

typedef struct {
      int longitud, prim, ult;
      int DATOS[TAM];
} cola;
```

- int crear_cola (cola * C)
- int borrar_cola (cola * C)
- int vacía? (cola C)
- int llena? (cola C)
- int tamaño (cola C)
- void queue (cola * C, ELTO E)
- ELTO dequeue (cola * C)