PROGRAMACION II

MATERIAL para TEORIA

COLAS

Código Asignatura 6A4

Año 2022

TIPO de DATOS COLA

Una cola es una estructura de datos que consiste en una colección de elementos, todos del mismo tipo, en la cual los datos se incorporan por un extremo y se eliminan por otro. No se permite acceder a elementos intermedios ni recorrerla.

Un ejemplo muy común es la cola que se forma en la caja de un negocio o la ventanilla de un banco. Recibe también el nombre de FIFO (First In First Out), ya que el elemento que se almacenó primero es el primero en removerse, de este modo podemos decir que una cola *preserva* el orden de los elementos que en ella se almacenan. A pesar de su sencillez, es muy adecuada para la información que se comporta de este modo en el almacenamiento y recuperación. Es el caso, por ejemplo de la simulación de procesos de atención (bancos, supermercados, semáforos) y de impresión, entre otros.

TDA COLA

Para definir un Tipo de Dato Abstracto Cola, es necesario definir el tipo y un conjunto de operadores que permitan declarar y utilizar variables del tipo. Para desarrollar algoritmos que almacenen y recuperen elementos de una Cola, es necesario conocer las cabeceras de los operadores (interface), pero no los detalles de su implementación (caja negra)

OPERADORES del TDA COLA

Siendo C : cola de objetos de tipo "TElementoC" y x : objeto de tipo "TElementoC"

- INICIAC(C) → Devuelve en C una cola vacía
- VACIAC(C) → Devuelve Verdadero si la cola C está vacía, y Falso en caso contrario
- CONSULTAC(C) → Devuelve el valor del elemento que se encuentra en el primer lugar de la cola C
- SACAC(C , x) → Devuelve en x el elemento que se encuentra en el primer lugar de la cola removiéndolo de C
- PONEC(C , x) → Inserta el elemento x en el último lugar de la cola.

INTERFACE del TDA COLA

void **IniciaC** (TCola * C)

int **VaciaC** (TCola C)

TElementoC **consultaC** (TCola C)

void **sacaC** (TCola *C, TElementoC* x)

void **poneC** (TCola *C, TElementoC x)

UTILIZACION del TDA COLA

Problema:

Leer un conjunto de números (distintos de 0) y mostrar los que son impares y mayores que el promedio, en el orden en que ingresaron.

```
#include <stdio.h>
#include <colas.h>

int main()  {
    TCola Cnros;
    TElementoC num;
    int sum = 0, cuenta = 0;
    float prom;

    IniciaC (&Cnros);
    scanf("%d",&num);
    while (num) {
        sum += num;
        cuenta++;
        if (num % 2)
           poneC (&Cnros, num);
        scanf("%d",&num);
    }
   if (cuenta) {
       prom = sum/cuenta;
       printf("Los numeros impares mayores al promedio son ”);
       while (!VaciaC(Cnros)) {
           sacaC (&Cnros, &num);
           if (num > prom)
               printf("%d  ", num);
      }
   else
    printf("No se registraron numeros impares mayores al promedio”);
 }
 return 0;
}
```

IMPLEMENTACION del TDA COLA

⇒ IMPLEMENTACION ESTATICA

```
#define MAX 50
typedef int TElementoC;
typedef struct {
 TElementoC datos[MAX];
    int pri, ult; } TCola;
void IniciaC (TCola *C) {
 (*C).pri=-1;
 (*C).ult=-1;
}
int VaciaC(TCola C){
  return C.pri==-1;
}
void poneC (TCola *C, TElementoC X) {
   if ((*C).ult != MAX-1) {
     if ((*C).pri==-1)
       (*C).pri = 0;
    (*C).datos[++((*C).ult)]=X;
  }
}
void sacaC (TCola *C, TElementoC *X) {
  if ((*C).pri != -1) { // !vaciaC(*C)
     *X = (*C).datos[(*C).pri];
    if ((*C).pri == (*C).ult)
       iniciaC(C);
    else
       (*C).pri +=1;
  }
}
TElementoC consultaC (TCola C){
if (C.pri !=-1)
   return C.dato[C.pri];
}
```

⇒ IMPLEMENTACION DINAMICA

```
typedef int TElementoC;
typedef struct nodo {
  TElementoC dato;
  struct nodo * sig;} nodo;
typedef struct {
 nodo *pri, *ult;} TCola;
void IniciaC (TCola *C){
 (*C).pri=NULL;
 (*C).ult=NULL;
}
int VaciaC(TCola C){
  return C.pri==NULL;
}
void poneC (TCola *C, TElementoC X) {
   nodo * aux;
   aux = (nodo *) malloc (sizeof(nodo));
   aux->dato = X;
   aux->sig = NULL;
   if ((*C).pri==NULL)
      (*C).pri=aux;
   else
      (*C).ult->sig=aux;
  (*C).ult=aux;
}
void sacaC (TCola *C, TElementoC *X){
  nodo * aux;
  if ((*C).pri !=NULL) {
     aux = (*C).pri;
   *X = aux->dato;
   (*C).pri = (*C).pri->sig;
   if ((*C).pri == NULL)
      (*C).ult = NULL;
   free(aux);
   }
}
TElementoC consultaC (TCola C){
if (C.pri !=NULL)
   return C.pri-> dato;
}
```

⇒ IMPLEMENTACION ESTATICA CIRCULAR

```
void poneC (TCola *C, TElementoC X) {
   if  (!((*C).ult==MAX-1 && (*C).pri==0 || (*C).ult+1==(*C).pri)) {
     if ((*C).pri==-1){
       (*C).pri = 0;
       (*C).ult = 0;
    }
    else
      if (*C).ult == MAX-1)
          (*C).ult = 0;
      else
           (*C).ult += 1;
    (*C).datos[(*C).ult]=X;
  }
}
void sacaC (TCola *C, TElementoC *X) {
  if ((*C).pri != -1) {
     *X = (*C).datos[(*C).pri];
    if ((*C).pri == (*C).ult)
       iniciaC(C);
    else
      if (*C).pri == MAX-1)
         (*C).pri = 0;
      else
         (*C).pri += 1;
  }
}
```