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

struct Node{
 int num;
 struct Node *prox;
};
typedef struct Node node;

int tam;

void inicia(node *LISTA);
int menu(void);
void opcao(node *LISTA, int op);
node *criaNo();
void insereFim(node *LISTA);
void insereInicio(node *LISTA);
void exibe(node *LISTA);
void libera(node *LISTA);
void insere (node *LISTA);
node *retiraInicio(node *LISTA);
node *retiraFim(node *LISTA);
node *retira(node *LISTA);


int main(void)
{
 node *LISTA = (node *) malloc(sizeof(node));
 if(!LISTA){
  printf("Sem memoria disponivel!\n");
  exit(1);
 }else{
 inicia(LISTA);
 int opt;

 do{
  opt=menu();
  opcao(LISTA,opt);
 }while(opt);

 free(LISTA);
 return 0;
 }
}

void inicia(node *LISTA)
{
 LISTA->prox = NULL;
 tam=0;
}

int menu(void)
{
 int opt;

 printf("Escolha a opcao\n");
 printf("0. Sair\n");
```

```
 printf("1. Zerar lista\n");
 printf("2. Exibir lista\n");
 printf("3. Adicionar node no inicio\n");
 printf("4. Adicionar node no final\n");
 printf("5. Escolher onde inserir\n");
 printf("6. Retirar do inicio\n");
 printf("7. Retirar do fim\n");
 printf("8. Escolher de onde tirar\n");
 printf("Opcao: "); scanf("%d", &opt);

 return opt;
}

void opcao(node *LISTA, int op)
{
 node *tmp;
 switch(op){
  case 0:
   libera(LISTA);
   break;

  case 1:
   libera(LISTA);
   inicia(LISTA);
   break;

  case 2:
   exibe(LISTA);
   break;

  case 3:
   insereInicio(LISTA);
   break;

  case 4:
   insereFim(LISTA);
   break;

  case 5:
   insere(LISTA);
   break;

  case 6:
   tmp= retiraInicio(LISTA);
   printf("Retirado: %3d\n\n", tmp->num);
   break;

  case 7:
   tmp= retiraFim(LISTA);
   printf("Retirado: %3d\n\n", tmp->num);
   break;

  case 8:
   tmp= retira(LISTA);
   printf("Retirado: %3d\n\n", tmp->num);
   break;
```

```
  default:
   printf("Comando invalido\n\n");
 }
}

int vazia(node *LISTA)
{
 if(LISTA->prox == NULL)
  return 1;
 else
  return 0;
}

node *aloca()
{
 node *novo=(node *) malloc(sizeof(node));
 if(!novo){
  printf("Sem memoria disponivel!\n");
  exit(1);
 }else{
  printf("Novo elemento: "); scanf("%d", &novo->num);
  return novo;
 }
}


void insereFim(node *LISTA)
{
 node *novo=aloca();
 novo->prox = NULL;

 if(vazia(LISTA))
  LISTA->prox=novo;
 else{
  node *tmp = LISTA->prox;

  while(tmp->prox != NULL)
   tmp = tmp->prox;

  tmp->prox = novo;
 }
 tam++;
}

void insereInicio(node *LISTA)
{
 node *novo=aloca();
 node *oldHead = LISTA->prox;

 LISTA->prox = novo;
 novo->prox = oldHead;

 tam++;
}

void exibe(node *LISTA)
{
```

```
 system("clear");
 if(vazia(LISTA)){
  printf("Lista vazia!\n\n");
  return ;
 }

 node *tmp;
 tmp = LISTA->prox;
 printf("Lista:");
 while( tmp != NULL){
  printf("%5d", tmp->num);
  tmp = tmp->prox;
 }
 printf("\n        ");
 int count;
 for(count=0 ; count < tam ; count++)
  printf("  ^  ");
 printf("\nOrdem:");
 for(count=0 ; count < tam ; count++)
  printf("%5d", count+1);


 printf("\n\n");
}

void libera(node *LISTA)
{
 if(!vazia(LISTA)){
  node *proxNode,
     *atual;

  atual = LISTA->prox;
  while(atual != NULL){
   proxNode = atual->prox;
   free(atual);
   atual = proxNode;
  }
 }
}

void insere(node *LISTA)
{
 int pos,
  count;
 printf("Em que posicao, [de 1 ate %d] voce deseja inserir: ",
tam);
 scanf("%d", &pos);

 if(pos>0 && pos <= tam){
  if(pos==1)
   insereInicio(LISTA);
  else{
   node *atual = LISTA->prox,
     *anterior=LISTA;
   node *novo=aloca();

   for(count=1 ; count < pos ; count++){
```

```
      anterior=atual;
      atual=atual->prox;
    }
    anterior->prox=novo;
    novo->prox = atual;
    tam++;
   }

 }else
  printf("Elemento invalido\n\n");
}

node *retiraInicio(node *LISTA)
{
 if(LISTA->prox == NULL){
  printf("Lista ja esta vazia\n");
  return NULL;
 }else{
  node *tmp = LISTA->prox;
  LISTA->prox = tmp->prox;
  tam--;
  return tmp;
 }

}

node *retiraFim(node *LISTA)
{
 if(LISTA->prox == NULL){
  printf("Lista ja vazia\n\n");
  return NULL;
 }else{
  node *ultimo = LISTA->prox,
    *penultimo = LISTA;

  while(ultimo->prox != NULL){
   penultimo = ultimo;
   ultimo = ultimo->prox;
  }

  penultimo->prox = NULL;
  tam--;
  return ultimo;
 }
}

node *retira(node *LISTA)
{
 int opt,
  count;
 printf("Que posicao, [de 1 ate %d] voce deseja retirar: ",
tam);
 scanf("%d", &opt);

 if(opt>0 && opt <= tam){
  if(opt==1)
   return retiraInicio(LISTA);
```

```
  else{
   node *atual = LISTA->prox,
     *anterior=LISTA;

   for(count=1 ; count < opt ; count++){
    anterior=atual;
    atual=atual->prox;
   }

  anterior->prox=atual->prox;
  tam--;
  return atual;
  }

 }else{
  printf("Elemento invalido\n\n");
  return NULL;
 }
}
```