23-11-2020

Vallisha.M

Program to perform addition, deletion at beginning or end or at any position of a linked list.

```
typedef struct node{
    int value;
    struct node * node; next;
} node;
node* head = NULL;

// To add at beginning
void putBeg (int value){
    node* pter = (node*) malloc (sizeof(node));
    pter -> value = value;
    if (head == NULL) {
        head = pter;
        head -> next = NULL;
        return;
    }
    ptr -> next = head;
    head = pter;
}
void put (int pos, int value) // add at any pos
{
    node* pter = (node*) malloc ((sizeof(node)));
    pter -> value = value;
    if (head == NULL && pos == 0)
```

```
{ head = ptr;
  head -> next = NULL;
  return;
}
if(pos == 0)
{
  free(ptr);
  putBeg(value);
  return;
}
int i = 0;
node* tmp = head;
while(i != pos-1 && tmp != NULL)
{
  i++;
  tmp = tmp -> next;
}
if(i != pos-1)
{
  printf("\n ERROR \n Enter Correct Index \n\n");
  return;
}
```

```
ptr -> next = temp -> next;
temp -> next = ptr;
}
```

```
// adding at end
void putEnd(int value){
    node * ptr = (node *) malloc((sizeof(node)));
    ptr -> value = value;
    ptr -> next = NULL
    if( head == NULL){
        head = ptr;
        return;
    }
    node*  tmp = head;
    for (; tmp -> next != NULL; tmp = tmp->next);
     tmp -> next = ptr;
}
void display()
{
    if(head == NULL) {
      printf("Linked list is empty ");
      return;
   }
   printf("\n\n Linked List contains : ");
   for(node * tmp = head; tmp != NULL;
      tmp  = tmp -> next)
          printf("%d  ", tmp -> value);
    printf("\n\n");
}
```

```
void delFirst(){ // delete first element

    if(head == NULL){
        printf("\n\n Linked List is empty\n\n");
        return;
    }
    if(head->next == NULL)
    {
    node * tmp = head -> next;
    free(head);
    head = tmp;
}
// delete last element
void delLast(){
    if(head == NULL){
        printf("\n\n LinkedList is empty\n\n");
        return;
    }
    if (head -> next == NULL)
    {
        free(head);
        head = NULL;
        return;
    }
    node* tmp = head;
    for(; (tmp->next)->next != NULL;
      tmp = tmp->next);
    node* tmp1 = tmp -> next;
     tmp -> next = NULL;
```

```
    free(tmpD);
}
// delete at any index
Void delPos(int pos){
    if(head == NULL)
    {
        printf("\n\nLinked List is Empty\n\n");
        return;
    }
if(pos == 0)
{
    delFirst();
    return;
}
int i = 0;
node * tmp = head;
while(i != pos-1 && tmp != NULL)
{
    i++;
    tmp = tmp -> next;
}
if(i != pos-1)
{
    printf("\n\nERROR\nEnter correct
    index\n\n");
    return;
}
```

```
node* tmp1 = tmp->next;
tmp->next = (tmp->next)->next;
  free(tmp1);
}
```