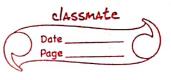


```
LINKED LIST IMPLEMENTATION
  struct Node 5
     int data;
    struct Node *next;
   struct Node * head-ref;
 void push (int new-data)
   Struct Node *new_node = (struct Node *) malloc (sizeof (struct Node));
   new-node → data = new-data;
   new-node -> next = NULL;
     if (nead-ref == NULL)
        head-ref = new-node;
      new-node -> next = head-ref;
     head-ref = new-node:
void append (int dat new-data)
 struct Node *new-node= (struct Node +) malloc (size of (struct Node));
  new_node → data = new-data;
  new-node - next = NULL;
  struct Node *last = head-ref;
  if (head-ref == NULL)
   head-ref = new-node;
  else f
    while (12st -> next != NULL)
      last = (ast-> next;
   12st → next = new-node; }
```

```
classmate
 Void insert pos (int new data, int pos) (
   Stuck Node + new-node = (struct Hode ) malloc (struct Mode);
   Struct Node *ptr = head-ref;
  new-node -> data= new-data;
  if (pos == 1) }
   new-node → next = ptr;
   head-ref = new-node;
   leturn',
   for (int i=1; 1 < pos; i++) [
   ptr = ptr -7 next;
  if (ptr == NULL)
   printf ("In Invalid position.");
     mode new-node -> next = ptr -> next;
     ptr → next = new_node;
Void pop ()
 struct Node *ptr = head-ref;
 if (head-ref == NULL)
  printf (" Empty List");
   head-ref = ptr → nout;
   ptr - next = NULL;
  free (ptr);
```



```
void end-delete ()
 Struct Node *ptr = head - ref , * ptr1,
  printf (" Empty List ");
 else if ( nead-Te) - next == NULL)
   free (head-ref):
  while (ptr -> next != NULL)
     Ptr1= ptr;
   free (ptr);
void du-any (int pos) [
 struct Node * ptr = nead-ref, * ptr1;
 for (int i= 0; i < pos; i+)
    ptr = ptr -> next;
 if (ptr == NULL)
 printf (" Invalid pas ");
   ptr1 -> next = ptr -> next;
 free (ptr);
```