LINKED LIST IMPLEMENTATION

```
struct Node {
    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 (head-ref == NULL) {
          head-ref = new-node;
    }
    else {
       new-node → next = head-ref;
       head-ref = new-node;
    }
}

void append (int ~~dat~~ new-data) {
  struct Node *new-node = (struct Node*) malloc( sizeof (struct Node));
  new-node → data = new-data;
  new-node → next = NULL;
  struct Node *last = head-ref;
   if (head-ref == NULL)
       ~~new~~ head-ref = new-node;
  else {
     while (last→next != NULL)
          last = last→next;
     last→next = new-node;  }
}
```

```
void insert_pos( int new_data, int pos) {
  struct Node* new_node = (struct Node*) malloc (sizeof(struct Node));
  struct Node *ptr = head_ref;
  new_node → data = new_data;
  if (pos == 1) {
     new_node → next = ptr;
     head_ref = new_node;
     return;
  }
  for (int i = 1; i < pos; i++) {
     ptr = ptr → next;
  }
  if (ptr == NULL)
     printf("\n Invalid position. ");
  else {
     ~~node~~ new_node → next = ptr → next;
          ptr → next = new_node;
  }
}

void pop( ) {
  struct Node *ptr = head_ref;
  if (head_ref == NULL)
     printf(" Empty List");
  else {
     head_ref = ptr → next;
     ptr → next = NULL;
     free(ptr);
  }
}
```

```
void end_delete () {
  struct Node *ptr = head_ref , *ptr1,
   if (head_ref == NULL)
       printf(" Empty List ");
   else if ( head_ref → next == NULL) {
       head_ref =• NULL;
       free (head_ref);
   }
   else {
      while (ptr → next != NULL) {
          ptr1 = ptr;
          ptr = ptr → next;
      }
      ptr1 → next = NULL;
      free (ptr);
   }
}
void del_any ( int pos) {
   struct Node *ptr = head_ref, *ptr1;
   for (int i = 0 ; i < pos ; i++) {
       ptr1 = ptr;
       ptr = ptr → next;
   }
   if (ptr == NULL)
      printf(" Invalid pos ");
   else {
      ptr1 → next = ptr → next;
      free (ptr);
   }
}
```