Week-6 [Stack Implementation

Executed stack implementing SLL ✓

25/1/24

```
#
#

struct node {
        int data;
        struct data node* next;
};

void append(struct node ** head, int new_data
{
    struct node *new_node = (struct node *)ma
    (sizeof(struct node));

    new_node -> data = new_data;
    new_node -> next = NULL;
    struct node* last = *head;

    if ( *head == NULL )
        *head = new_node;
    else {
        while (last -> != NULL)
            last = last -> next;

        last -> next = new_mode;
    }
}
```

```
void display(struct node *head){
    if (head == NULL) {
        printf("Linked list empty");
        return;
    }
    while (head != NULL){
        printf("%d ", head->data);
        head = head->next;
    }
    printf("\n");
}
void del_end (struct node *head) {
    if (head == NULL){
        printf ("List empty");
        return;
    }
    struct node * last = head;

    struct node *prev;
    while (last->next != NULL) {
        prev = last;
        last = last->next;
    }
    free (last);
    prev->next = NULL;
}
```

```
int main(){
    struct node *head = NULL;
    int choice, a;
    while (choice < 4){
        printf("1. Push \n 2. Pop \n 3. Display \n
        Choice");
    scanf("%d", &choice);
    switch (choice);
    {
      case 1:
          printf("Enter value");
          scanf("%d", &a);
          append (&head, a);
          display (head);
          break;
      case 2:
          del-end (head);
          display (head);
          break;
      case 3:
          display (head);
      default;
          break;
        }
    }
    return 0;
}
```

Queue Implementation

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

struct node {
    int data;
    struct node *next;
};

void append (struct node **head, int 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;

    if (*head == NULL)
        *head = new_node;
    else {
        while (last -> next != NULL)
            last = last -> next;

        last -> next = new_node;
    }
}
```

```
void display (struct node *head) {
    if (head != NULL) {
        printf ("%d", head->data
    if (head == NULL) {
        printf ("Empty");
    }
    while (head != NULL) {
        printf ("%d", head->data);
        head = head->next;
    }
}
void del_head (struct node **head) {
    if (*head == NULL) {
        printf ("List is empty");
    }
    struct node *temp = (*head)->next;
    free (*head);
    *head = temp;
}
int main () {
    struct node *head = NULL;
    int choice, a;

    while (choice < 4) {
        printf ("1. Push 2. Pop 3. Display \n Choice:");
        scanf ("%d", &choice);
        switch (choice)
        {
```

```
case 1:
    printf("Enter value");
    scanf("%d", &a);
    append(&head, a);
    display(head);
    break;
case 2:
    del_head(&head);
    display(head);
    break;
case 3:
    display(head);
default:
    break;
}
return 0;
}
```

Output:

1. Push
2. Pop
3- Display
Choice: 1
Queue : 1
Choice
1. Push
2. Pop
3. Display

Choice: 1
Value: 2
Queue : 12
1. Push
2. Pop
3. Display
Choice 2
Queue : 1