PAGE NO :

DATE : 19/11/2024

Lab 6

Stack and Queue Implementation Using Linked Lists

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

struct Node{
        int data;
        struct Node *link;
};

typedef struct Node node;

//Stack
node *top = NULL;
void push();
void pop ();
void displayStack;

void push(){
        node * new1 = (node *)malloc (sizeof (node));
        if (new1 == NULL){
                printf("\n Stack Overflow.\n");
                return;
        }
        printf("\n Enter Value to Push : ");
        scanf("%d", &new1 -> data);
        new1 -> link = top;
        top = new1;
}
```

```
void pop(){
    if (top == NULL){
        printf("\n Stack Underflow.\n");
        return;
    }
    node *temp = top;
    printf("\n Popped Element : %d \n", temp->data);
    top = top -> link;
    free(temp);
}

void displayStack(){
    if (top == NULL){
        printf("\n The Stack is Empty.\n");
        return;
    }
    printf("\n Elements in the Stack: ");
    node * temp = top;
    while (temp != NULL){
        printf("%d", temp -> data);
        temp = temp -> link;
    }
    printf("\n");
}


// Queue
node * front = NULL, * rear = NULL;
void insert();
void del();
void displayQueue();
void insert(){
    node *new1 = (node *)malloc(sizeof(node));
    if (new1 == NULL){
        printf("\n Queue Full. \n");
        return;
    }
    printf("\n Enter Value to Insert : ");
    scanf("%d", &new1->data);
    new1->link = NULL;
    if(rear==NULL){
        front= new1;
        rear =new1;
        return;
    }
    rear->link=new1;
    rear= new1;
}

void del(){
    if (front == NULL){
        printf("\n Queue Empty. \n");
        return;
    }
    node *temp = front;
    printf("\n Deleted Element : %d \n", temp->data);
    front = front->link;
    if(front==NULL){
        rear=NULL;
    }
    free(temp);
}
```

```
void displayQueue(){
    if(front==NULL){
        printf("\n The Queue is Empty.\n");
        return;
    }
    printf("\n Elements in the Queue: ");
    node *temp=front;
    while(temp!=NULL){
        printf("%d", temp->data);
        temp=temp->link;
    }
    printf("\n");
}

//Main
void main(){
    int ch;
    while(1){
        printf("\n 1.Push (Stack) \n2. Pop (Stack) \n3. Display (Stack) ");
        printf("\n 4. Insert (Queue) \n5. Delete (Queue) \n6. Display
         (Queue) \n7.Exit ");
        printf("\n Enter Your Choice: ");
        scanf("%d", &ch);
        switch(ch){
            case 1: push();
                    break;
            case 2: pop();
                    break;
            case 3: displayStack();
                    break;
            case 4: insert();
                    break;
            case 5:  del();
                        break;
            case 6:  display Queue;
                        break;
            case 7:  exit(0);
            default:
                   printf("Enter Your Choice: \n");
      }
   }
}
```

Output

```
 1.  Push ( Stack)
 2.  Pop ( Stack)
 3.  Display ( Stack)
 4.  Insert ( Queue)
 5.  Delete ( Queue)
 6.  Display ( Queue)
→  Enter Your Choice: 1
   Enter Value to Push : 10

→  Enter Your Choice: 1
   Enter Value to Push: 20

→  Enter Your Choice : 1
   Enter Value to Push: 30

→  Enter Your Choice: 1
   Enter Value to Push: 40

→  Enter Your Choice: 3
   Elements in the Stack: 40  30  20  10
```

```
→ Enter Your Choice : 2
Popped Element : 40

→ Enter Your Choice: 2
Popped Element: 30

→ Enter Your Choice: 3
Elements in the Stack: 20   10

→ Enter Your Choice: 4
Enter Value to Insert : 10

→ Enter Your Choice: 4
Enter Value to Insert : 20

→ Enter Your Choice: 4
Enter Value to Insert: 30

→ Enter Your Choice: 4
Enter Value to Insert: 40

→ Enter Your Choice: 6
Elements in the Queue : 10   20   30   40.

→ Enter Your Choice: 5
Deleted Element: 10

→ Enter Your Choice: ~~6~~ 5
Deleted Element : 20

→ Enter Your Choice: 6
Elements in the Queue: 30   40
```

3/12/2024