15 - LL stack

Aim

Create stack using Linked List

Algorithm

I struct stack

1. START
2. int data
3. struct stack *next
4. struct stack *prev
5. STOP

II Create *temp, *head and *tail using struct stack

III void ~~insert~~ display()

1. START
2. temp = head
3. Check if stack is empty
4. Else do print("%d", temp → data)
   while (temp != NULL)
5. STOP

IV void insert()

1. START
2. Initialise val of int
3. Initialise and allocate struct stack *p
4. Prompt and Input from user p→data
5. if (head == NULL) then
   1. head = p
   2. head → next = NULL
   3. head → prev = NULL
   4. tail = head,
6. Else
   1. tail → next = p
   2. tail → next → prev = tail
   3. tail = p
   4. tail → next = NULL
7. End if
8. STOP

V void pop()

1 START
2. check if stack is empty then print empty
3. Else
    1. If (head == tail)
        1 head = NULL
        2 tail = NULL
    2 Else
        1 temp = tail
        2. tail = tail → prev
        3. tail → next = NULL
        4 free (temp)
    3 End if
4. End if
5. STOP

VI ~~void out main()~~

~~1. START~~

Output

Obtained

16 - LL queue

Aim

Create Queue data structure using Linked List

Algorithm

I. struct que

1. START
2. int data
3. struct que *next
4. struct que *prev
5. STOP

II create struct que variables like *temp, *head, *tail

III void display()

1. START
2. temp = head
3. check if list is empty then print empty
4. Else
    1. iterate through each element and

print its data

5. End If
6. STOP

IV void append()

1. START
2. int val
3. Create and allocate struct que *p
4. Prompt and Input from user p->data
5. If head is NULL Then put p to head and tail
6. Else append &p to the end of tail and set it as tail
7. STOP

V void popleft()

1. START
2. check if que is empty then print empty
3. Else
    1. print (head->data)
    2. set ~~remove~~ head ~~->data~~ = head->next
    3. remove head -> ~~data~~ prev

4. End If
5. STOP

VI int main()

1. START
2. Initialise the variables
3. display the menu
4. int q = 1
5. while (q != 0) // infinite loop
    1. Input the choice
    2 case 1 : display
    3 Cas 2 : append
    4 : case 3: pop left.
    5. case 4 : set q = 0
6 End While
7 STOP

Output

Output obtained.

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

int sz=0;

struct stack{
    int data;
    struct stack *next;
    struct stack *prev;
} *temp ,*head=NULL ,*tail=NULL;

void display(){
    temp=head;
    if(sz==0)//(temp==NULL)
        printf("Stack Empty");
    else{
        printf("%d ",temp->data);
        temp=temp->next;
        while(temp!=NULL){
            printf("->%d ",temp->data);
            temp=temp->next;
        }
    }
    printf("\n");
}

void insert(){
    int val;
    struct stack *p;
    p=(struct stack*)malloc(sizeof(struct stack));

    printf("Enter the val: ");
    scanf("%d",&(p->data));
    if(head==NULL){
        head=p;
        head->next=NULL;
        head->prev=NULL;
        tail=head;
    }
    else{
        tail->next=p;
        tail->next->prev=tail;//p.prev
        tail=p;
        tail->next=NULL;//p.next
    }
    sz++;
}
void pop(){
    if(sz==0)//(temp==NULL)
        printf("Stack Empty\n");
```

```
    else{
        printf("%d is removed\n",tail->data);
        if(head==tail){
            head=NULL;
            tail=NULL;
        }
        else{
            temp=tail;
            tail=tail->prev;
            tail->next=NULL;
            free(temp);
        }
        sz--;
    }
}

int main(){
    int choice;
    int pos;
    printf("1...display\n");
    printf("2...insert\n");
    printf("3...pop\n");
    printf("4...quit\n");

    int quit=1;
    while(quit!=0){
        printf("\nOption : ");
        scanf("%d",&choice);
        switch(choice){
                case 1: display();
                        break;
                case 2: insert();
                        break;
                case 3: pop();
                        break;
                case 4: quit=0;
                break;
            default:
                printf("\n1...display\n");
                printf("2...insert\n");
                printf("3...pop\n");
                printf("4...quit\n");
        }
    }

    return 0;
}
```

```
1...display
2...insert
3...pop
4...quit

Option : 2
Enter the val: 1

Option : 2
Enter the val: 2

Option : 2
Enter the val: 3

Option : 1
1 ->2 ->3

Option : 3
3 is removed

Option : 3
2 is removed

Option : 3
1 is removed

Option : 3
Stack Empty

Option : 1
Stack Empty

Option : 4
>
```

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

int sz=0;

struct que{
    int data;
    struct que *next;
    struct que *prev;
} *temp ,*head=NULL ,*tail=NULL;

void display(){
    temp=head;
    if(sz==0)//(temp==NULL)
        printf("Queue Empty");
    else{
        printf("%d ",temp->data);
        temp=temp->next;
        while(temp!=NULL){
            printf("->%d ",temp->data);
            temp=temp->next;
        }
    }
    printf("\n");
}
void revdisp(){
    temp=tail;
    if(sz==0)//(temp==NULL)
        printf("Queue Empty");
    else{
        printf("%d",temp->data);
        temp=temp->prev;
        while(temp!=NULL){
            printf("<- %d",temp->data);
            temp=temp->prev;
        }
    }
    printf("\n");
}

void append(){
    int val;
    struct que *p;
    p=(struct que*)malloc(sizeof(struct que));

    printf("Enter the val: ");
    scanf("%d",&(p->data));
    if(head==NULL){
        head=p;
        head->next=NULL;
```

```
        head->prev=NULL;
        tail=head;
    }
    else{
        tail->next=p;
        tail->next->prev=tail;//p.prev
        tail=p;
        tail->next=NULL;//p.next
    }
    sz++;
}

void popleft(){
    if(sz==0)//(temp==NULL)
        printf("Queue Empty\n");
    else{
        printf("%d is removed\n",head->data);
        if(head==tail){
            head=NULL;
            tail=NULL;
        }
        else{
            temp=head;
            head=head->next;
            head->prev=NULL;
            free(temp);
        }
        sz--;
    }
}

int main(){
    int choice;
    int pos;
    printf("1...display\n");
    printf("2...append\n");
    printf("3...popleft\n");
    printf("4...quit\n");

    int quit=1;
    while(quit!=0){
        printf("\nOption : ");
        scanf("%d",&choice);
        switch(choice){
                case 1: display();
                        break;
                case 2: append();
                        break;
                case 3: popleft();
                        break;
```

```
                case 4: quit=0;
                    printf("Exiting from program");
                    break;
                default:
                printf("\n1...display\n");
                printf("2...append\n");
                printf("3...popleft\n");
                printf("4...quit\n");
        }
    }

    return 0;
}
```

```
1...display
2...append
3...popleft
4...quit

Option : 2
Enter the val: 1

Option : 2
Enter the val: 2

Option : 2
Enter the val: 3

Option : 1
1 ->2 ->3

Option : 3
1 is removed

Option : 3
2 is removed

Option : 3
3 is removed

Option : 3
Queue Empty

Option : 1
Queue Empty

Option : 4
Exiting from program
```