Experiment 14
Date 01-12-2022

QUEUE USING LINKED LIST

Aim

Write a menu driven C Program to implement a queue using linked list with the following operations:
a.Insert elements to the queue.
b.Delete elements from the queue.
c.Display the queue after each operation

1 Queue using Linked List

1.1 Algorithm

```
Step 1: Start
Step 2: Create structure having members item and a self referential link
Step 3: Declare structure variable queue_head and dynamically allocate memory to the node
Step 4: Set default values of item and link of queue_head as NULL
Step 5: Create front and rear pointers for the queue, initially pointing to the queue_head
Step 6: Display menu items(1.Insert, 2.Delete, 3.Exit)
Step 7: Input option, op
Step 8: If op=1, then enqueue()
Step 9: If op=2, then dequeue()
Step 10: If op=3, then goto STEP 14
Step 11: Else print "Invalid input", goto STEP 6
Step 12: display()
Step 13: Goto STEP 6
Step 14: Stop
```

Function enqueue() inserts element to end of the queue

```
Step 15: Start of function enqueue()
Step 16: Allocate memory for node, new using get_node()
Step 17: If new=NULL, return
Step 18: Else set ptr←header, let ptr←(ptr→link) until (ptr→link)=NULL
Step 19: Let (ptr→link)←new, input new→item, let (new→link)←NULL
Step 20: Set rear pointer to new
Step 21: return
```

Function dequeue() deletes the element at beginning of the queue

```
Step 22: Start of function dequeue()
Step 23: If front=rear=header or (header→link)=NULL, return
Step 24: Else let (header→link)←(ptr→link)
Step 25: Return the deleted node to memory pool using return_node(ptr)
Step 26: Set front pointer to ptr→link
Step 27: return
```

Function get_node() returns a node of required size from memory pool

```
Step 28: Start of function get_node()
Step 29: Return a dynamically allocated node of the size of structure
```

Function return_node() returns the deleted node to memory pool

```
Step 30: Start of function return_node(pr)
Step 31: Set the value of data and link of ptr to NULL
Step 32: return
```

Function display() displays elements of the queue

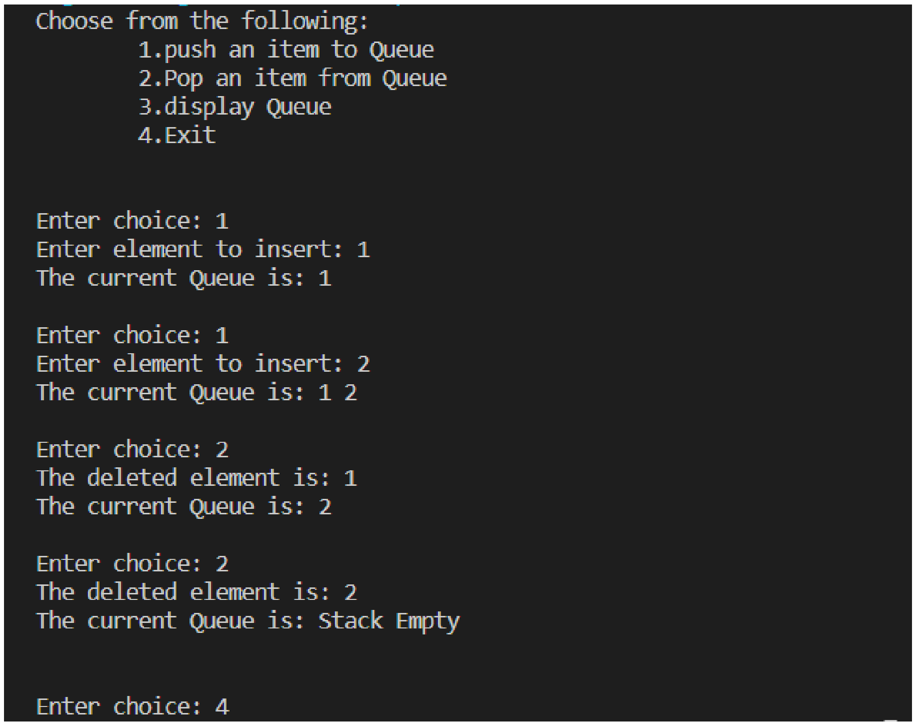
```
Step 33: Start of function display()
Step 34: Print ptr→item from ptr←(header→link) until ptr=NULL with ptr←
         (ptr→link) after each iteration
Step 35: return
```

1.2 Program

```
#include <stdio.h>
#include <stdlib.h>
struct node{
    int data;
    struct node *next;
};
struct node *head, *top;
struct node* createNode(){
return (struct node*)malloc(sizeof(struct node));
}
void popLL()
{
struct node *ptr;
if(head->next==NULL){
printf("Stack Empty\n");
return;
}
printf("The deleted element is: %d\n",top->data);
ptr=head;
while(ptr->next!=top){
ptr=ptr->next;
}
ptr->next=NULL;
free(top);
top=ptr;
}
void pushLL(int item)
{
struct node *new;
new=createNode();
new->data=item;
new->next=NULL;
```

```
if(head->next==NULL){
top=head;
}
top->next=new;
top=new;
}
void displayLL()
{
struct node *ptr;
printf("The current stack is: ");
if(head->next==NULL){
printf("Stack Empty\n");
return;
}
ptr=head->next;
while(ptr!=NULL){
printf("%d ",ptr->data);
ptr=ptr->next;
}
}
void main()
{
int choice,num;
head= createNode();
head->next=NULL;
printf("Choose from the following:\n\t1.push an item to stack\n");
printf("\t2.Pop an item from stack\n\t3.display stack\n\t4.Exit\n");
while(1)
{
printf("\n\nEnter choice: ");
scanf("%d",&choice);
switch(choice)
{
case 1: printf("Enter Item to push: ");
scanf("%d",&num);
pushLL(num);
displayLL();
break;
case 2: popLL();
displayLL();
break;
case 3: displayLL();
break;
case 4: break;
default: printf("Invalid Input\n");
}
if(choice == 4)
break;
}
}
```

1.3 Sample Output



```
Choose from the following:
        1.push an item to Queue
        2.Pop an item from Queue
        3.display Queue
        4.Exit


Enter choice: 1
Enter element to insert: 1
The current Queue is: 1

Enter choice: 1
Enter element to insert: 2
The current Queue is: 1 2

Enter choice: 2
The deleted element is: 1
The current Queue is: 2

Enter choice: 2
The deleted element is: 2
The current Queue is: Stack Empty


Enter choice: 4
```

Figure 1: Input and Output

1.4 Result

A menu driven C program was made with the operations: Insert, Delete and Display. Insert inserts an element onto a queue in the form of a linked list, Delete deletes and item from the front of the queue and Display prints the entire contents of the stack. Stack implemented using linked list.