**Institute of Engineering & Management**

**Department of Computer Science & Engineering**

**Data Structure Laboratory for 2nd year 3rd semester 2017**

**Code: CS 392**

**Date:** 16/8/17

**ASSIGNMENT-3**

**Problem-1**

**Problem Statement:** Implement simple queue data structure using array

**Algorithm:** Step-1: START  
Step-2: Declare global variables front=rear=-1 as integer and a integer array queue[100]  
Step-3: Inside main(), declare flag=1, in as integers.  
Step-4: Repeat  
 Print the commands for user  
 Scan for in.  
 Switch for values of i between  
 case 1: call insert()  
 case 2: call del()  
 case 3: call display()  
 default: print “wrong input”.  
 Ask user whether to continue or exit  
 scan for flag  
 while flag is equal to 1  
Step-5: inside insert(), declare variables I, n=0, flag=0, flag1=0, len & character array buffer[100]  
Step-6: if rear >= 99  
 print “Queue full” & return  
Step-7: print “enter the data separated by spaces”  
Step-8: fflush(stdin) & gets(buffer)  
Step-9:len=strlen(buffer)  
Step-10: if len=0  
 print “no input” & return  
 else if front = -1  
 front = front +1  
Step-11: for i=0 to i=len repeat  
 if buffer[i]= ‘-’  
 flag = flag+1 & continue  
 if buffer[i]= ‘ ‘ and buffer[i] = ‘\0’  
 n=(n\*10) + (buffer[i]-‘0’)  
 continue  
 if flag is not equal to 0  
 rear = rear+1  
 queue[rear]=-n  
 else rear = rear+1  
 queue[rear]=n;  
 assigne n=0 & flag=0  
 if rear >= 99  
 flag1 = flag1+1 & break  
Step-12: if flag1 = 1  
 print “queue full”  
 return  
Step-13: inside del(), if rear=-1  
 print “Queue empty” & return;  
Step-14: front=front+1 & print “deleted”  
Step-15: if rear < front  
 rear=-1 & front=-1  
Step-16: inside display(), if rear = -1  
 print “Queue empty” & return  
Step-17: for i=front to i=rear repeat  
 print “queue[i]”  
Step-18: END

**Source code:** #include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
  
int queue[100], front=-1, rear=-1;  
  
void delete();  
void insert();  
void display();  
  
void main()  
{  
 int in,flag=0;  
 do  
 {  
 printf("Enter the command\n '1' to insert\n '2' to delete\n '3' to display\n");  
 scanf("%d",&in);  
 switch(in)  
 {  
 case 1: insert(); break;  
 case 2: delete(); break;  
 case 3: display(); break;  
 default: printf("wrong input\n");  
 }  
 printf("enter 1 to continue\n");  
 scanf("%d",&flag);  
 } while(flag==1);  
}  
  
void insert()  
{  
 int i, n=0, flag=0, flag1=0,len;  
 char buffer[200];  
 if(rear>=99)  
 {  
 printf("queue full\n"); return;  
 }  
 printf("enter the data separated by spaces\n");  
 fflush(stdin); gets(buffer);  
 len=strlen(buffer);  
 if(len<1)  
 {  
 printf("no input\n");  
 return;  
 } else if(front==-1) front++;  
 for(i=0;i<=len;i++)  
 {  
 if(buffer[i]=='-')  
 {  
 flag++; continue;  
 }  
 if(buffer[i]!=' ' && buffer[i]!='\0')  
 {  
 n=(n\*10)+(buffer[i]-'0');  
 continue;  
 }  
 if(flag!=0)  
 queue[++rear]=-n;  
 else queue[++rear]=n;  
 n=0; flag=0;  
 if(rear>=99)  
 { flag1++; break;}  
 }  
 if(flag1==1)  
 {  
 printf("queue full\n"); return;  
 }  
}  
  
void delete()  
{  
 if(rear==-1)  
 {  
 printf("queue empty\n"); return;  
 }  
 ++front; printf("deleted\n");  
 if(rear<front)  
 {  
 rear=-1; front=-1;  
 }  
}  
  
void display()  
{  
 int i;  
 if(rear==-1)  
 {  
 printf("queue empty\n");  
 return;  
 }  
 printf("The elements in the queue are\n");  
 for(i=front;i<=rear;i++)  
 printf("%d, ", queue[i]);  
}

**Input/Output:** Enter the command  
 '1' to insert  
 '2' to delete  
 '3' to display  
1  
enter the data separated by spaces  
45 65 76 78 89 53  
enter 1 to continue  
1  
Enter the command  
 '1' to insert  
 '2' to delete  
 '3' to display  
3  
The elements in the queue are  
45, 65, 76, 78, 89, 53, enter 1 to continue  
1  
Enter the command  
 '1' to insert  
 '2' to delete  
 '3' to display  
2  
Deleted  
enter 1 to continue  
1  
Enter the command  
 '1' to insert  
 '2' to delete  
 '3' to display  
3  
The elements in the queue are  
65, 76, 78, 89, 53, enter 1 to continue  
0

**Problem-2**

**Problem Statement:** Implement circular queue using array

**Algorithm:** Step-1: START  
Step-2: define MAX as 100  
Step-3: declare golobal variables front=rear=-1 & an array queue[MAX]  
Step-4: inside main(), declare variables in & flag=0 as integers  
Step-5: do (repeat)  
 print the user commands  
 scan for in  
 switch for value of ‘in’ in between  
 case 1: call insert() & break  
 case 2: call del() & break  
 case 3: call display() & break  
 default: print “wrong input”  
 print “enter 1 to continue”  
 scan for flag  
 while flag=1  
Step-6: inside insert(), if (rear+1)%MAX=front  
 print “queue full” & return  
Step-7: print “enter the data”  
Step-8: scan for queue[ (rear+1)%MAX ]  
Step-9: if front=-1  
 front = front+1  
Step-10: rear = (rear+1)%MAX  
Step-11: inside del(), if rear=1 and front=-1  
 print “queue empty” & return  
Step-12: print “deleted”  
Step-13: if rear%MAX=front%MAX  
 rear=front=-1  
 else front = (front +1)%MAX  
Step-14: inside display(), declare integer variable i  
Step-15: if rear=-1  
 print “queue empty” & return  
Step-16: if rear < front  
 from i = 0 to i = ( MAX - rear + front )  
 print queue[ (front + 1)%MAX ]  
 else from i = front to i = rear  
 print queue[ i ]  
Step-17: END

**Source code:** #include <stdio.h>  
#include <stdlib.h>  
  
#define MAX 100  
  
int queue[MAX], front=-1, rear=-1;  
  
void del();  
void insert();  
void display();  
  
void main()  
{  
 int in,flag=0;  
 do  
 {  
 printf("Enter the command\n '1' to insert\n '2' to delete\n '3' to display\n");  
 scanf("%d",&in);  
 switch(in)  
 {  
 case 1: insert(); break;  
 case 2: del(); break;  
 case 3: display(); break;  
 default: printf("wrong input\n");  
 }  
 printf("enter 1 to continue\n");  
 scanf("%d",&flag);  
 } while(flag==1);  
}  
  
void insert()  
{  
 if((rear+1)%MAX==front)  
 {  
 printf("queue full\n"); return;  
 }  
 printf("enter the data\n");  
 scanf("%d", &queue[(rear+1)%MAX]);  
 if(front==-1)  
 front=0;  
 rear++;  
}  
  
void del()  
{  
 if(rear==-1 && front==-1)  
 {  
 printf("queue empty\n"); return;  
 }  
 printf("deleted\n");  
 if(rear%MAX==front%MAX)  
 {  
 rear=-1; front=-1;  
 } else front=(front+1)%MAX;  
}  
  
void display()  
{  
 int i;  
 if(rear==-1)  
 {  
 printf("queue empty\n");  
 return;  
 }  
 printf("The elements in the queue are\n");  
 if(rear<front)  
 for(i=0;i<=(front+MAX-rear);i++)  
 printf("%d, ", queue[(front+i)%MAX]);  
 else  
 for(i=front;i<=rear;i++)  
 printf("%d, ", queue[i]);  
}

**Input/Output:** Enter the command  
 '1' to insert  
 '2' to delete  
 '3' to display  
1  
enter the data  
34  
enter 1 to continue  
1  
Enter the command  
 '1' to insert  
 '2' to delete  
 '3' to display  
1  
enter the data  
45  
enter 1 to continue  
1  
Enter the command  
 '1' to insert  
 '2' to delete  
 '3' to display  
3  
The elements in the queue are  
34, 45, enter 1 to continue  
1  
Enter the command  
 '1' to insert  
 '2' to delete  
 '3' to display  
2  
deleted  
enter 1 to continue  
1  
Enter the command  
 '1' to insert  
 '2' to delete  
 '3' to display  
3  
The elements in the queue are  
45, enter 1 to continue  
0

**Problem-3**

**Problem Statement:** Implement DE-queue using array

**Algorithm:** Step-1: START  
Step-2: declare global variables front = rear =-1 as integer & queue[ 100 ]  
Step-3: inside main(), declare variables in & flag = 0  
Step-4: do (repeat)  
 print the user commands  
 scan for in  
 switch for the values of ‘in’ in between  
 case 1: call insertf() & break  
 case 2: call deletef() & break  
 case 3: call insertr() & break  
 case 4: call deleter() & break  
 default: print “wrong input”  
 print “enter 1 to continue”  
 scan for flag  
 while flag = 1  
Step-5: inside insertr(), if rear >= 99  
 print “queue full” & return  
Step-6: print “enter the data”  
Step-7: scan for queue[ rear+1 ]  
Step-8: if front=-1  
 front = 0  
Step-9: rear = rear+1  
Step-10: inside deletef(), if rear=1 and front=-1  
 print “queue empty” & return  
Step-11: front = front+1  
Step-12: print “deleted”  
Step-13: if rear < front  
 rear=front=-1  
Step-14: inside insertf(), if front = 0  
 print “no space in front” & return  
Step-15: print “enter the data”  
Step-16: if front = -1  
 scan for queue[ 0 ]  
 front = rear = 0 & return  
Step-17: scan for queue[ front-1 ] & front = front-1  
Step-18: inside deleter(), if rear=1 and front=-1  
 print “queue empty” & return  
Step-19: rear = rear-1  
Step-20: print “deleted”  
Step-21: if rear < front  
 rear=front=-1  
Step-22: inside display(), declare integer variable i as integer  
Step-23: if rear=-1  
 print “queue empty” & return  
Step-24: print “the elements in the queue are ”  
Step-25: from i = front to i = rear repeat  
 print queue[ i ]  
Step-26: END

**Source code:** #include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
  
int queue[100], front=-1, rear=-1;  
  
void deletef();  
void insertf();  
void deleter();  
void insertr();  
void display();  
  
void main()  
{  
 int in,flag=0;  
 do  
 {  
 printf("Enter the command\n '1' to insert in front\n '2' to delete from front\n '3' to insert rear\n '4' to delete from rear\n '5' to display\n");  
 scanf("%d",&in);  
 switch(in)  
 {  
 case 1: insertf(); break;  
 case 2: deletef(); break;  
 case 3: insertr(); break;  
 case 4: deleter(); break;  
 case 5: display(); break;  
 default: printf("wrong input\n");  
 }  
 printf("enter 1 to continue\n");  
 scanf("%d",&flag);  
 } while(flag==1);  
}  
  
void insertr()  
{  
 if(rear>=99)  
 {  
 printf("queue full\n"); return;  
 }  
 printf("Enter the Data\n");  
 scanf("%d", &queue[rear+1]);  
 if(front==-1)  
 front=0;  
 rear++;  
}  
  
void deletef()  
{  
 if(rear==-1 && front==-1)  
 {  
 printf("queue empty\n"); return;  
 }  
 ++front; printf("deleted\n");  
 if(rear<front)  
 {  
 rear=-1; front=-1;  
 }  
}  
  
void insertf()  
{  
 if(front==0)  
 {  
 printf("no space in front\n"); return;  
 }  
 printf("Enter the Data\n");  
 if(front==-1)  
 {  
 scanf("%d", &queue[0]);  
 front=0; rear=0; return;  
 }  
 scanf("%d", &queue[front-1]);  
 front--;  
}  
  
void deleter()  
{  
 if(rear==-1 && front==-1)  
 {  
 printf("queue empty\n"); return;  
 }  
 --rear; printf("deleted\n");  
 if(rear<front)  
 {  
 rear=-1; front=-1;  
 }  
}  
  
void display()  
{  
 int i;  
 if(rear==-1)  
 {  
 printf("queue empty\n");  
 return;  
 }  
 printf("The elements in the queue are\n");  
 for(i=front;i<=rear;i++)  
 printf("%d, ", queue[i]);  
}

**Input/Output:** Enter the command  
 '1' to insert in front  
 '2' to delete from front  
 '3' to insert rear  
 '4' to delete from rear  
 '5' to display  
1  
Enter the Data  
34  
enter 1 to continue  
1  
Enter the command  
 '1' to insert in front  
 '2' to delete from front  
 '3' to insert rear  
 '4' to delete from rear  
 '5' to display  
3  
Enter the Data  
56  
enter 1 to continue  
1  
Enter the command  
 '1' to insert in front  
 '2' to delete from front  
 '3' to insert rear  
 '4' to delete from rear  
 '5' to display  
5  
The elements in the queue are  
34, 56, enter 1 to continue  
1  
Enter the command  
 '1' to insert in front  
 '2' to delete from front  
 '3' to insert rear  
 '4' to delete from rear  
 '5' to display  
2  
deleted  
enter 1 to continue  
1  
Enter the command  
 '1' to insert in front  
 '2' to delete from front  
 '3' to insert rear  
 '4' to delete from rear  
 '5' to display  
5  
The elements in the queue are  
56, enter 1 to continue  
0