12-11-20 Lab - 6

Date Page 26

Q- WAP for multiple priority queue

Sol

```
#include <stdio.h>
#include <stdlib.h>
#define N 3
int queue[3][N];
int front[3] = {0, 0, 0};
int rear[3] = {-1, -1, -1};
int item, pr;
void fpinsert(int pr)
{   if (rear[pr] == N-1)
        printf("Queue Overflow\n");
    else {
        printf("Enter item\n");
        scanf("%d", &item);
        rear[pr]++;
        queue[pr][rear[pr]] = item; }
    return; }
void fpdelete()
{   int i;
    for (i = 0; i < 3; i++) {
        if(rear[i] == front[i] - 1)
            printf("Empty queue");
        else {
            printf("Deleted = %d of queue = %d",
                queue[i][front[i]], i+1);
```

```
            front[i]++;
            return;}
        }
}
void display()
{   int i,j;
    for(i=0;i<3;i++){
        if(rear[i]==front[i]-1)
            printf("Queue %d is empty",i+1);
        else{
            printf("Queue %d :",i+1);
            for(j=front[i];j<=rear[i];j++){
                printf("%d ",queue[i][j]);}
        }
    }return;
}
int main()
{   int ch;
    while(1){
        printf("1.insert 2.delete 3.display\n");
        printf("Enter choice\n");
        scanf("%d",&ch);
        switch(ch){
            case1:printf("Enter priority:");
                  scanf("%d",&pr);
```

```
        if (pr > 0 && pr < 4)
            fqinsert (pr - 1);
        else
            printf (" Enter proper priority No ");
        break;
    case 2 : fqdelete ();
            break;
    case 3 : display ();
            break;
    case 4 : exit (0);
    }
}
return 0;
}
```

O/p →

1: insert
2: delete
3: display
4: exit
Enter choice : 1
Enter priority : 1
Enter item : 23

1: insert
2: delete

3: display
4: exit
Enter choice: 3
Queue 1: 45
Queue 2 empty
Queue 3 empty

Q → WAP for Input & Output restricted Queue

```
#include <stdio.h>
#include <stdlib.h>
#define size 5
int dequeue_arr[size];
int front=-1, rear=-1;
void insert_rear()
{ int added_item;
  if((front==0 && rear==size-1)||(front
      ==rear+1)) {
      printf("Queue Overflow\n");
      return; }
  if(front==-1) {
      front=0; rear=0; }
  else if(rear==size-1){rear=0;}
  else {rear=rear+1;}
  printf("Enter element:");
  scanf("%d",&added_item);
```

```
    deque_arr[rear] = added_item;
}
void insert_front()
{   int added_item;
    if ((front == 0 && rear == size-1) ||
        (front == rear+1)) {
        printf("Queue overflow\n");
        return; }
    if (front == -1) { front = 0; rear=0; }
    else if (front == 0) { front = size-1; }
    else { front = front-1; }
    printf("Enter element: ");
    scanf("%d", &added_item);
    deque_arr[front] = added_item; }
void delete_front()
{   if (front == -1) {
        printf("Queue Underflow\n");
        return; }
    printf("Element deleted from queue is");
    printf("%d", deque_arr[front]);
    if (front == rear) { front=-1; rear=-1; }
    else{if (front == size-1) { front = 0; }
        else { front = front+1; } } }
void delete_rear()
{   if (front == -1) {
```

```
        printf("Queue Underflow");
        return;}
    printf("Element deleted : %d", deque_arr[rear]);
    if(front == rear){front=-1; rear=-1;}
    else{if(rear==0){rear = size-1;}
        else{rear = rear-1;}}}
void display_queue()
{   int front_pos = front;
    int rear_pos = rear;
    if(front == -1){
        printf("Empty Queue");
        return;}
    printf("Queue elements : ");
    if(front_pos <= rear_pos){
        while(front_pos <= rear_pos){
            printf("%d", deque_arr[front_pos]);
            front_pos++;}}
    else{while(front_pos <= size-1){
            printf("%d", deque_arr[front_pos]);
            front_pos++;}
        front_pos = 0;
        while(front_pos <= rear_pos){
            printf("%d", deque_arr[front_pos]);
            front_pos++;}}}
void input_queue(){
```

```
int choice;
do {
printf("1. Insert rear 2. delete front 3. delete
rear 4. display 5. quit");
printf("Enter choice");
scanf("%d", &choice);
switch(choice) {
    case 1: insert_rear();
            break;
    case 2: delete_front();
            break;
    case 3: delete_rear();
            break;
    case 4: display_queue();
            break;
    default: exit(0); } } while(choice != 5); }
void output_que()
{ int choice;
    do {
    printf("1. Insertrear 2. Insert front 3. delete front
     4. display 5. quit");
    printf("Enter choice");
    scanf("%d", &choice);
    switch(choice) {
        case 1: insertrear();
                break;
```

```
case 2 : insert_front();
         break;
case 3 : delete_front();
         break;
case 4 : display_queue();
         break;
default : exit(0); } } while(choice != 5); }
int main()
{ int choice;
printf("1. I/p Rest  2. O/p rest \n");
printf("Enter choice : ");
scanf("%d", &choice);
switch(choice) {
    case 1 : input_queue();
             break;
    case 2 : output_que();
             break;
    default : exit(0); }
return 0;
}
```

O/p =>

1. I/P rest 2. O/p rest
Enter choice : 1
1. Insert rear 2. delete front 3. delete rear 4. dis
Enter choice : 1

Date
Page 54

Enter item : 45

1. Insert rear 2 del rear 3 del front 4. display

Enter choice : 1

Enter item : 23

1. Insert rear 2. delete rear 3 del front 4 display

Enter choice : 2

Item deleted : 23

1. Insert rear 2. del rear 3 del front 4 display

Enter ~~item~~ choice : 4

Queue items :

45