

DSA LAB 6

1) Implement an ascending priority queue.

Note: An ascending priority queue is a collection of items into which items can be inserted arbitrarily and from which only the smallest item can be removed. If apq is an ascending priority queue, the operation pqinsert(apq,x) inserts element x into apq and pqmindelete(apq) removes the minimum element from apq and returns its value.

Solution:

```
#include<stdio.h>
#include<stdlib.h>
#include<stdbool.h>
#define max 3
typedef struct queue{
    int front,rear;
    int arr[max];
}queue;
bool isFull(queue *q){
    return (q->rear==max-1);
}
bool isEmpty(queue *q){
    return q->rear==q->front;
}
void pqinsert(queue *q,int data){
    q->rear++;
    for(int i=q->front+1;i<=q->rear;i++)
        if(data<=q->arr[i]){
            for(int j=q->rear;j>=i;j--)
                q->arr[j+1]=q->arr[j];
            q->arr[i]=data;
            return;
        }
    q->arr[++q->rear]=data;
}
void pqmindelete(queue *q){
    for(int i=q->front+1;i<=q->rear;i++)
        q->arr[i]=q->arr[i+1];
    q->rear--;
}
void display(queue *q){
```

```

        for(int i=q->front+1;i<=q->rear;i++)
            printf("%d ",q->arr[i]);
        printf("\n");
    }
    int main(){
        printf("Yashas Kamath; 200905132; Rno: 20\n");
        queue q;
        q.front=-1;
        q.rear=-1;
        int opt,ele;
        char cont='y';
        while(cont=='y'){
            printf("Enter the choice\n1.Insert an element\n2.Delete an element\n3.Display the elements\n");
            scanf("%d",&opt);
            switch(opt){
                case 1:
                    if(isFull(&q)){
                        printf("The queue is full!\n");
                        break;
                    }
                    printf("Enter the element ");
                    scanf("%d",&ele);
                    pqinsert(&q,ele);
                    break;
                case 2:
                    if(isEmpty(&q)){
                        printf("The queue is empty!\n");
                        break;
                    }
                    printf("The deleted element is %d\n",q.arr[q.front+1]);
                    pqmindelete(&q);
                    break;
                case 3:
                    printf("The array elements are \n");
                    display(&q);
                    break;
                default: printf("Invalid option\n");
            }
            printf("Do you want to continue? y/n ");
            scanf(" %c",&cont);
        }
    }

```

```
    printf("Thankyou! Visit again!");  
}
```

```
Student@dblab-hp-37:~/Documents/200905132/lab6$ gcc -o l6q1 l6q1.c  
Student@dblab-hp-37:~/Documents/200905132/lab6$ ./l6q1  
Yashas Kamath; 200905132; Rno: 20  
Enter the choice  
1.Insert an element  
2.Delete an element  
3.Display the elements  
1  
Enter the element 6  
Do you want to continue? y/n y  
Enter the choice  
1.Insert an element  
2.Delete an element  
3.Display the elements  
1  
Enter the element 5  
Do you want to continue? y/n y  
Enter the choice  
1.Insert an element  
2.Delete an element  
3.Display the elements  
1  
Enter the element 7  
Do you want to continue? y/n y  
Enter the choice  
1.Insert an element  
2.Delete an element  
3.Display the elements  
1  
The queue is full!  
Do you want to continue? y/n y  
Enter the choice  
1.Insert an element  
2.Delete an element  
3.Display the elements  
3  
The array elements are  
5 6 7
```

```

5
The array elements are
5 6 7
Do you want to continue? y/n y
Enter the choice
1.Insert an element
2.Delete an element
3.Display the elements
2
The deleted element is 5
Do you want to continue? y/n y
Enter the choice
1.Insert an element
2.Delete an element
3.Display the elements
2
The deleted element is 6
Do you want to continue? y/n y
Enter the choice
1.Insert an element
2.Delete an element
3.Display the elements
2
The deleted element is 7
Do you want to continue? y/n y
Enter the choice
1.Insert an element
2.Delete an element
3.Display the elements
2
The queue is empty!
Do you want to continue? y/n n
Thankyou! Visit again!Student@dblab-h

```

2) Implement a queue of strings using an output restricted dequeue (no deleteRight). Note: An output-restricted deque is one where insertion can be made at both ends, but deletion can be made from one end only, whereas an input-restricted deque is one where deletion can be made from both ends, but insertion can be made at one end only.

Solution:

```

#include <stdio.h>
#include<stdbool.h>
#include<stdlib.h>
#define MAX 3
typedef struct dequeue{
char **data;
int rear,front;
}dequeue;
void initialize(dequeue *P){
P->rear=-1;

```

```

P->front=-1;
}
bool empty(dequeue *P){
return(P->rear==-1);
}
bool full(dequeue *P){
return((P->rear+1)%MAX==P->front);
}
void enqueueR(dequeue *P){
    char *x=(char*)calloc(25,sizeof(char));
    printf("\nEnter the element to be inserted:");
    scanf("%s",x);
    if(empty(P)){
        P->rear=0;
        P->front=0;
        P->data[0]=x;
    }
    else{
        P->rear=(P->rear+1)%MAX;
        P->data[P->rear]=x;
    }
}
void enqueueF(dequeue *P){
    char *x=(char*)calloc(25,sizeof(char));
    printf("\nEnter the element to be inserted:");
    scanf("%s",x);
    if(empty(P)){
        P->rear=0;
        P->front=0;
        P->data[0]=x;
    }
    else{
        P->front=(P->front-1+MAX)%MAX;
        P->data[P->front]=x;
    }
}
char* dequeueF(dequeue *P){
    char *x;
    x=P->data[P->front];
    if(P->rear==P->front)
        /*delete the last element */
        initialize(P);
}

```

```

    else
    P->front=(P->front+1)%MAX;
    return(x);
}
void print(dequeue *P){
    if(empty(P)){
        printf("\nQueue is empty!!");
        return;
    }
    int i=P->front;
    while(i!=P->rear){
        printf("\n%s",P->data[i]);
        i=(i+1)%MAX;
    }
    printf("\n%s\n",P->data[P->rear]);
}
int main(){
    printf("Yashas Kamath ; 200905132 ; Rno:20 \n");
    int i,op;
    dequeue q;
    initialize(&q);
    q.data=(char**)calloc(MAX,sizeof(char*));
    do{
        printf("\n1.Insert(rear)\n2.Insert(front)\n3.Delete(front)");
        printf("\n4.Print\n5.Exit\nEnter your choice:");
        scanf("%d",&op);
        switch(op){
            case 1: if(full(&q)){
                printf("\nQueue is full!!");
                break;
            }
            enqueueR(&q);
            break;
            case 2: if(full(&q)){
                printf("\nQueue is full!!");
                break;
            }
            enqueueF(&q);
            break;
            case 3: if(empty(&q)){
                printf("\nQueue is empty!!");
                break;
            }

```

```
}  
printf("\nElement deleted is %s\n",dequeueF(&q));  
break;  
case 4: print(&q);  
}  
}while(op!=5);  
return 0;  
}
```


Yashas Kamath ; 200905132 ; Rno:20

1.Insert(rear)

2.Insert(front)

3.Delete(front)

4.Print

5.Exit

Enter your choice:1

Enter the element to be inserted:Avinash

1.Insert(rear)

2.Insert(front)

3.Delete(front)

4.Print

5.Exit

Enter your choice:2

Enter the element to be inserted:Komal

1.Insert(rear)

2.Insert(front)

3.Delete(front)

4.Print

5.Exit

Enter your choice:2

Enter the element to be inserted:Prakash

1.Insert(rear)

2.Insert(front)

3.Delete(front)

4.Print

5.Exit

Enter your choice:2

Queue is full!!

1.Insert(rear)

2.Insert(front)

3.Delete(front)

4.Print

5.Exit

Enter your choice:4

Enter your choice:4

Prakash
Komal
Avinash

1.Insert(rear)
2.Insert(front)
3.Delete(front)
4.Print
5.Exit

Enter your choice:2

Queue is full!!

1.Insert(rear)
2.Insert(front)
3.Delete(front)
4.Print
5.Exit

Enter your choice:3

< Element deleted is Prakash

1.Insert(rear)
2.Insert(front)
3.Delete(front)
4.Print
5.Exit

Enter your choice:3

Element deleted is Komal

1.Insert(rear)
2.Insert(front)
3.Delete(front)
4.Print
5.Exit

Enter your choice:3

Element deleted is Avinash

1.Insert(rear)
2.Insert(front)
3.Delete(front)
4.Print
5.Exit
Enter your choice:3

Queue is empty!!

1.Insert(rear)
2.Insert(front)
3.Delete(front)
4.Print
5.Exit

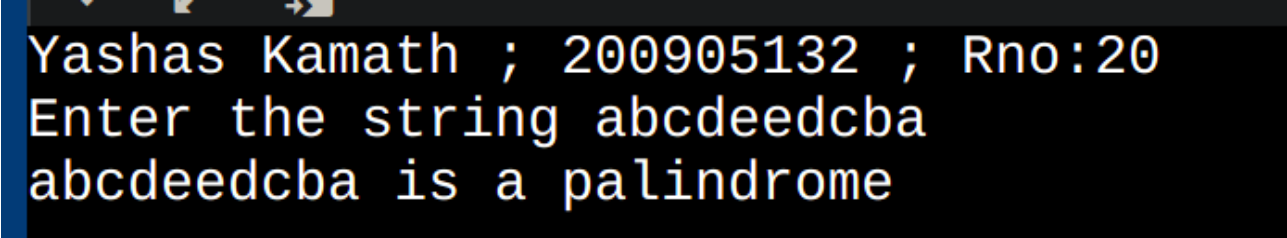
Enter your choice:5

3) Write a program to check whether given string is a palindrome using a dequeue.

Solution:

```
#include <stdio.h>
#include<stdbool.h>
#include<stdlib.h>
#define max 25
typedef struct dequeue{
    char arr[max];
    int front,rear;
}dequeue;
bool isFull(dequeue *q){
    return (q->rear==max-1) && (q->front==-1);
}
bool isEmpty(dequeue *q){
    return q->front==q->rear;
}
void enqrear(dequeue *q,char x){
    q->arr[++q->rear]=x;
}
void enqfront(dequeue *q,int x){
    q->arr[q->front--]=x;
}
char deqfront(dequeue *q){
    return q->arr[++q->front];
}
char deqrear(dequeue *q){
    return q->arr[q->rear--];
}
int main()
{
    printf("Yashas Kamath ; 200905132 ; Rno:20 \n");
    dequeue q;
    char s[max];
    int i;
    q.front=-1;
    q.rear=-1;
    printf("Enter the string ");
    scanf("%s",s);
    for(i=0;s[i]!='\0';i++)
        enqrear(&q,s[i]);
```

```
for(int j=0;j<i/2;j++)  
if(deqrear(&q)!=deqfront(&q)){  
    printf("%s is not a palindrome",s);  
    return 0;  
}  
printf("%s is a palindrome",s);  
return 0;  
}
```

A screenshot of a terminal window with a black background and yellow text. The text displays the name and ID of the user, the string input, and the program's output.

Yashas Kamath ; 200905132 ; Rno:20
Enter the string abcdeedcba
abcdeedcba is a palindrome