

**19BIT0292**

**Bhaumik Tandan**

**LAB SHEET**

**DATA STRUCTURES**

**AND**

**ALGORITHMS**

**LABORATORY**

CSE2011

L57+L58

**Q1) Implementation of double ended queue using array.**

**CODE**

#include<stdio.h>

#define s 20

int ar[s],f=0,r=-1,e;

add\_rear(int a)

{

if(e==s){

printf("\nQueue Full");

return 1;

}

r=(r+1)%s;

ar[r]=a;

e++;

printf("\n%d added to rear",a);

}

add\_front(int a)

{

if(e==s){

printf("\nQueue Full");

return 1;

}

f-=1;

if(f==-1)

f=s-1;

ar[f]=a;

e++;

printf("\n%d added to front",a);

}

del\_front()

{

if(e==0)

{

printf("\nQueue Empty");

return;

}

e--;

int t=f;

f=(f+1)%s;

return ar[t];

}

del\_rear()

{

if(e==0)

{

printf("\nQueue Empty");

return;

}

e--;

int t=r;

r-=1;

if(r==-1)

r=s-1;

return ar[t];

}

disp()

{

if(e==0)

{

printf("\nQueue Empty");

return;

}

printf("\nFront ---> ");

int i=f;

while (i!=r)

{

printf("%d ",ar[i]);

i=(i+1)%s;

}

printf("%d ---> Rear",ar[i]);

}

main()

{

add\_rear(3);

add\_front(23);

add\_rear(94);

disp();

printf("\n %d deleted from front",del\_front());

add\_rear(232);

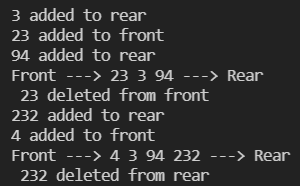
add\_front(4);

disp();

printf("\n %d deleted from rear",del\_rear());

}

**OUTPUT**

****

# [**CLICK HERE FOR GITHUB LINK OF WHOLE SOURCE CODE**](https://github.com/Bhaumik-Tandan/DSA_CODES/blob/master/queue/double_ended_queue.c)

**Q2) Implementation of doubly linked list.**

**CODE**

#include<stdio.h>

struct node

{

int d;

struct node \*n;

struct node \*p;

} \* h;

ins\_r(int n)

{

struct node \*t = (struct node \*)malloc(sizeof(struct node));

t->d = n;

t->n = 0;

printf("\n%d inserted at rear",n);

if (!h){

h = t;

return;

}

struct node \*p = h;

while (p->n)

p = p->n;

p->n = t;

t->p=p;

}

del\_r()

{

if (!h || !(h->n))

{

h=0;

return;

}

struct node \*p = h;

while (p->n->n)

p = p->n;

p->n=0;

}

ins\_f(int n)

{

struct node \*t = (struct node \*)malloc(sizeof(struct node));

printf("\n%d inserted at front",n);

t->d = n;

t->n = h;

t->p=0;

if(h)

h->p=t;

h=t;

}

disp()

{

struct node \*t = h;

printf("\n");

while (t)

{

printf("%d ",t->d);

t = t->n;

}

}

ins\_p(int n,int po)

{

struct node \*t = (struct node \*)malloc(sizeof(struct node));

t->d = n;

t->n = 0;

printf("\n%d inserted at position %d",n,po);

if (!h)

{

h = t;

return;

}

struct node \*p = h;

while (p->n && --po>0)

p = p->n;

p->n = t;

t->p=p;

}

del\_f()

{

if(!h)

return;

h=h->n;

if(h)

h->p=0;

}

del\_p(int po)

{

if (!h || !(h->n))

{

h = 0;

return;

}

struct node \*p = h;

while (p->n->n && --po>1)

p = p->n;

struct node \*t=p->n;

p->n=t->n;

t->n->p=p;

}

main()

{

ins\_r(21);

ins\_f(42);

ins\_f(4290);

ins\_r(90);

disp();

del\_r();

disp();

del\_f();

disp();

ins\_f(426);

ins\_r(490);

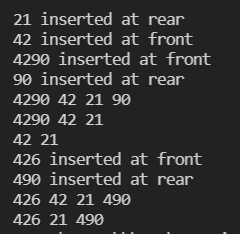
disp();

del\_p(2);

disp();

}

**OUTPUT**

****

# [CLICK HERE FOR GITHUB LINK OF WHOLE SOURCE CODE](https://github.com/Bhaumik-Tandan/DSA_CODES/blob/master/linked list/doubly_linked_list.c)

**Q3) Realization of double ended queue with doubly linked list.**

**CODE**

#include <stdio.h>

struct node

{

int d;

struct node \*n;

struct node \*p;

} \* f, \*r;

struct node \*make\_node(int a, struct node \*n, struct node \*p)

{

struct node \*t = (struct node \*)malloc(sizeof(struct node));

t->d = a;

t->n = n;

t->p = p;

return t;

}

add\_rear(int a)

{

printf("\n%d inserted at the rear",a);

if (!f)

{

f=r=make\_node(a,0,0);

return;

}

struct node \*p = f;

while (p->n)

p = p->n;

p->n=make\_node(a,0,p);

r=p->n;

}

add\_front(int a)

{

printf("\n%d inserted at the front",a);

if (!f)

{

f=r=make\_node(a,0,0);

return;

}

f=make\_node(a,f,0);

f->n->p=f;

}

del\_front()

{

if(!f)

return;

int a=f->d;

f=f->n;

if(f)

f->p=0;

return a;

}

del\_rear()

{

if(!f)

return;

int a=r->d;

r=r->p;

if(r)

r->n=0;

return a;

}

disp()

{

struct node \*t = f;

printf("\n");

while (t)

{

printf("%d ",t->d);

t = t->n;

}

}

main()

{

add\_rear(3);

add\_front(23);

add\_rear(94);

disp();

printf("\n %d deleted from front",del\_front());

add\_front(4);

add\_rear(232);

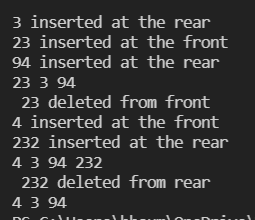
disp();

printf("\n %d deleted from rear",del\_rear());

disp();

}

**OUTPUT**



# [CLICK HERE FOR GITHUB LINK OF WHOLE SOURCE CODE](https://github.com/Bhaumik-Tandan/DSA_CODES/blob/master/queue/doubly_endedque_using_dll.c)

**Q4) Preform Linear Search.**

**CODE**

#include<stdio.h>

search(int p,int \*a,int s)

{

if(p==0 || a[p-1]==s)

return p-1;

return search(p-1,a,s);

}

main()

{

int a[]={1,2,3,4,5,6,7,8};

printf("\n%d found at %d",8,search(8,a,8));

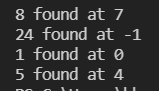
printf("\n%d found at %d",24,search(8,a,24));

printf("\n%d found at %d",1,search(8,a,1));

printf("\n%d found at %d",5,search(8,a,5));

}

**OUTPUT**

****

# [CLICK HERE FOR GITHUB LINK OF WHOLE SOURCE CODE](https://github.com/Bhaumik-Tandan/DSA_CODES/blob/master/search/linear_search.c)

**Q5) Preform Binary Search.**

**CODE**

#include<stdio.h>

search(int st,int en,int \*a,int s)

{

if(st+1>=en)

return -1;

int m=st+(en-st)/2;

if(a[m]==s)

return m;

if(a[m]>s)

return search(st,m,a,s);

if(a[m]<s)

return search(m,en,a,s);

}

main()

{

int a[]={1,2,3,4,5,6,7,8,10,11,12};

printf("\n%d found at %d",4,search(0,12,a,4));

printf("\n%d found at %d",5,search(0,12,a,5));

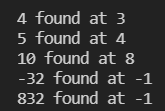
printf("\n%d found at %d",10,search(0,12,a,10));

printf("\n%d found at %d",-32,search(0,12,a,-32));

printf("\n%d found at %d",832,search(0,12,a,832));

}

**OUTPUT**

****

# [CLICK HERE FOR GITHUB LINK OF WHOLE SOURCE CODE](https://github.com/Bhaumik-Tandan/DSA_CODES/blob/master/search/binary_search.c)