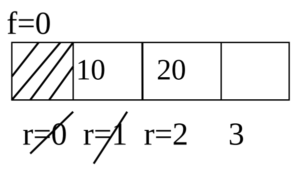
Dr. Ohidujjaman Tuhin
Assistant Professor
Dept. of CSE, UIU

Basic of Queue

Queue is a First in First out (FIFO) data structure. The element which will insert first will be deleted first due to deletion. It consists of two parts such as 'front' (f) and 'rear '(r). At the 'rear' end insertion and the 'front' end deletion operation should be performed.

Example:



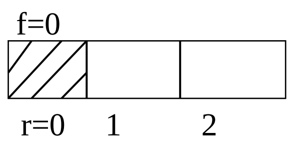
f= front (deletion end)
r=rear (insertion end)

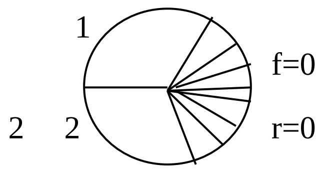
Queue Operation

There are two types of Queue operation like insertion and deletion. The insertion and deletion operations of queue are pictorially represented in the following:

Suppose queue size , m=2; and the instructions are insert(10), insert(20), delete( ), insert(30) insert(40), delete( ), delete( ), delete( ).

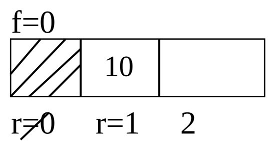
Initial:

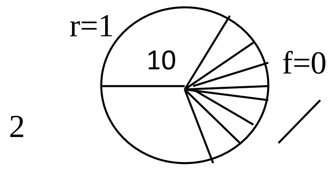




When queue is empty, front and rear are at the same position.

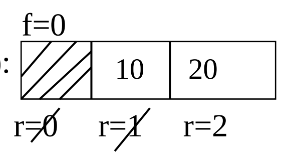
insert(10):

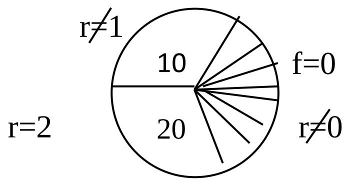




In circular fashion the position of rear(r):
r=(r+1) mod (m+1)
=(0+1) mod (2+1)
= 1 mod 3
=1

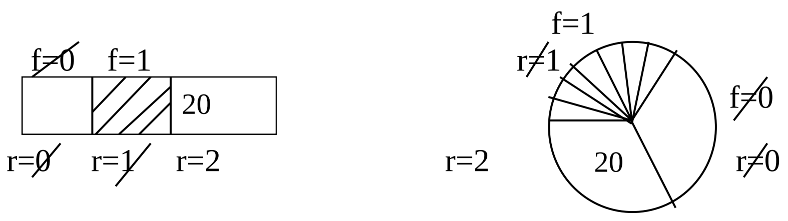
insert(20):





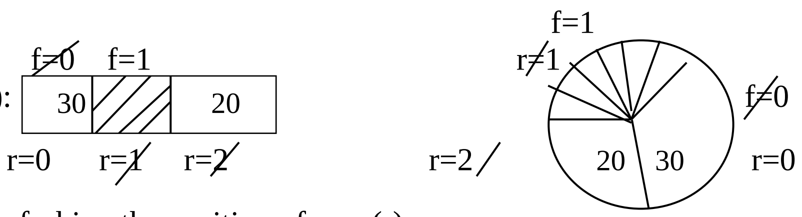
In circular fashion the position of rear(r):
r=(r+1) mod (m+1)
 =(1+1) mod (2+1)
 = 2 mod 3
 =2

delete( ):



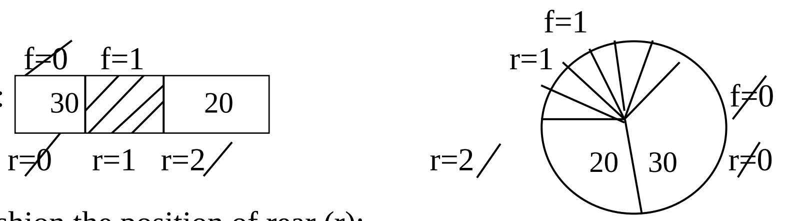
In circular fashion the position of front (f):
f=(f+1) mod (m+1)
 =(0+1) mod (2+1)
 = 1 mod 3
 =1

insert(30):



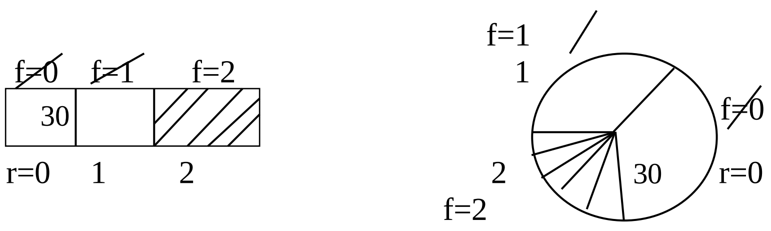
In circular fashion the position of rear (r):
r=(r+1) mod (m+1)
 =(2+1) mod (2+1)
 = 3 mod 3
 =0

insert(40):



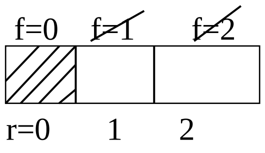
In both fashion the position of rear (r):
r=(r+1) mod (m+1)
 =(0+1) mod (2+1)
 = 1 mod 3
 =1= f
When front and rear are at the same position, the queue is full or overflows.

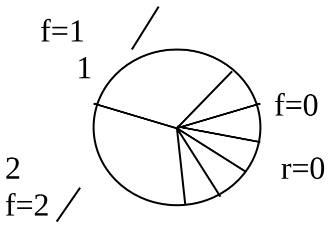
delete ( ):



In both fashion the position of front (f):
f=(f+1) mod (m+1)
  =(1+1) mod (2+1)
  = 2 mod 3
  =2

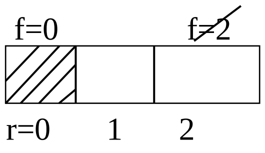
delete ( ):

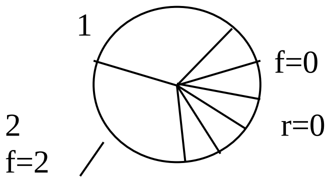




In both fashion the position of front (f):
f= (f+1) mod (m+1)
  = (2+1) mod (2+1)
  = 3 mod 3
  = 0

delete ( ):





In both fashion the position of front (f):
f= (f+1) mod (m+1)
  = (0+1) mod (2+1)
  = 1 mod 3
  = 1
So, f=1 position, there is no value to be deleted at all. That means the queue is underflow or empty. Otherwise, when queue is empty, front and rear are at the same position and nothing for deleting.

Implementation of Queue using Array

Insertion:

```
        r =(r+1) %( M+1);                    // M queue size
        if(f= =r)                            // f= front(delete), r = rear(insert)
        {
                printf("\n Queue is Overflow!!");
        }
        else
        {
                queue[r]=x;
                printf("\n Inserted Item:%d", x);
        }
```

Deletion:

```
        if(f= =r)
        {
                printf("\n Queue is underflow!!");
        }
        else
        {
                f=(f+1)%(M+1);
                x=queue[f];
                printf("\nDeleted value:%d",x);
                queue[f]=NULL;
        }
```

Write a C program for insert and delete operation of queue using array:

```
#include<stdio.h>
#define M 3
int r=0;
int f=0;
int queue[M]={0};
int insert(int x);
void deletion();
int main( ){
        int i,x;
        for(i=0;i<M+1;i++)
        {
        printf("Enter the  value to be inserted:");
        scanf("%d",&x);
        insert(x);
        printf("\n");
        }
   printf(" Inserted Item: ");
   for(i=1;i<M+1;i++)
        {
            printf("  %d",queue[i]);
            }
   printf("\n");
            for(i=0;i<M+1;i++)
            {
   deletion( );
            }
            printf("\n");
            return 0;
}
int insert(int x){
    r=(r+1) %(M+1);
            if(f==r)
            {
                        printf("\n Queue is Overflow!!");
            }
            else
            {
                        queue[r]=x;
                        //printf("\n Inserted Item:%d",queue[r]);
            }
            return x;
}
 void deletion( ){
             int x;
             int r=M;
            if(f==r)
            {
                        printf("\n Queue is underflow!!");
            }
            else
            {
                        f=(f+1)%(M+1);
                        x=queue[f];
                        printf("\nDeleted value:%d",x);
                        queue[f]=NULL;
            }
return ;
}
```

Output on monitor:

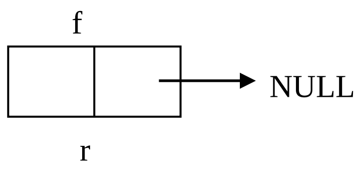
```
Enter the  value to be inserted:10
Enter the  value to be inserted:20
Enter the  value to be inserted:30
Enter the  value to be inserted:40
Queue is Overflow!!
Inserted Item:   10  20  30
Deleted value:10
Deleted value:20
Deleted value:30
Queue is underflow!!
```

Implementation of Queue using Linked Lists

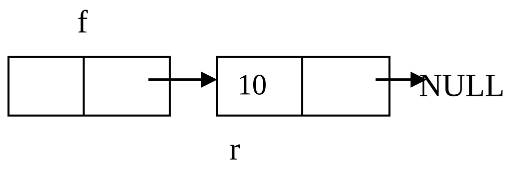
There are two types of Queue operation like insertion and deletion. The insertion and deletion operations of queue are pictorially represented in the following using linked lists:

Suppose the instructions are insert(10), insert(20), delete( ), insert(30) insert(40), delete( ), delete( ), delete( ), delete( ).The status of queue using linked lists for each instructions are depicted as follows pictorially.

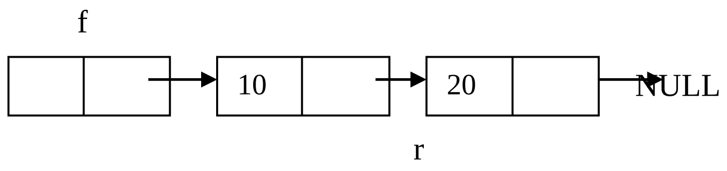
Initial:



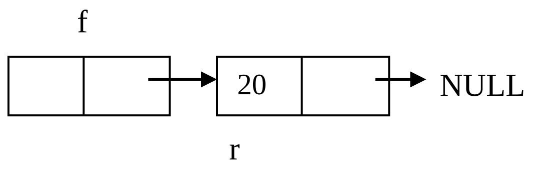
insert(10):



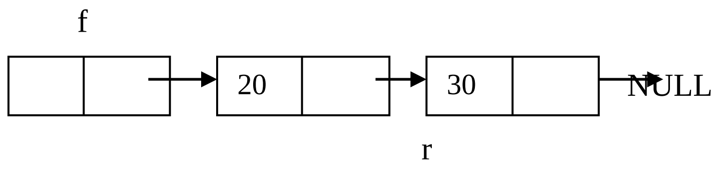
insert(20):



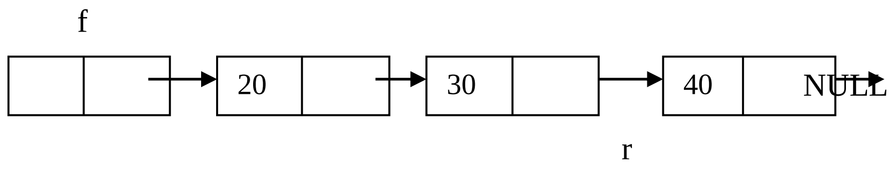
delete( ):



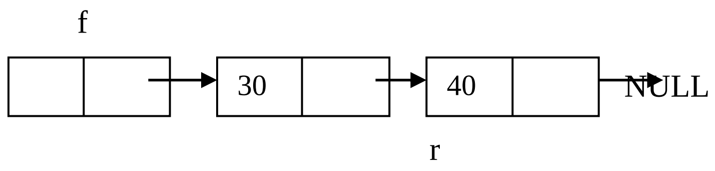
insert (30):



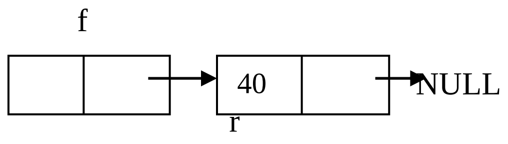
insert(40):

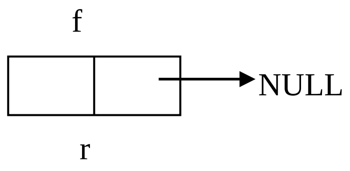


delete( ):



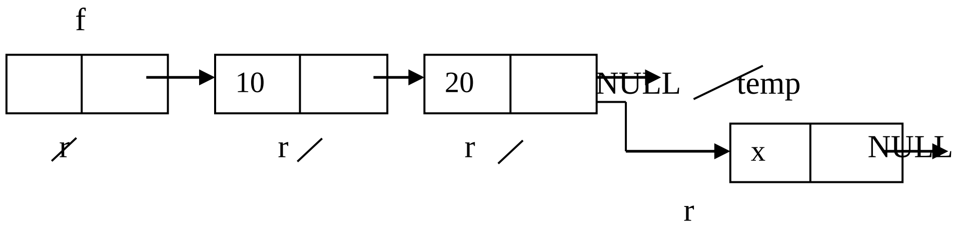
delete( ):



delete( ):


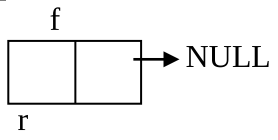
delete( ): “Queue empty” or “queue is underflow”

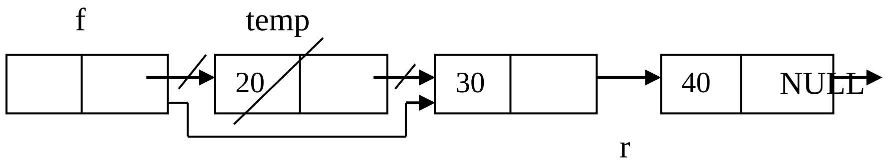
insertion:

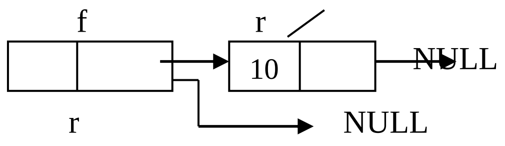


temp=(node*)malloc(sizeof(node));
temp->data=x;
temp->next=NULL;
r->next=temp;
r=r->next;

Deletion:







if(f= =r) {
printf(“\n Queue is underflow”);
}
else
{
temp=f->next;
f->next=temp->next;
free(temp);
if(f->next==NULL)
r=f;
}

Write a C program for insert and delete operation of queue using linked list:

```
#include<stdio.h>
#include<stdlib.h>
#include<ctype.h>
struct list
{
        int data;
        struct list *next;
};
typedef struct list node;
void display (node *head);
int main( )
{
        node *front,*rear,*temp1,*temp;
        char ans;
        int newitem;
        front=NULL;
        rear=NULL;
  do{
     fflush(stdout);
     printf("\n Do you want to create a node(Y/N):=");
     fflush(stdin);
     ans=toupper(getchar());
     if(ans=='Y')
      {
       if(rear= =NULL)
          {
           front=(node*)malloc(sizeof(node));
           printf("\n Front node Empty:");
           front->next=NULL;
           rear=front;
          }
       else
          {
          temp1=(node*)malloc(sizeof(node));
          printf("\n Enter a item into queue:=");
          scanf("%d",&newitem);
           temp1->data=newitem;
           temp1->next=NULL;
           rear->next=temp1;
           rear=temp1;
          display(front);
          }
        }
    }while(ans=='Y');

do{
        printf("\n Do you want to delete a node(Y/N):=");
```

```
            fflush(stdin);
            ans=toupper(getchar());
            if(ans=='Y')
          {
              if(front==rear)
              {
               printf("\nQueue is Underflow");
              }
             else
             {
               printf("\n Deleted  Item:");
                 while(front!=rear)
                {
                        temp=front->next;
                        front->next=temp->next;
                        printf("   %d",temp->data);
                        free(temp);
                          if(front->next==NULL)
                         {
                          rear=front;
                         }
                                 if(front= =rear)
                                {
                               printf("\nQueue is Underflow");
                                }
                    }
                }
            }
       }while(ans=='Y');
            printf("\n\n");
            return 0;
}
          void display(node *head)
             {
            node *temp;
            printf("\n Created list:=");
            temp=head;
            while(temp->next!=NULL)
                       {
                       printf("  %d",temp->next->data);
                        temp=temp->next;
                       }
            return;
             }
```

Output on monitor:

```
Do you want to create a node(Y/N):=y
 Front node Empty:
 Do you want to create a node(Y/N):=y
 Enter a item into queue:=10
 Created list:=  10
 Do you want to create a node(Y/N):=y
 Enter a item into queue:=20
 Created list:=  10  20
 Do you want to create a node(Y/N):=y
 Enter a item into queue:=30
 Created list:=  10  20  30
 Do you want to create a node(Y/N):=n
 Do you want to delete a node(Y/N):=y
 Deleted  Item:   10   20   30
Queue is Underflow
```