PES UNIVERSITY ONLINE

Data Structures and its Applications

Dinesh Singh

Department of Computer Science & Engineering

DATA STRUCTURES AND ITS APPLICATIONS

Queues – Linked List Implementation

Dinesh Singh

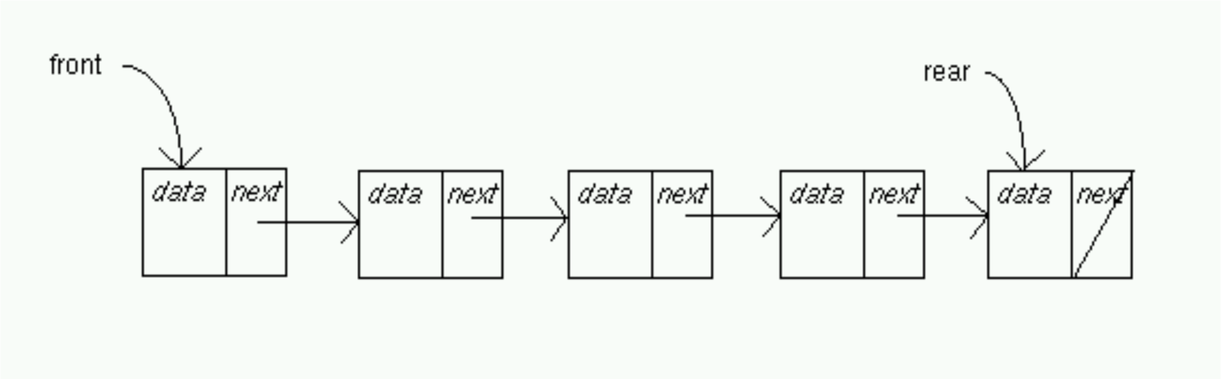
Department of Computer Science & Engineering

Queues - Linked list Implementation

In a linked list implementation two pointers are maintained : front and rear .

- front points to the first item of the queue

- rear points to the last item of the queue



Operations :

- Insert() : adds a new node after the rear and moves rear to the next node
- Remove() : removes the first node and moves front to the next node
- Empty() : Checks if the queue is empty

Queues - Linked list Implementation

Structure of queue

```
struct node
{
  int data;
  struct node *next;
};
struct queue
{
   struct node * front;
   struct node *rear;
};

Struct queue q;
q.front=q.rear = NULL;
```

Queues - Linked list Implementation – Operations

Insert operation

```
Insert(q,x)

p=getnode();
initialise the node
if(q.rear=NULL)
    q.front=p;
else
   next(q.rear) =p;
q.rear = p;
```

remove operation

```
remove(q)

If(empty(q))
   print empty queue
else
  p=q.front;
  x=info(p);
  q.front = next(p);
  if(q.front =NULL)
     q,rear=NULL
  freenode(p);
  return x;
```

Queues - Linked list Implementation – Operations

Insert operation of queue implemented by a linked list

```
void qinsert(struct node * q, int x)
  {
    struct node *temp;

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

     //if this is the first node
    if(q->front==NULL)
      q->front=q->rear=temp;
    else //insert at the end
      {
         q->rear->next=temp;
        q->rear=temp;
       }
   }
```

Queues - Linked list Implementation – Operations

remove operation of a queue implemented by a linked list

```
int qremove(struct queue * q)
   {
        struct node *p;
        int x;
        p=q->front;
        if(p==NULL)
         {
          printf("Empty queue\n");
          return -1;
         }
```

Queues - Linked list Implementation – Operations

```
else
     {
      x=q->data;
      if(q->front==q->rear) //only one node
       q->front=q->rear=NULL;
      else
      {
      q->front=q->next; // move front to next node
      return x;
      }
     free(q);
    }
  }
```

Queues - Linked list Implementation – Operations

```
void qdisplay(struct queue q)
  {
     struct node * f, *r;
     if(q.front==NULL)
       printf("Queue Empty\n");
   else
    {
     f=q.front; r=q.rear;
     while(f!=r)
     {
       printf("%d-> ",f->data);
       f=f->next;
     }
   printf("%d-> ",f->data); // print the last node
   }
 }
```

Queues - Linked list Implementation – Operations

Insert operation in an alternate way

```
void qinsert(int x, struct node **f, struct node **r)
//f and r are pointers to variables front and rear of a queue
  {
   struct node *temp;

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

    //if this is the first node
   if(*f==NULL)
     *f=*r=temp;
   else //insert at the end
      {
        (*r)->next=temp;
         *r=temp;
       }
  }
```

Data Structures and its Applications

Queues - Linked list Implementation – Operations

Remove operation in an alternate way

```
int qdelete(struct node **f, struct node **r)
  {
      struct node *q;
      int x;
      q=*f;
      if(q==NULL)
       {
        printf("Empty queue\n");
        return -1;
       }
else
       {
         x=q->data;
         if(*f==*r) //only one node
           *f=*r=NULL;
         else
         {
         *f=q->next;
         return x;
        }
       free(q);
      }
  }
```

Queues - Linked list Implementation – Operations

Disadvantages of representing queue by a linked list

- A node in linked list occupies more storage than the corresponding element in an array.

- Two pieces of information per element is necessary in a list node, where as only one piece of information is needed in an array implementation

PES UNIVERSITY ONLINE

THANK YOU

Dinesh Singh

Department of Computer Science & Engineering

dineshs@pes.edu

+91 8088654402