A. Tujuan Pembelajaran

- Mahasiswa mampu menjelaskan pengertian queue dan dequeue
- Mahasiswa mampu menjelaskan dan menunjukkan cara pembuatan queue, operasi push dan pop pada array
- Mahasiswa mampu menjelaskan dan menunjukkan program dengan ADT (Abstract Data Type) queue dan dequeue dengan array

B. Dasar Teori

QUEUE

Queue atau antrian adalah suatu kumpulan data yang penambahan elemennya hanya bisa dilakukan pada suatu ujung (disebut dengan sisi belakang atau rear), dan penghapusan atau pengambilan elemen dilakukan lewat ujung yang lain (disebut dengan sisi depan atau front).

Kalau tumpukan dikenal dengan menggunakan prinsip LIFO (Last In First Out), maka pada antrian prinsip yang digunakan adalah FIFO (First In First Out).

Implementasi Antrian dengan Array

Untuk memahami penggunaan antrian dalam array, kita membutuhkan deklarasi antrian, misalnya:

```
#define MAXN 6
typedef enum {NOT_OK, OK} Tboolean;
typedef struct {
    Titem array[MAXN];
    int first;
    int last;
    int number_of_items;
} Tqueue;
```

Dengan deklarasi di atas, elemen antrian dinyatakan dalam tipe integer yang semuanya terdapat dalam struktur. Variabel first menunjukkan posisi elemen pertama dalam *array*, dan variable last menunjukkan posisi elemen terakhir dalam *array*.

Algoritma dari penggalan program di atas adalah:

1. Tentukan elemen yang akan dimasukkan ke dalam antrian (dalam hal ini adalah 6 elemen)
2. Deklarasikan struktur untuk menampung elemen pada antrian
3. Selesai

Untuk menambah elemen baru dan mengambil elemen dari antrian dalam antrian, diperlukan deklarasi berikut ini:

```
void initialize_queue (Tqueue *Pqueue) {
    Pqueue->first = 0;
    Pqueue->last= -1;
    Pqueue->number_of_items = 0;
}
Tboolean enqueue (Tqueue *Pqueue, Titem item) {
    if (Pqueue->number_of_items>=MAXN)
        return(NOT_OK);
    else {
        Pqueue->last++;
        if (Pqueue->last>MAXN-1)
            Pqueue->last=0;
        Pqueue->array[Pqueue->last]=item;
        Pqueue->number_of_items++;
        return (OK);
    }
}

Tboolean dequeue (Tqueue *Pqueue, Titem *Pitem){
    if (Pqueue ->number_of_items == 0)
        return (NOT_OK);
    else {
        *Pitem = Pqueue->array[Pqueue->first++];
        if (Pqueue->first>MAXN-1)
            Pqueue->first=0;
        Pqueue->number_of_items--;
        return(OK);
    }
}
```

Implementasi Antrian dengan Pointer

Untuk mengimplementasikan antrian dengan menggunakan pointer, perhatikan algoritma berikut ini:

1. Tentukan struktur untuk menampung node yang akan dimasukkan pada antrian. Deklarasi struktur pada penggalan program berikut ini:

```
struct queueNode {
    char data;
    struct queueNode *nextPtr;
};
typedef struct queueNode QUEUENODE;
typedef QUEUENODE *QUEUENODEPTR;
QUEUENODEPTR headPtr=NULL, tailPtr=NULL;
```

2. Deklarasikan penambahan elemen baru pada antrian, di mana letaknya adalah paling belakang. Deklarasi penambahan elemen baru tersebut dapat dilihat pada penggalan program berikut ini:

```
void enqueue (QUEUENODEPTR *headPtr, QUEUENODEPTR *tailPtr,
              char value) {
    QUEUENODEPTR newPtr=malloc (sizeof(QUEUENODE));
    if(newPtr!=NULL){
        newPtr->data=value;
        newPtr->nextPtr=NULL;
        if(isEmpty(*headPtr))
            *headPtr=newPtr;
        else
            (*tailPtr)->nextPtr=newPtr;
        *tailPtr=newPtr;
    }
```

3. Lakukan pengecekan terhadap antrian, apakah antrian dalam kosong atau tidak. Kalau kondisi antrian kosong, maka elemen bisa dihapus. Penggalan program berikut ini akan menunjukkan kondisi tersebut.

```
int isEmpty(QUEUENODEPTR headPtr) {
    return headPtr==NULL;
}
```

C. Latihan

```
#include<iostream.h>
#include<stdlib.h>
#define MAX 10

void insert(int queue[],int*rear,int nilai);
void del(int queue[],int*front,int rear,int*nilai);

void main()
{
int queue[MAX];
int front,rear;
int n,nilai;

front=rear=(-1);
do
{
do
{
cout<<"masukkan nilai elemen:";
cin>>nilai;
insert(queue,&rear,nilai);

cout<<endl;
cout<<"tekan 1 untuk melanjutkan:";
cin>>n;
}while(n==1);

cout<<endl;
cout<<"tekan 1 untuk menghapus ssebuah elemen"<<endl;
cin>>n;
```

```
while(n==1)
{
del(queue,&front,rear,&nilai);
cout<<"nilai telah dihapus:"<<nilai<<endl;
cout<<endl;
cout<<"tekan 1 untuk menghapus sebuah elemen:"<<endl;
cin>>n;
}

cout<<endl;
cout<<"tekan 1 untuk melanjutkan:";
cin>>n;
} while(n==1);
}

void insert(int gueue[],int*rear,int nilai)
{
if(*rear<MAX-1)
{
*rear=*rear+1;
gueue[*rear]=nilai;
}
else
{
cout<<"queue penuh,insert tidak dapat dilakukan"<<endl;
exit(0);
}
}
```

```
void del(int queue[],int*front,int rear,int *nilai)
{
if(*front==rear)
{
cout<<"queue kosong,delete tidak dapat dilakukan"<<endl;
exit(0);
}
*front=*front+1;
*nilai=queue[*front];
}
```

D. Tugas Praktikum

```
#include<iostream.h>
#include<conio.h>

class Queue
{
private:
 struct kue
 {
  int data;
  kue*kuebaru;
 };
 kue*tail;
 kue*entry;
 kue*print;
 kue*head;

public:
 Queue();
 void Delete();
 void Insert();
 void cetak();
 void menu();
};

Queue::Queue()
{
 tail=1;
 head=NULL;
}

void Queue::Insert()
{
 int num;
 cout<<"\n\n\n\n\n\t Masukkan data dalam Queue:";
 cin>>num;
 entry=new kue;
 if(tail==1)
```

```
 {
  entry->data=num; //(*entry).data=num
  entry->kuebaru=NULL;//(*entry).kuebaru=null
  tail=entry;
  head=tail;
 }
 else
 {
  entry*data=num;//(*entry).data=num
  entry->kuebaru=NULL;
  tail->kuebaru=entry;//(*entry).kuebaru=entry
  tail=entry;
 }
 cout<<"\n\n\t***"<<num<<"sudah masuk dalam Queue."<<endl;
 getch();//getch:bwan conio
}

void Queue::Delete()
{
 if(head==NULL)
  cout<<"\n\n\n\t***Error:Queue is empty.\n"<<endl;
 else
 {
  int deleted_element=head->data;//(*head).data=deleted_elemnt
  kuebaru*temp;//kue: nm struct, *temp=data yg ditunjuk olh tbl
  temp=head;
  head=head->kuebaru;//(*head).kuebaru=head
  delete temp;
  cout<<"\n\n\n\t***"<<deleted_element<<"dihapus dari queue."<<endl;
 }
 cout<<"\n\n\n\t\t press any key to return to menu";
 getch();
}
```

```
void Queue::cetak()
{
 print=head;
 if(print!=NULL)
  cout<<"\n\n\n\n\t angka-angka yang ada di dalam queue adalah:\n"<<endl;
 else
  cout<<"\n\n\n\n\t***tidak ada yang ditampilkan"<<endl;
  while(print=NULL)//while cek dulu
  {
  cout<<"\t"<<print->data<<endl;
  print=print->kuebaru;
  }
  cout<<"\n\n\n\t\t press any key to return to menu.";
  getch();
}

void Queue::menu()
{
 char Key = NULL;
 do
 {


  cout<<"**Implementasikan Queue**"<<endl;

  cout<<"pilih salah satu menu:"<<endl;

  cout<<"Tekan\'I\' to Masukkan data dalam Queue"<<endl;

  cout<<"Tekan\'D\' to hapus data dari Queue"<<endl;


 cout<<"Tekan\'P'\'to Tampilkan data dari Queue"<<endl;

 cout<<"Tekan\'E\'to Exit"<<endl;
 Input:


 cout<<"Masukkan pilihan:";
 Key = getche();
 if(Key=='e'||Key=='e'||Key=='E')/*untuk fungsi exit*/
  break;
 else if(Key=='i'||Key=='I')
  Insert();
 else if(Key=='d'||Key=='D')
  Delete();
 else if(Key=='p'||Key=='P')
  cetak();
 else
  goto Input;
 }
 while(0);
}

int main()
{
 Queue obj;
 obj.menu();
 return 1;
}
```

E. Tugas

- Silahkan Anda buat sebuah program untuk memasukkan, menghapus, dan mencetak suatu antrian data yang berupa string