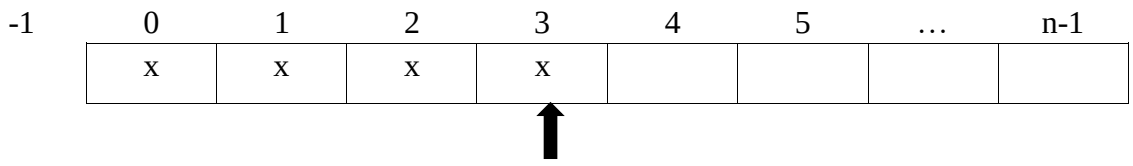
Stack



Top

1. Ciri bahwa stack :
    a. Kosong : apabila Top = -1
    b. Penuh : apabila Top = n - 1
    c. Bisa diisi : apabila Top < n - 1
    d. Ada isinya : apabila Top > - 1

2. Algoritma dasar (dengan deskripsi kata-kata) untuk :
    a. Push :

        infotype x; //deklarasi variabel X

        S.Top ++; //top akan bertambah 1

        S.info[S.Top] = x; //info dari indexnya top berisi variabel X

    b. Pop :

        infotype x; //deklarasi variabel X

        x = S.info[S.Top]; //X akan berisi info dari indexnya top

        S.Top = S.Top - 1; //top akan dikurangi 1

3. Tulis algoritma lengkap (dengan notasi algoritma) untuk :
    a. Push

        Algoritma :

        If (Top != n - 1) then

            Top = Top + 1

            S[Top] = X

        else

Output = “stack penuh”

b. Pop

Algoritma :

```
If (Top = -1) then
    Output = “stack kosong”
Else
 infotype x
 x = S.info[S.Top]
 S.Top = S.Top – 1
```

4. Penggalan program dalam bahasa C++ untuk menginput data melalui keyboard satu persatu dan meng-push data tersebut ke stack sampai stack penuh tak bisa diisi lagi.

   Jawaban :

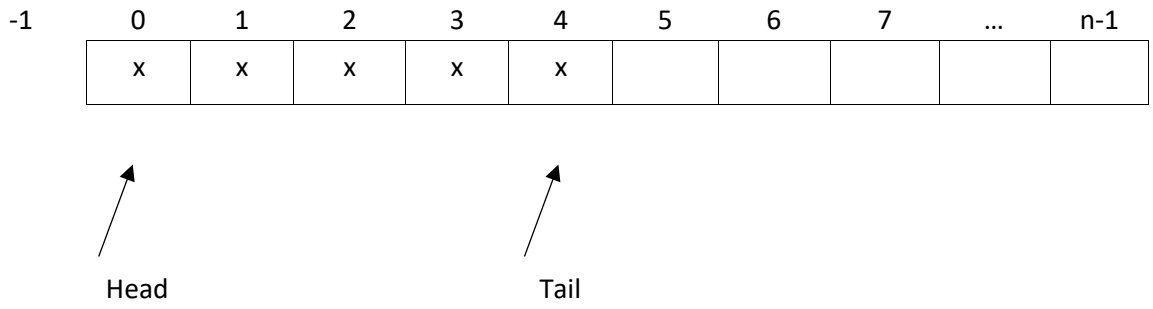
```
void push(stack &S, infotype x){
    if (isFull(S) == false ){
        S.Top ++;
      S.info[S.Top] = x;
}
}
```

5. Tulis penggalan program dalam C++ untuk mengeluarkan isi stack (pop) satu persatu dan mencetaknya sampai stack menjadi kosong.

   Jawaban :

```
infotype pop(stack &S){
    infotype x;
    x = S.info[S.Top];
    S.Top = S.Top - 1;
    return x;
}
```

Queue



6. Ciri bahwa Queue :
    a. Kosong tidak ada isinya => apabila counter = 0
    b. Penuh tidak bisa diisi => apabila counter = n
    c. Bisa diisi => apabila counter < n
    d. Ada isinya => apabila counter > 0

7. Algoritma dasar (dengan deskripsi kata-kata) untuk :
    a. Enqueue

```
Void enqueue(Queue &Q, infotype x) {
        TAIL++;         //tail akan bertambah 1
        Q.info[TAIL] = x;       //info dari indexnya tail berisi variabel X
}
```

    b. Dequeue

```
Void dequeue(Queue &Q) {
        int i = 0;          //deklarasi iterator
        while (i < TAIL) {
                Q.info[i] = Q.info[i + 1];        // indexnya info Queue ditambah 1
                i++;     //i ditambah 1
        }
        TAIL--;          //TAIL dikurangi 1
```

}

8. Tulis algoritma lengkap (dengan notasi algoritma) untuk :

   a. Enqueue

      If (counter < n) then

         (R+1) % n ← R

         X ← Q[R]

      Counter ++

      output (“antrian penuh”)

   b. Dequeue

      If (counter > 0) then

         (F+1) % n ← F

      Counter --

      output (“antrian kosong”)

9. Penggalan program dalam bahasa C++ untuk menginputkan data melalui keyboard satu persatu dan men-enqueue data tersebut ke Queue sampai Queue penuh tak bisa diisi lagi.

```
void enqueue(Queue &Q, infotype x) {
if (isEmptyQueue(Q) == true){
        HEAD = 0;
        TAIL = 0;
        Q.info[TAIL] = x;
} else if (isFull(Q) == true) {
 cout << "QueueFull" << endl;
} else {
        TAIL++;
        Q.info[TAIL] = x;
```

```
}
```

10. Penggalan program dalam bahasa C++ untuk mengeluarkan isi Queue (dequeue) satu persatu dan mencetaknya sampai Queue menjadi kosong.

```
void dequeue(Queue &Q){
        if (HEAD == TAIL){
                HEAD = -1;
                TAIL = -1;
        } else if (isFull(Q) == true) {
                cout << "QueueFull" << endl;
        } else {
                int i = 0;
                while (i < TAIL) {
                        Q.info[i] = Q.info[i + 1];
                        i++;
                }
                TAIL--;
        }
}
```