Queues

- ✓ A queue is a linear list of elements in which…
  - ➢ deletions can take place only at one end, called the *front*, and
  - ➢ insertions can take place only at the other end, called the *rear.*
- ✓ Queues are also called first-in first-out (FIFO) list
- ✓ This contrasts with stacks, which are LIFO

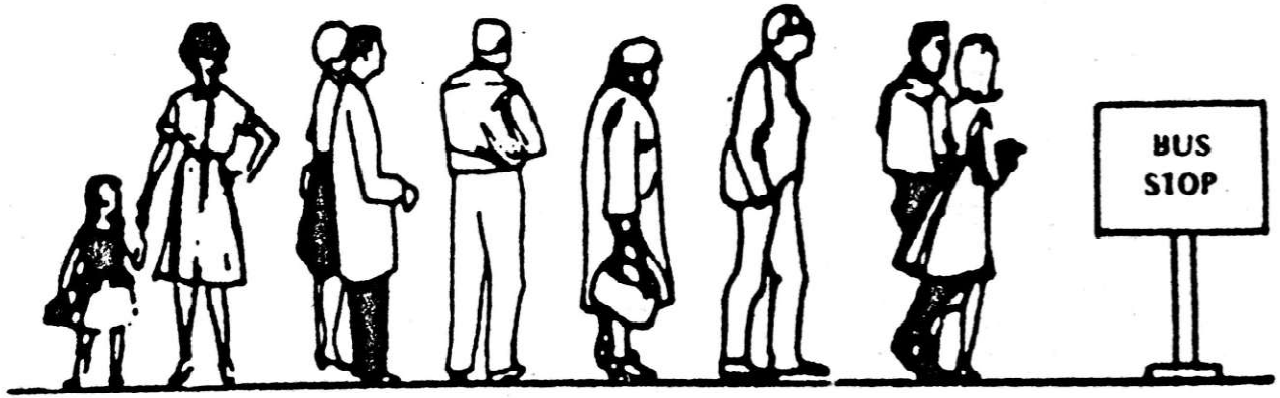
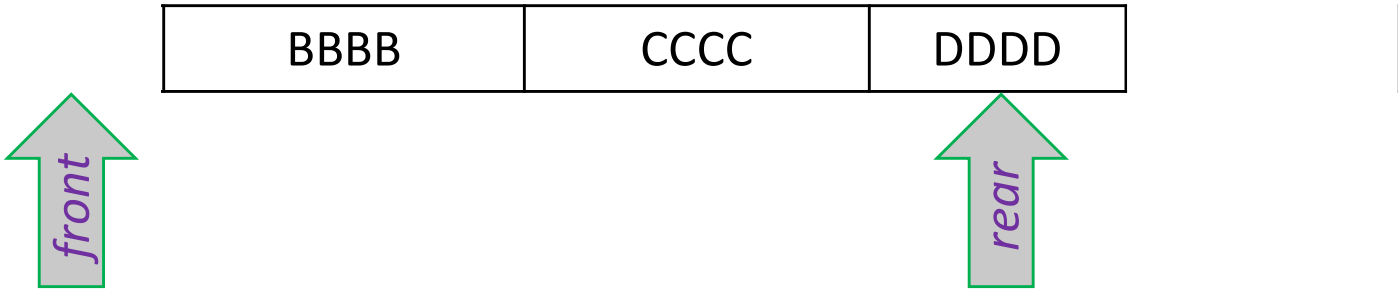


Fig. 6-2 Queue waiting for a bus.

- ✓ An important example of a queue in computer science occurs in a timesharing system in which programs with the same priority form a queue while waiting to be executed.

Queues: Example

Consider a Queue with 4 elements



✓**Suppose an element is deleted from the 4 element queue.**

Which is we can delete??

AAAA

Then which one is considered to be the front element?

BBBB

Suppose EEEE is added to the queue

Then which one is considered to be the rear element?

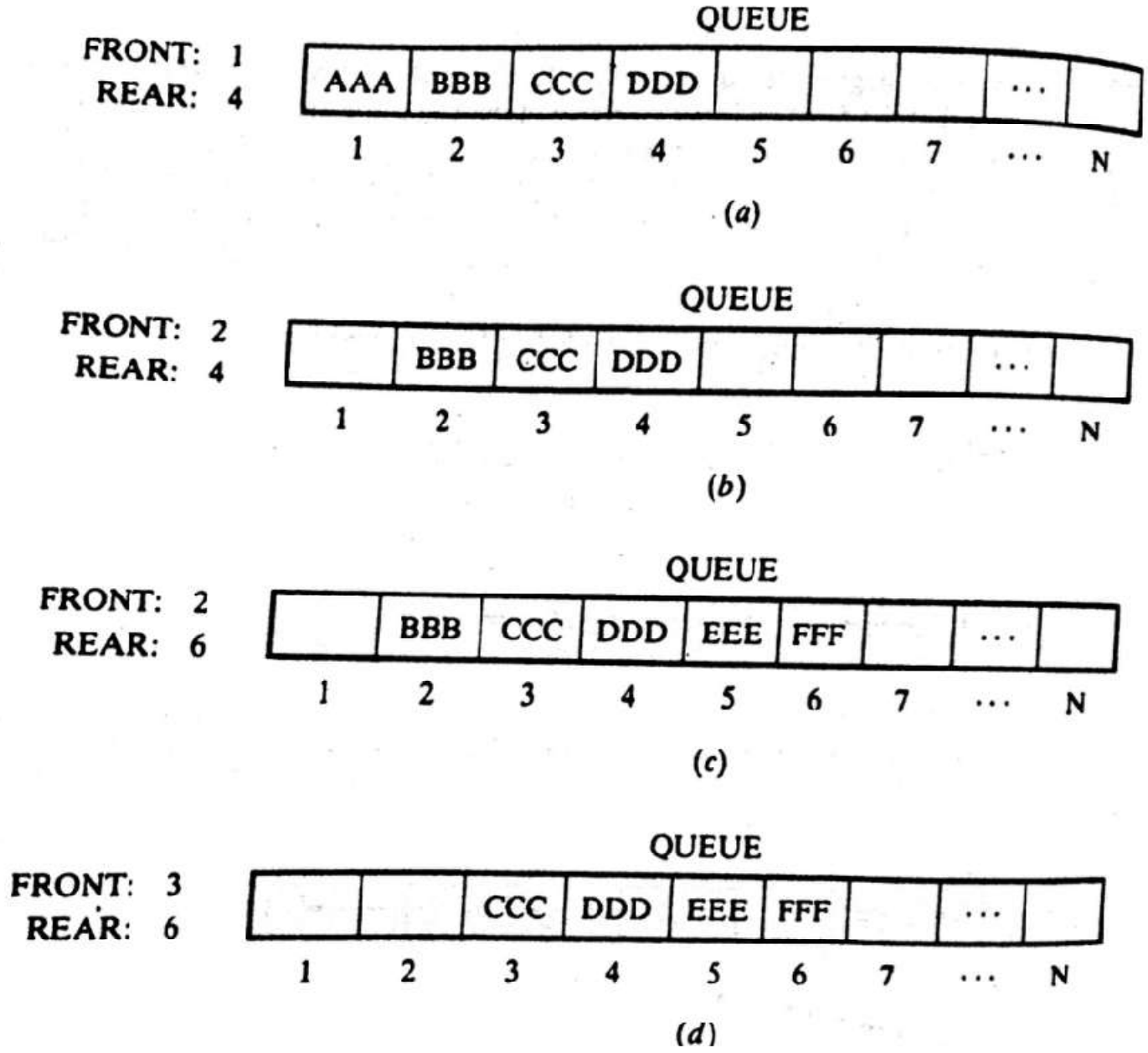
EEEE

Queues: Representation

- ✓ Queues may be represented in the computer in various ways.
- ✓ Usually by means of
  - ➢ One-way list
  - ➢ Linear arrays
- ✓ Each of the queues will be maintained by …
  - ➢ A Linear Array **QUEUE**
  - ➢ Two pointer variables:
    - ○ **FRONT** (containing the location of the front element of the Queue)
    - ○ **REAR** (containing the location of the rear element of the Queue)
- ✓ The condition **FRONT= NULL** will indicate that the Queue is……

EMPTY.

Queues: Representation



Queues: Representation

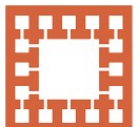
- ✓ After N insertion, the REAR element of the QUEUE will occupy QUEUE [N]
- ✓ This occurs even though the queue itself may not contain many elements

- ✓ Suppose, we want to insert an element **ITEM** into a queue at the time the queue does occupy the last part of the array, that is **REAR=N.**

- ✓ One way to do this is to simply move the entire queue to the beginning of the array changing **FRONT** and **REAR** accordingly.
- ✓ Then insert item as above.

- ✓ This procedure is very expensive
- ✓ The procedure we adopt is to assume that the array **QUEUE is Circular. ie, QUEUE[1] comes after QUEUE[N]**

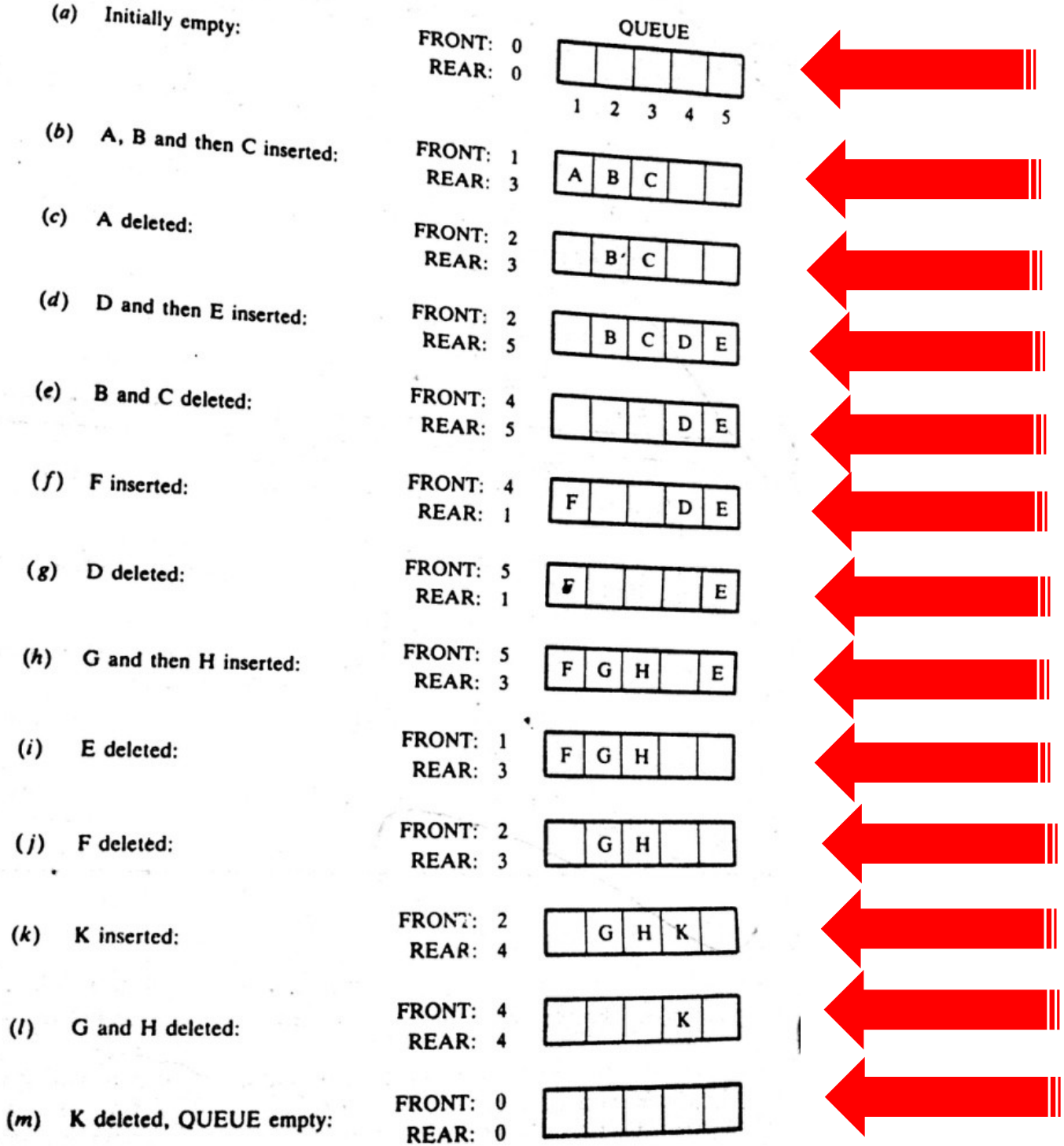
Queues: Representation



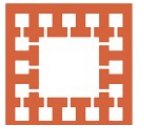
CSE 2103

- ✓ Similarly, if FRONT=N and an element of Queue is deleted.
- ✓ We reset FRONT=1, instead of increasing FRONT to N+1.

Queues: Representation



Queues: Insertion Algorithm



QINSERT (QUEUE, N, FRONT, REAR, ITEM)

Step1. [QUEUE already filled?]

If FRONT=1 and REAR=N, or FRONT=REAR+1, then:

Write: OVERFLOW, and Return

Step2. [Find new value of REAR.]

If **FRONT:=NULL**, then: [Queue initially empty]

Set FRONT:=1 and REAR:=1

Else if **REAR=N**, then:

Set REAR:=1,

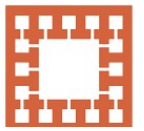
Else:

Set REAR:=REAR+1. [End of If structure]

Step3. **Set QUEUE[REAR]=ITEM** [This inserts new item]

Step4. Return

Queues: Deletion Algorithm



QDELETION (QUEUE, N, FRONT, REAR, ITEM)

Step1. [QUEUE already empty?]

If FRONT=NULL then:

Write: UNDERFLOW, and Return

Step2. **Set** ITEM:= QUEUE[FRONT],

Set3. [Find new value of FRONT.]

If **FRONT:=REAR**, then: [Queue has only one element to start]

Set FRONT:=NULL and REAR:=NULL

Else if **FRONT=N**, then:

Set FRONT:=1,

Else:

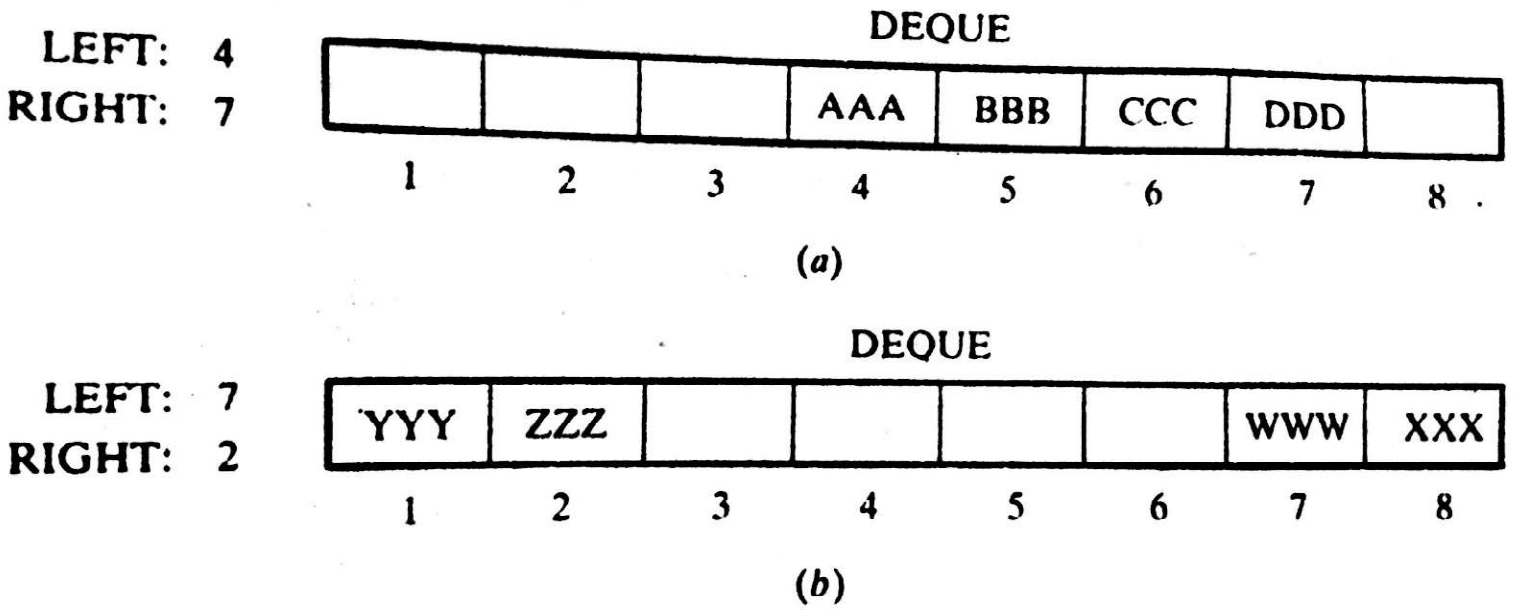
Set FRONT:=FRONT+1.

[End of If structure]

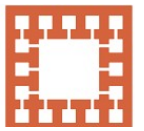
Step4. Return

Deques: Definition

- ✓ A deque is a linear list in which elements can be added or removed at either end but not in the middle.
- ✓ Deque can be maintained by……
  - ➢ a CIRCULAR array
  - ➢ Pointer LEFT-which points left end of the deque
  - ➢ Pointer RIGHT-which points right end of the deque.



Deques: Variation



- ✓ There are TWO variations of a Deque-
  - ➢ An Input-restricted deque, and
  - ➢ An Output-restricted deque
- ✓ This are intermediate between a Deque and Queue

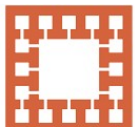
- ✓ An Input-restricted deque is a deque which allow insertion at only one end of the list but allows deletions at both ends of the list
- ✓ An output-restricted deque is a deque which allows deletions at only one end of the list but allows insertions at both ends of the list.

Priority Queues: Definition

A priority queue is a collection of elements such that each elements has been assigned a priority and such that the order in which elements are deleted and processed comes from the following rules:

- An element of higher priority is processed before any element of lower priority
- Two elements with same priority are processed according to the order in which they were added to the queue.

Priority Queues: One-way List Representation



One way to maintain a priority queue in memory is by means of a one-way list, as follows

a) Each node in the list will contain three items of information:
   - An Information field INFO,
   - A priority number PRN, and
   - A link number LINK

b) A node X precedes a node Y in the list

(1) when X has higher priority then Y

(2) When both have the same priority but X was added to the list before Y

❖ Priority numbers will operate in the usual way: the Lower the priority number, the higher the priority.

Priority Queues: Schematic Diagram with 7 elemen

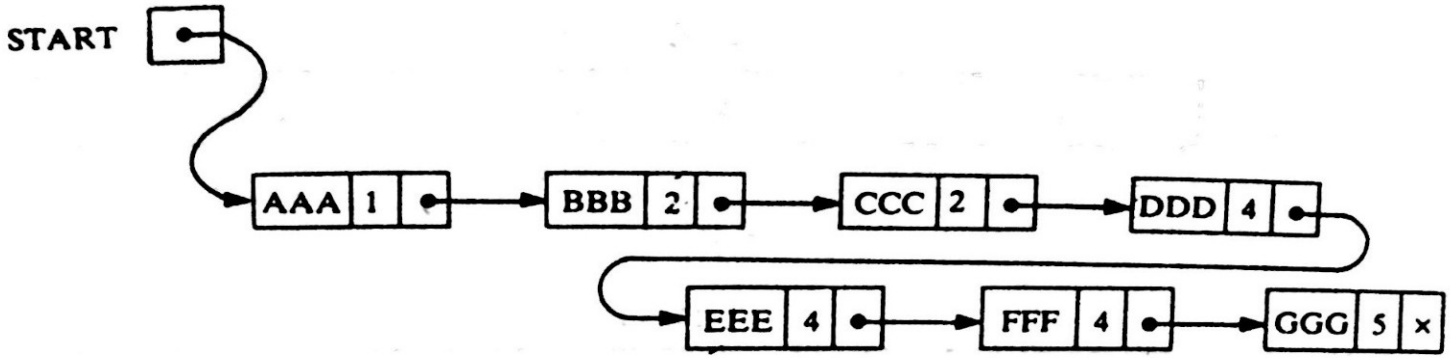


Fig. 6-19

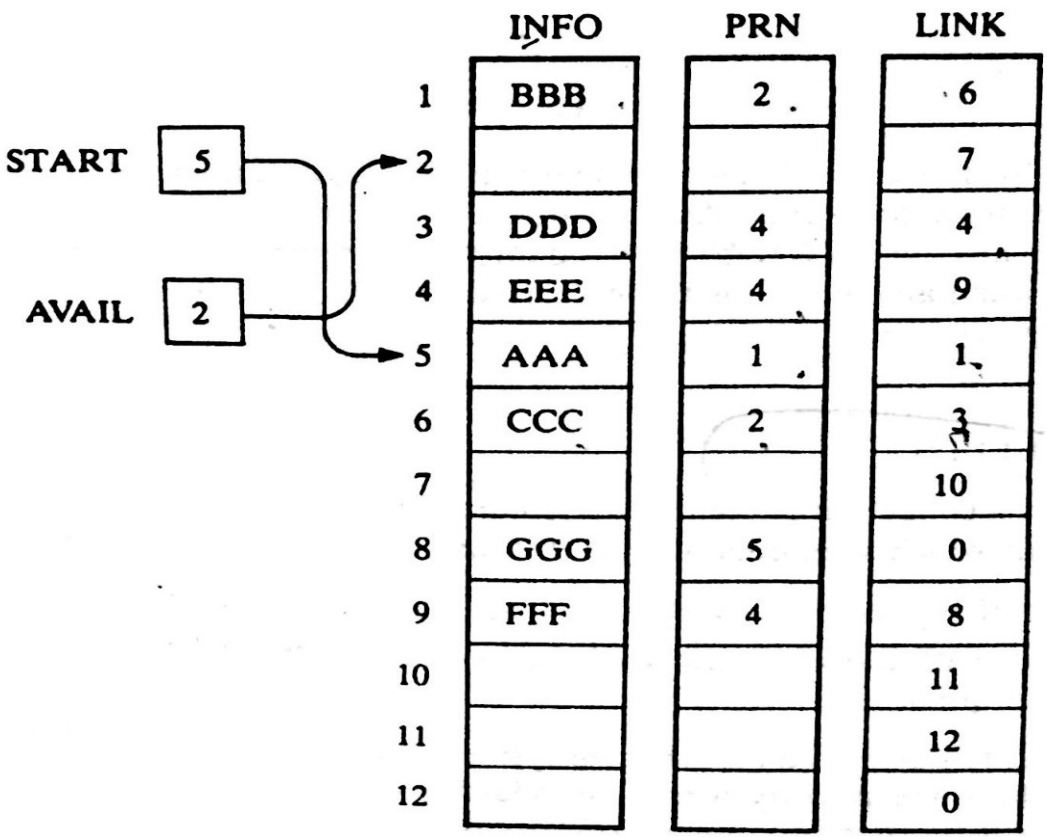
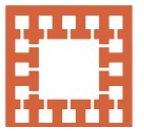


Fig. 6-20

Upcoming Presentation from Students:



- Implement STACK using one QUEUE
- Implements STACK using two QUEUES
- Implement QUEUE using one STACK
- Implement QUEUE using two STACK