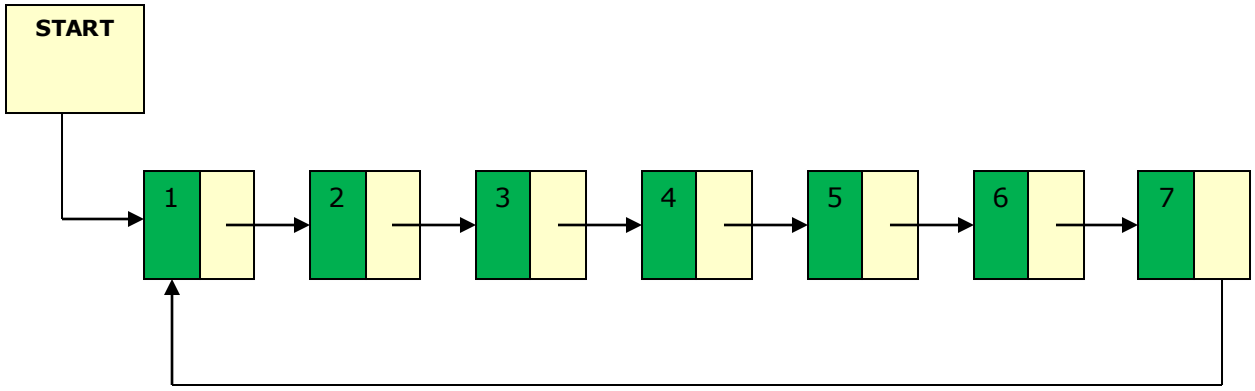
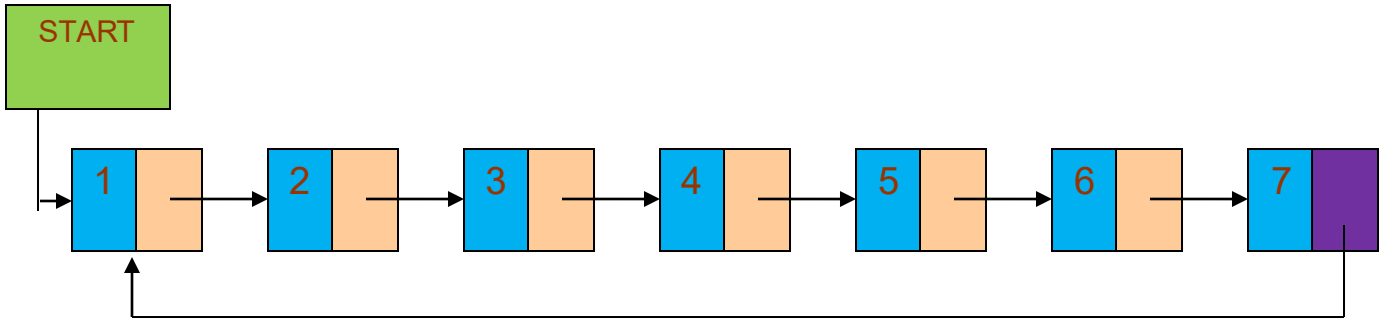
Circularly Linked List

- *In a **circular linked list**, the last node contains a pointer to the first node of the list.*
  - ***Circular singly listed list***
  - ***Circular doubly linked list.***



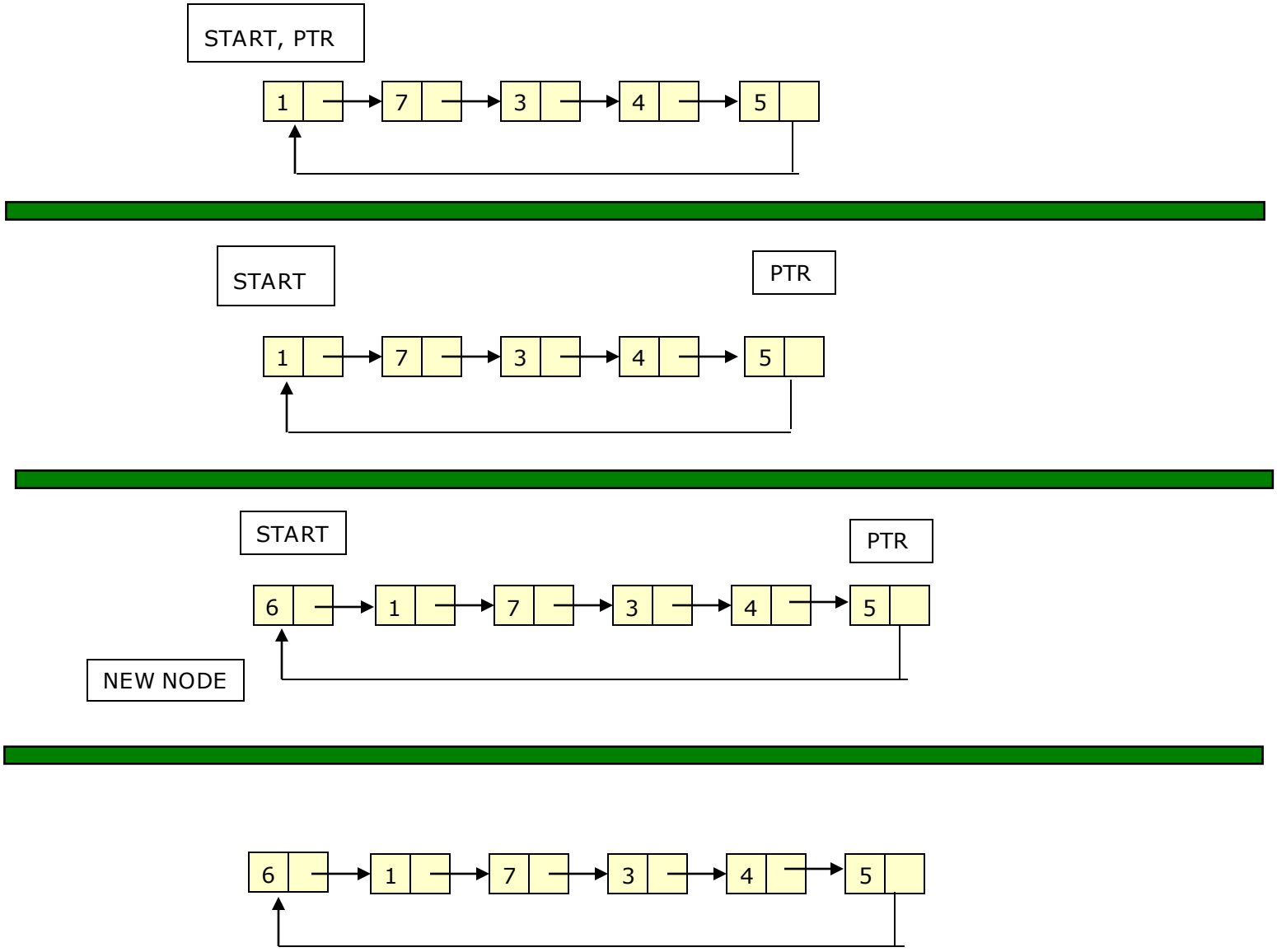
CSLL - TRAVERSING A LINKED LIST



ALGORITHM FOR TRAVERSING A LINKED LIST

```
Step 1: [INITIALIZE] SET PTR = START
Step 2: Repeat Steps 3 and 4 while PTR -> NEXT != START
Step 3:              Apply Process to PTR  ->  DATA
Step 4:              SET PTR = PTR  ->  NEXT
        [END OF LOOP]
Step 5: Apply Process to PTR  ->  DATA
Step 6: EXIT
```

CSLL - INSERTING A NODE AT THE BEGINNING

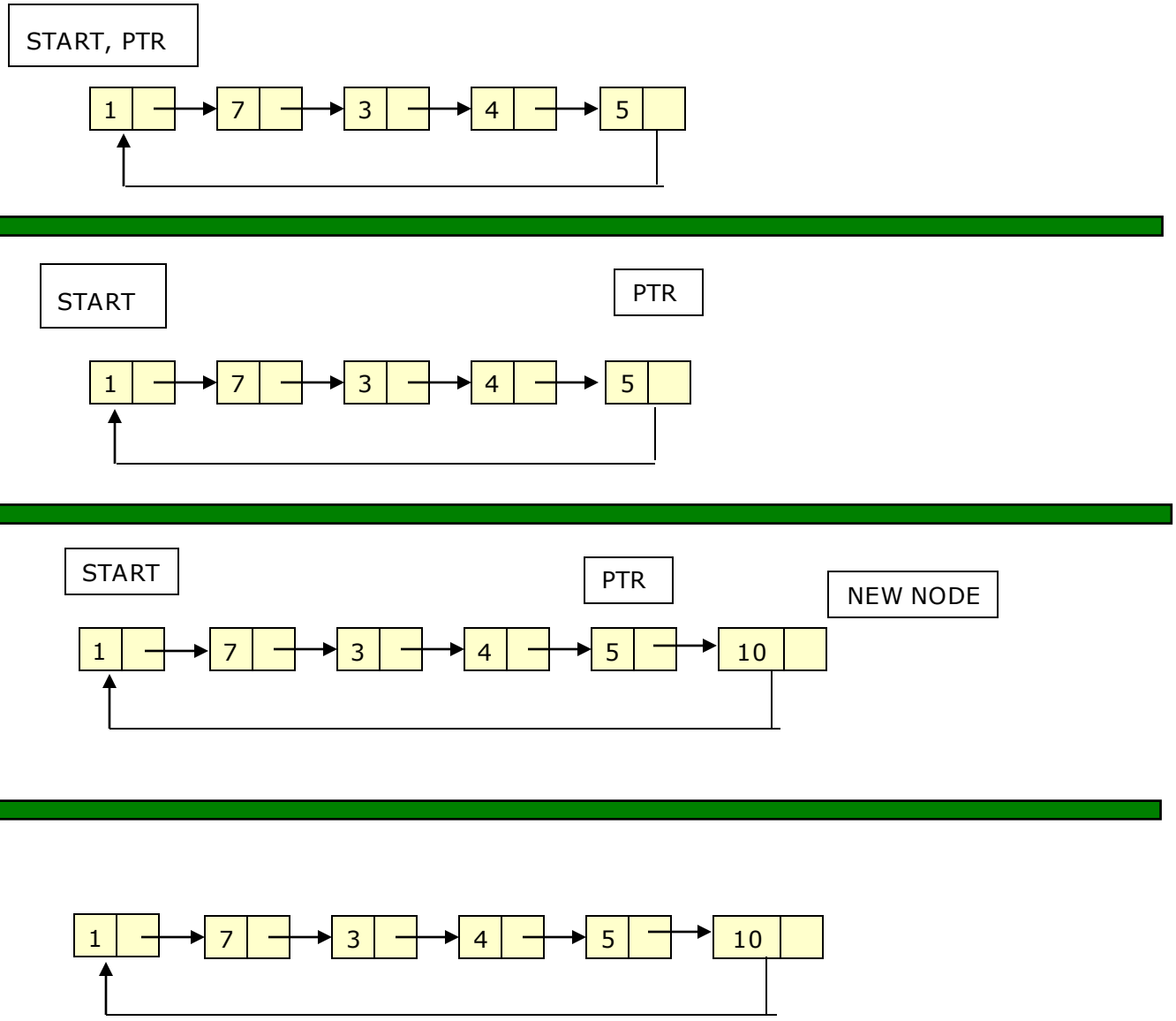


CSLL - INSERTING A NODE AT THE BEGINNING

ALGORITHM TO INSERT A NEW NODE IN THE BEGINNING OF THE CIRCULAR LINKED LIST

```
Step 1: IF AVAIL = NULL, then
                    Write OVERFLOW
                    Go to Step 7
        [END OF IF]
Step 2: SET New_Node = AVAIL
Step 3: SET New_Node -> DATA = VAL
Step 4: SET PTR = START
Step 5:   Repeat Step 6 while PTR -> NEXT != START
Step 6:             PTR = PTR -> NEXT
Step 7: SET New_Node -> Next = START
Step 8: SET PTR -> NEXT = New_Node
Step 9: SET START = New_Node
Step 10: EXIT
```

CSLL - INSERTING A NODE AT THE END

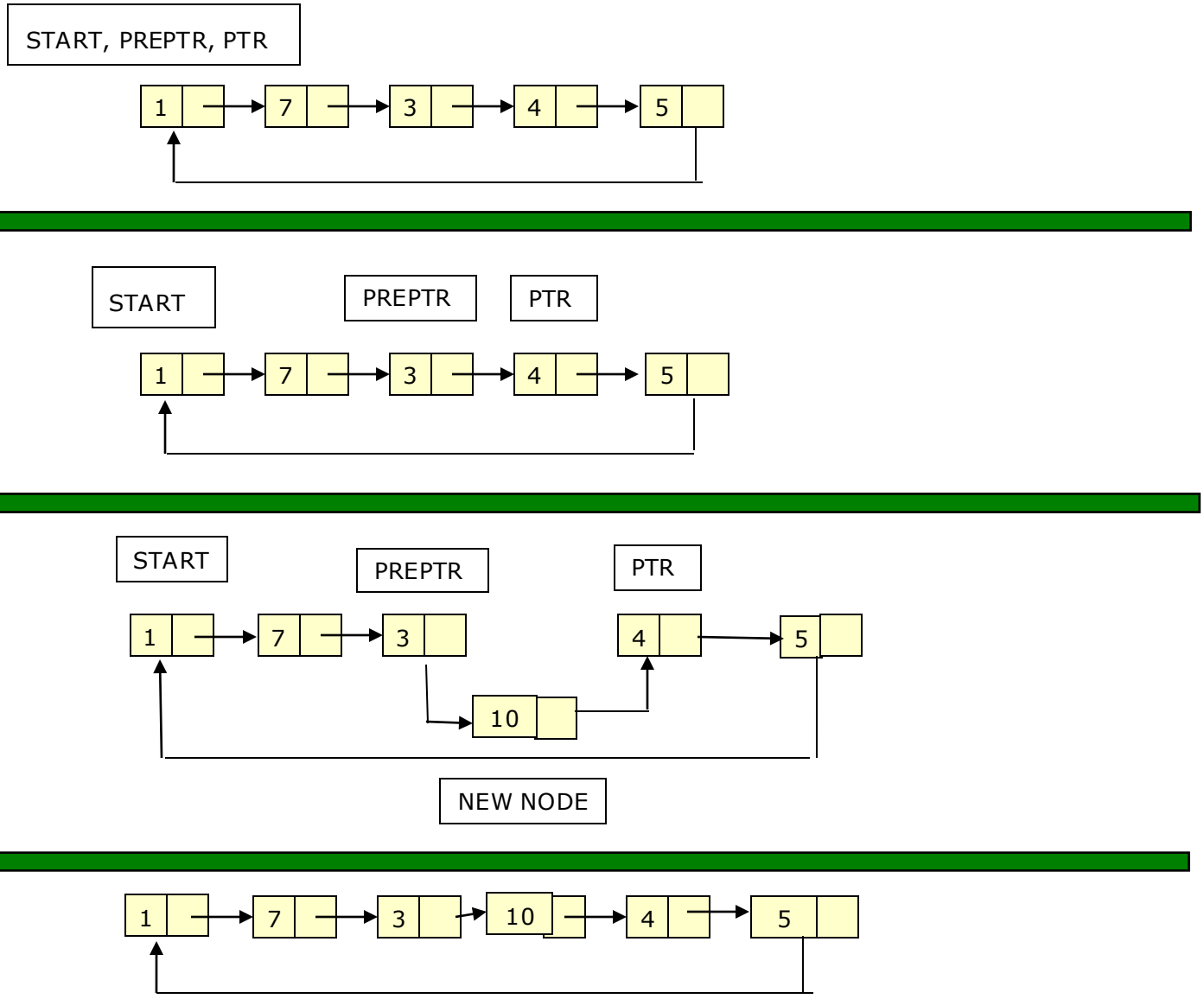


CSLL - INSERTING A NODE AT THE END

ALGORITHM TO INSERT A NEW NODE AT THE END OF THE CIRCULAR LINKED LIST

```
Step 1: IF AVAIL = NULL, then
                    Write OVERFLOW
                    Go to Step 9
          [END OF IF]
Step 2: SET New_Node = AVAIL
Step 3: SET New_Node -> DATA = VAL
Step 4: SET New_Node -> Next = START
Step 5: SET PTR = START
Step 6: Repeat Step 7 while PTR -> NEXT != START
Step 7:             SET PTR = PTR -> NEXT
        [END OF LOOP]
Step 8: SET PTR -> NEXT = New_Node
Step 9: EXIT
```

CSLL - INSERTING A NODE AFTER NODE THAT HAS VALUE NUM

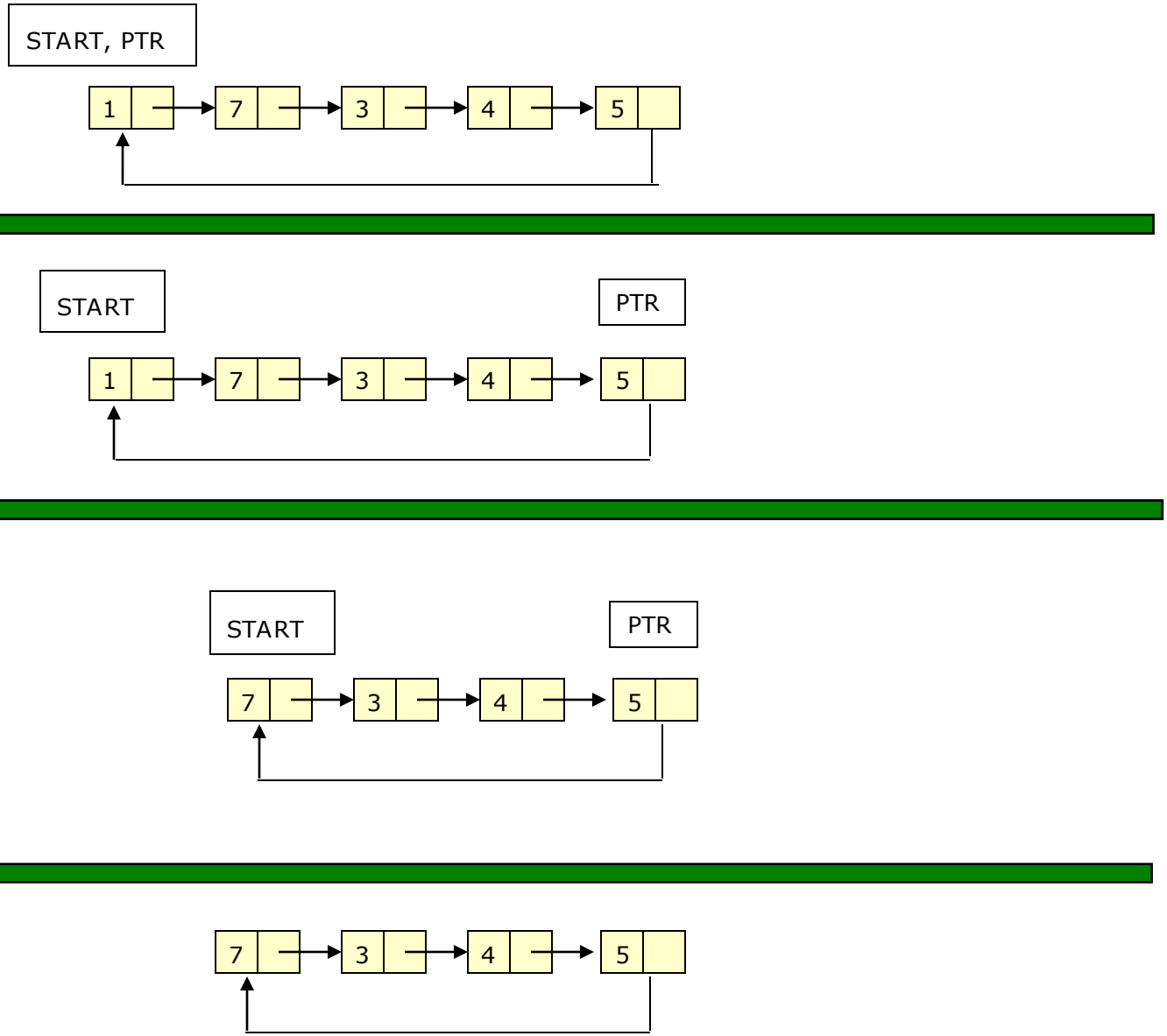


CSLL - INSERTING A NODE AFTER NODE THAT HAS VALUE NUM

ALGORITHM TO INSERT A NEW NODE AFTER A NODE THAT HAS VALUE NUM

```
Step 1: IF AVAIL = NULL, then
                        Write OVERFLOW
                        Go to Step 11
        [END OF IF]
Step 2: SET New_Node = AVAIL
Step 3: SET New_Node - > DATA = VAL
Step 4: SET PTR = START
Step 5: SET PREPTR = PTR
Step 6: Repeat Steps 7 and 8 while PREPTR -> DATA  != NUM
Step 7:                 SET PREPTR = PTR
Step 8:                 SET PTR = PTR -> NEXT
        [END OF LOOP]
Step 9: PREPTR -> NEXT = New_Node
Step 10: SET New_Node -> NEXT = PTR
Step 11: EXIT
```

CSLL - DELETING THE FIRST NODE

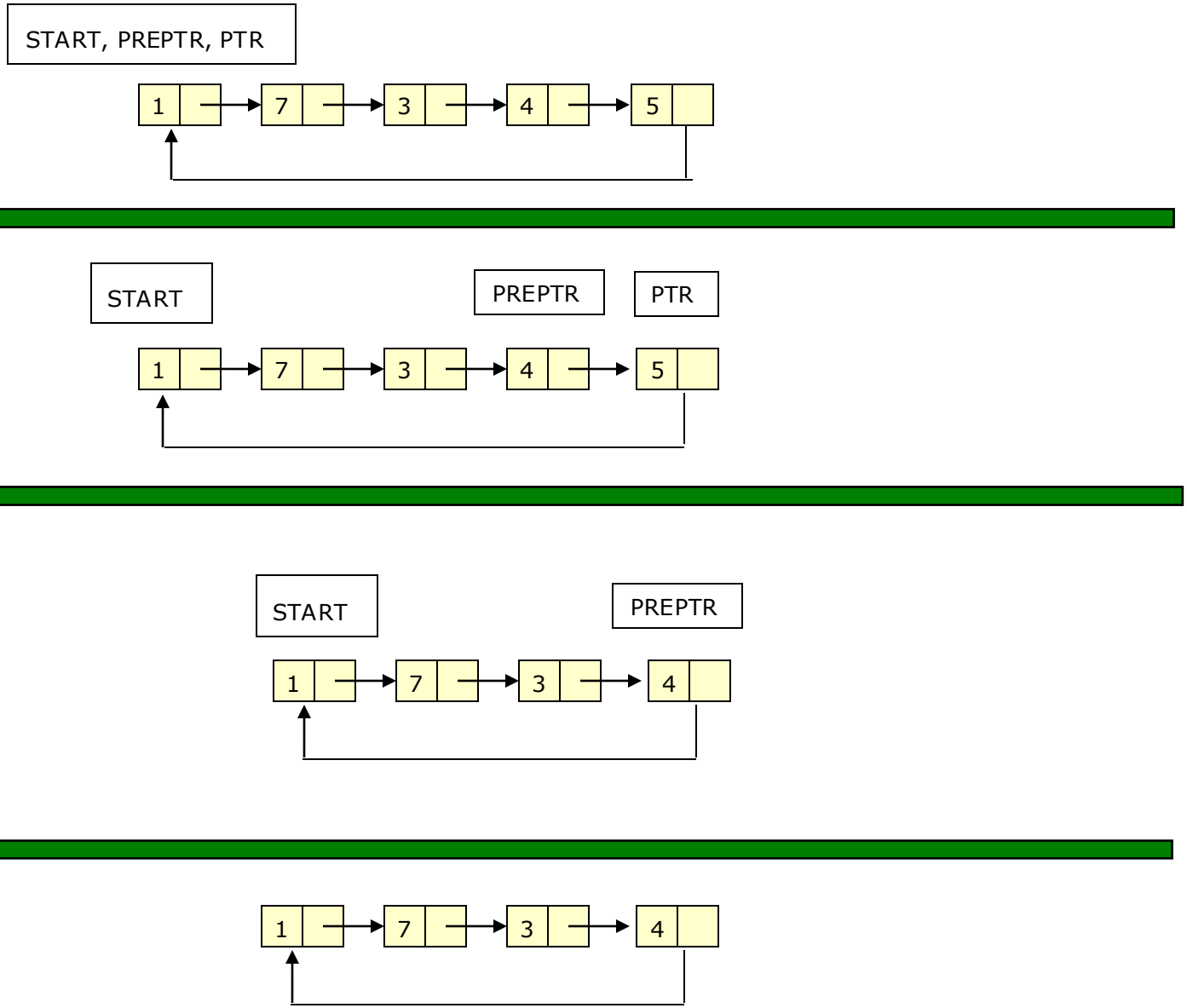


SLL - DELETING THE FIRST NODE

•ALGORITHM TO DELETE THE FIRST NODE OF THE CIRCULAR LINKED LIST

```
Step 1: IF START = NULL, then
                        Write UNDERFLOW
                        Go to Step 8
            [END OF IF]
Step 2: SET PTR = START
Step 3: Repeat Step 4 while PTR->NEXT != START
Step 4:                 SET PTR = PTR->NEXT
            [END OF IF]
Step 5: SET PTR->NEXT = START->NEXT
Step 6: FREE START
Step 7: SET START = PTR->NEXT
Step 8: EXIT
```

CSLL - DELETING THE LAST NODE

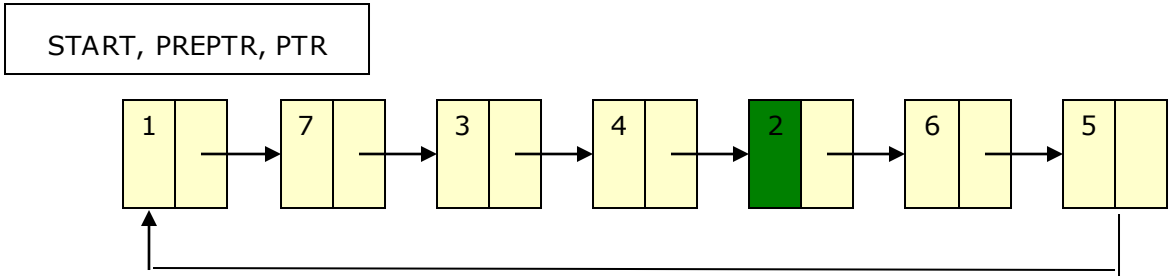


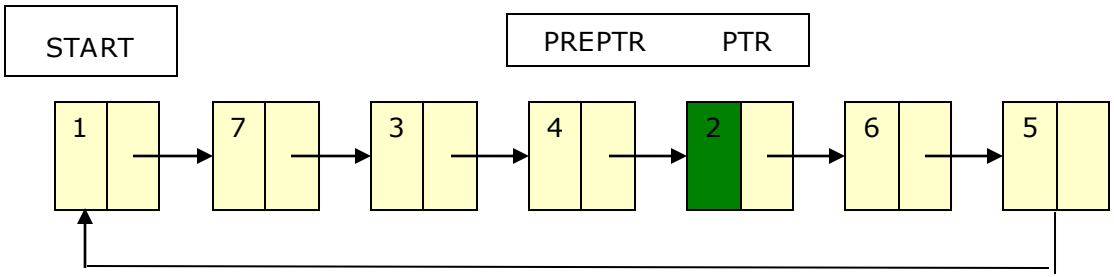
CSLL - DELETING THE LAST NODE

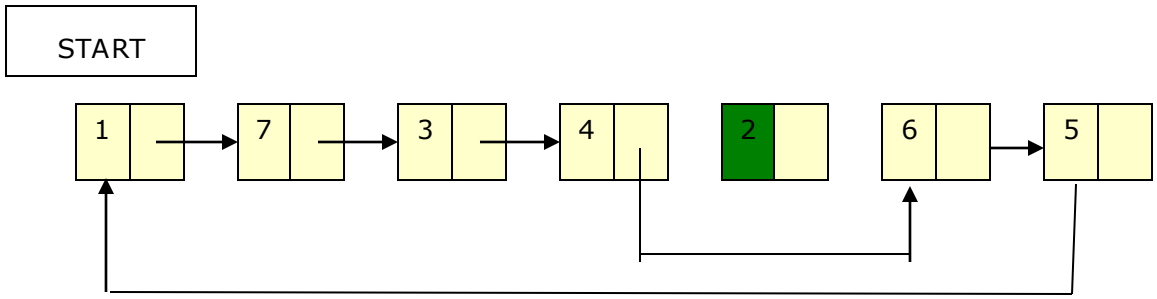
ALGORITHM TO DELETE THE LAST NODE OF THE LINKED LIST

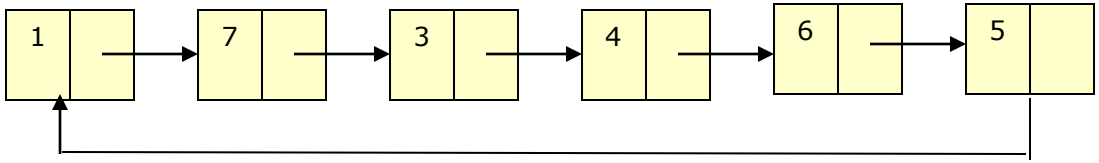
```
Step 1: IF START = NULL, then
                    Write UNDERFLOW
                    Go to Step 8
        [END OF IF]
Step 2: SET PTR = START
Step 3: Repeat Steps 4 and 5 while PTR -> NEXT != START
Step 4:             SET PREPTR = PTR
Step 5:             SET PTR = PTR -> NEXT
          [END OF LOOP]
Step 6: SET PREPTR -> NEXT = START
Step 7: FREE PTR
Step 8: EXIT
```

CSLL - DELETING THE NODE WHOSE VALUE NUM









CSLL - DELETING THE NODE WHOSE VALUE NUM

ALGORITHM TO DELETE THE NODE WHOSE VALUE NUM FROM THE CIRCULAR LINKED LIST

```
Step 1: IF START = NULL, then
                    Write UNDERFLOW
                    Go to Step 9
          [END OF IF]
Step 2: SET PTR = START
Step 3: SET PREPTR = PTR
Step 4: Repeat Step 5 and 6 while PREPTR -> DATA != NUM
Step 5:             SET PREPTR = PTR
Step 6:             SET PTR = PTR -> NEXT
          [END OF LOOP]
Step 7: SET PREPTR -> NEXT = PTR -> NEXT
Step 8: FREE PTR
Step 9: EXIT
```

SLL - INSERTING A NODE AT THE END

```
IF AVAIL = NULL, then
   Write OVERFLOW
[END OF IF]
SET New_Node = AVAIL
SET New_Node -> DATA = VAL
SET New_Node -> Next =
NULL
SET PTR = START
while PTR -> NEXT != NULL
   SET PTR = PTR -> NEXT
[END OF LOOP]
SET PTR -> NEXT = New_Node
EXIT
```

CSLL - INSERTING A NODE AT THE END

```
IF AVAIL = NULL, then
   Write OVERFLOW
[END OF IF]
SET New_Node = AVAIL
SET New_Node -> DATA = VAL
SET New_Node -> Next = START
SET PTR = START
while PTR -> NEXT != START
   SET PTR = PTR -> NEXT
[END OF LOOP]
SET PTR -> NEXT = New_Node
EXIT
```

SLL - DELETING A NODE AT THE END

```
IF START = NULL, then
   Write UNDERFLOW
[END OF IF]
SET PTR = START
while PTR -> NEXT != NULL
   SET PREPTR = PTR
   SET PTR = PTR -> NEXT
[END OF LOOP]
SET PREPTR -> NEXT = NULL
FREE PTR
EXIT
```

CSLL - DELETING A NODE AT THE END

```
IF START = NULL, then
   Write UNDERFLOW
[END OF IF]
SET PTR = START
while PTR -> NEXT != START
   SET PREPTR = PTR
   SET PTR = PTR -> NEXT
[END OF LOOP]
SET PREPTR -> NEXT = START
FREE PTR
EXIT
```