DSA Write-up
Experiment number 09

Name: Aamir Ansari **Batch:** A **Roll no.** 01

AIM: To implement of Singly Circular linked lists

THEORY:

A circular linked list is the type of linked list in which the last node contains a pointer to the first node of the list. A circular linked list has no beginning and no ending.

ALGORITHM

INSERT

At the beginning

```
Step 1: [INITIALIZE] newNode
Step 2: SET newNode->data = data
Step 3: IF end == NULL
                SET end = newNode
                SET newNode->next = end
                Goto Step 5
Step 4: ELSE
                newNode->next = end->next
                end->next = newNode
        [END IF]
Step 5: EXIT
```

At the end

```
Step 1: [INITIALIZE] newNode, ptr
Step 2: SET newNode->data = data
Step 3: IF end == NULL
                SET end = newNode
                SET newNode->next = end
                Goto Step 5
Step 4: ELSE
                SET ptr = end->next
                Repeat while ptr->next != end
                        ptr=ptr-> next
                [END LOOP]
                newNode->next = end->next
                end->next = newNode
                end = newNode
        [END IF]
Step 5: EXIT
```

At a position:

```
Step 1: [INITIALIZE] newNode, ptr, prePtr
Step 2: SET newNode = end->next->next , prenewNode = newNode
Step 3: SET new->data = data
Step 4: IF end == NULL
```

PRINT “LIST EMPTY”
Goto Step 12
[END IF]
Step 4: SET count = 1
Step 5:Repaet step 6 to 8 while count!=position AND ptr->next!=end->next
Step 6: SET prePtr = ptr;
Step 7: SET ptr = ptr->next;
Step 8: count = count + 1
Step 9: IF count == 1
newNode->next = ptr
end->next = newNode
Step 10: ELSE IF ptr->next == end->next AND count < pos
newNode->next = end->next
end->next = newNode
end = newNode
Step 11:ELSE
prePtr->next = newNode;
newNode->next = ptr;
[END IF]
Step 12:EXIT

Before a given value:

Step 1: [INITIALIZE] newNode, ptr, prePtr
Step 2: SET new->data= data
Step 3: SET newNode=end->next
Step 4: SET prePtr = ptr
Step 5: IF end == NULL
PRINT “LIST IS EMPTY”
Goto Step 11
Step 6: Repeat step 7&8 while newNode->data != val
Step 7: SET prePtr = ptr;
Step 8: SET ptr = ptr->next;
Step 9:IF ptr == end->next
SET newNode->next = end->next;
SET end->next = newNode;
Step 10: ELSE
SET prePtr->next = newNode
SET newNode->next = ptr
Step 11:EXIT

After a given Value:

Step 1: [INITIALIZE] ptr, prePtr, newNode
Step 2: IF end == NULL
PRINT “LIST IS EMPTY”
Goto Step 9
Step 3: SET ptr = end->next
Step 4: DO steps 5&6 while ptr->data != val
Step 5: SET prePtr = ptr;
Step 6: SET ptr = ptr->next;
[END LOOP]
Step 7: IF prePtr->next == end->next
newNode->next = end->next;

```
                prePtr->next = newNode;
                end = newNode;
Step 8: ELSE
                prePtr->next = newNode;
                newNode->next = ptr;
        [END IF]
Step 9:EXIT
```

DELETE

Value at the beginning

```
Step 1: [INITIALIZE] ptr
Step 2: IF end == NULL
                PRINT "LIST IS EMPTY"
                Goto Step 6
        [END IF]
Step 3: SET end->next == ptr->next
Step 4: IF ptr == end
                end = NULL
        [END IF]
Step 5: free(ptr)
Step 6: EXIT
```

At the end

```
Step 1: [INITIALIZE] ptr, prePtr
Step 2: IF end == NULL
                PRINT "LIST IS EMPTY"
                Goto Step 10
        [END IF]
Step 3: Repeat Steps 4, 5 while ptr->next != end->next
Step 4:         SET prePtr = ptr;
Step 5:         SET ptr = ptr->next;
        [END LOOP]
Step 6: SET prePtr->next = end->next;
Step 7: SET end = prePtr;
Step 8: IF prePtr == ptr
                SET end = NULL
        [END IF]
Step 9: free(ptr)
Step 10: EXIT
```

Value at a Position

```
Step 1: [INITIALIZE] ptr, prePtr
Step 2: IF end == NULL
                PRINT "LIST IS EMPTY"
                Goto Step 13
        [END IF]
Step 3: SET count = 1
Step 4: Repeat steps  while count != pos AND ptr->next != end->next
Step 5:         SET prePtr = ptr;
Step 6:         SET ptr = ptr->next;
```

```
Step 7:         SET count = count + 1;
        [END LOOP]
Step 8: IF POS > count
                PRINT “NO NODE AVAILABLE”
                Goto Step 13
        [END IF]
Step 9: IF end->next == ptr
                SET end->next = ptr->next;
                free(ptr);
Step 10: ELSE IF ptr->next == end->next
                SET prePtr->next = end->next;
                SET end = prePtr;
                SET end->next = prePtr->next;
                free(ptr);
Step 11: ELSE
                SET prePtr->next = ptr->next;
                free(ptr);
        [END IF]
Step 12: IF ptr->next == end->next
                SET end =NULL
        [END IF]
Step 13: EXIT
```

Before a given value

```
Step 1: [INITIALIZE] ptr, prePtr
Step 2: IF end == NULL
                PRINT "LIST IS EMPTY"
                Goto Step 9
        [END IF]
Step 3: IF ptr->data == val
                PRINT “NO NODE BEFORE THIS”
                Goto Step 9
        [END IF]
Step 4: Repeat Steps 5, 6 while ptr->next->data != val
Step 5:         SET prePtr = ptr;
Step 6:         SET ptr = ptr->next;
        [END LOOP]
Step 7: prePtr-next = ptr->next
Step 8: free(ptr)
Step 9: EXIT
```

After a given value

```
Step 1: [INITIALIZE] ptr, prePtr
Step 2: IF end == NULL
                PRINT "LIST IS EMPTY"
                Goto Step 10
        [END IF]
Step 3: Repeat Steps 5, 6 while ptr->data != val
Step 4:         SET prePtr = ptr;
Step 5:         SET ptr = ptr->next;
        [END LOOP]
Step 6: prePtr = ptr
```

Step 7: ptr = ptr->next
Step 8: IF ptr->next == end->next
SET prePtr->next = end->next;
SET end = prePtr;
free(ptr);
Step 9: ELSE
SET prePtr->next = ptr->next;
free(ptr);
[END IF]
Step 10: EXIT

3.UPDATE

Value at the beginning

Step 1: IF end == NULL
PRINT "LIST IS EMPTY"
Goto Step 3
[END IF]
Step 2: SET end->next->data = toUpdate
Step 3:EXIT

At the end

Step 1: IF end == NULL
PRINT "LIST IS EMPTY"
Goto Step 3
[END IF]
Step 2: SET end->data = toUpdate
Step 3:EXIT

Value at a given Position

Step 1: IF end == NULL
PRINT "LIST IS EMPTY"
Goto Step 9
[END IF]
Step 2: SET count = 1
Step 3: [INITIALIZE] ptr
Step 4: Repeat step 5, 6 while count != pos AND ptr->next!=end->next
Step 5: SET ptr = ptr->next
Step 6: SET count = count + 1
[END LOOP]
Step 7: IF pos > count
PRINT “NO NODE AT GIVEN POSITION”
Goto Step 9
[END IF]
Step 8: SET ptr->data = toUpdate
Step 9: EXIT

Before a particular value

Step 1: IF end == NULL
PRINT "LIST IS EMPTY"
Goto Step 6

[END IF]
Step 2: [INITIALIZE] ptr
Step 3:Repeat step 4&5 while ptr->next->data != val
Step 4: SET ptr = ptr->next
[END LOOP]
Step 5: SET ptr->data = toUpdate;
Step 6: EXIT

After a particular value

Step 1: IF end == NULL
PRINT "LIST IS EMPTY"
Goto Step 6
[END IF]
Step 2: [INITIALIZE] ptr
Step 3:Repeat step 4&5 while ptr->next->data != val
Step 4: SET ptr = ptr->next
[END LOOP]
Step 5: SET ptr->next->data = toUpdate;
Step 6: EXIT

4. SEARCH

Step 1: IF end == NULL
PRINT “LIST IS EMPTY”
Goto Step 9
[END IF]
Step 2: [INITIALIZE] ptr
Step 3: SET Count = 1
Step 4: Repeat step 4&5 while ptr->data != val && count<=countNodes()+1
Step 5: SET ptr = ptr->next
Step 6: SET count = count + 1
[END LOOP]
Step 7: IF count > countNodes()
PRINT “NOT FOUND”
Step 8: ELSE
PRINT “FOUND”
Step 9: EXIT

5. COUNT NODES

Step 1: IF end == NULL
return 0
[END IF]
Step 2: [INITIALIZE] ptr
Step 3: SET Count = 1
Step 4:Repeat step 5, 6 while ptr->next->data != val
Step 5: SET ptr = ptr->next
Step 6: SET count = count + 1
[END LOOP]
Step 7: return count

6. DISPLAY

Step 1: IF end == NULL
PRINT “LIST IS EMPTY”
Goto Step 7
[END IF]
Step 2: [INITIALIZE] ptr
Step 3: Repeat steps 4, 5 while ptr->next != end->next
Step 4: PRINT ptr->data
Step 5: ptr = ptr->next;
[END LOOP]
Step 6: PRINT ptr->data
Step 7: EXIT

****