Experiment no 5

Doubly Linked List

OBJECTIVES:

- To learn about the concept of Doubly linked list.
- To learn different types of linked lists i.e Singly and Doubly linked list.
- To implement Doubly linked lists using class and functions.

1. LINKED LIST:

A singly linked list is a linear data structure in which the elements are not stored in contiguous memory locations and each element is connected only to its next element using a pointer.

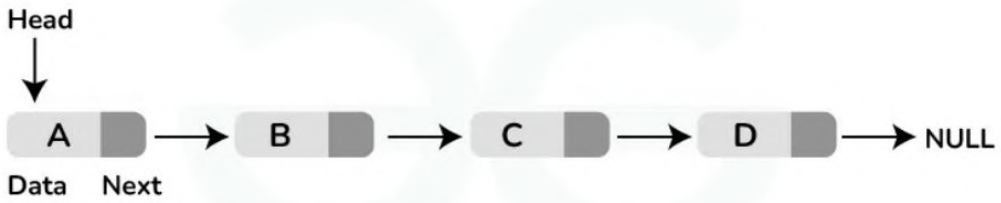

Figure 1: Linked List Block Diagram

2. DOUBLY LINKED LIST:

Inserting a new node in a doubly linked list is very similar to inserting new node in linked list. There is a little extra work required to maintain the link of the previous node. A node can be inserted in a Doubly Linked List in four ways:

- At the front of the DLL.
- In between two nodes
  After a given node.
  Before a given node.
- At the end of the DLL.

2.1 Insertion at the beginning of Doubly linked list:

To insert a new node at the beginning of the doubly list, we can use the following steps:

- Allocate memory for a new node (say new_node) and assign the provided value to its data field.
- Set the previous pointer of the new_node to nullptr.
- If the list is empty:
  - Set the next pointer of the new_node to nullptr.
  - Update the head pointer to point to the new_node.
- If the list is not empty:
  - Set the next pointer of the new_node to the current head.
  - Update the previous pointer of the current head to point to the new_node.
  - Update the head pointer to point to the new_node.

```
# Adding a node at the front of the list
def push(self, new_data):

    # 1 & 2: Allocate the Node & Put in the data
    new_node = Node(data=new_data)

    # 3. Make next of new node as head and previous as NULL
    new_node.next = self.head
    new_node.prev = None

    # 4. change prev of head node to new node
    if self.head is not None:
        self.head.prev = new_node

    # 5. move the head to point to the new node
    self.head = new_node

# This code is contributed by jatinreaper
```

Figure 2: Doubly Linked List with 5 nodes

3. INSERTION IN BETWEEN NODES:

To insert a new node at the beginning of the doubly list, we can use the following steps:

- Allocate memory for a new node (say new_node) and assign the provided value to its data field.
- Set the previous pointer of the new_node to nullptr.
- If the list is empty:
  - Set the next pointer of the new_node to nullptr.
  - Update the head pointer to point to the new_node.
- If the list is not empty:
  - Set the next pointer of the new_node to the current head.
  - Update the previous pointer of the current head to point to the new_node.
  - Update the head pointer to point to the new_node.

Below is the implementation of the 5 steps to insert a node at the front of the linked list:

```
# Adding a node at the front of the list
def push(self, new_data):

    # 1 & 2: Allocate the Node & Put in the data
    new_node = Node(data=new_data)

    # 3. Make next of new node as head and previous as NULL
    new_node.next = self.head
    new_node.prev = None

    # 4. change prev of head node to new node
    if self.head is not None:
        self.head.prev = new_node

    # 5. move the head to point to the new node
    self.head = new_node

# This code is contributed by jatinreaper
```

Figure 3: Node in between nodes

4. INSERTION AT THE END OF THE DOUBLY LINKED LIST:

The new node is always added after the last node of the given Linked List. This can be done using the following steps:

- Create a new node (say new_node).
- Put the value in the new node.
- Make the next pointer of new_node as null.
- If the list is empty, make new_node as the head.
- Otherwise, travel to the end of the linked list.
- Now make the next pointer of last node point to new_node.
- Change the previous pointer of new_node to the last node of the list.

Below is the implementation of the steps to insert a node at the end of the linked list:

```
# Add a node at the end of the DLL
def append(self, new_data):

    # 1. allocate node
    # 2. put in the data
    new_node = Node(data=new_data)
    last = self.head

    # 3. This new node is going to be the
    # last node, so make next of it as NULL
    new_node.next = None

    # 4. If the Linked List is empty, then
    #  make the new node as head
    if self.head is None:
        new_node.prev = None
        self.head = new_node
        return

    # 5. Else traverse till the last node
    while (last.next is not None):
        last = last.next

    # 6. Change the next of last node
    last.next = new_node
    # 7. Make last node as previous of new node */
    new_node.prev = last
```

Figure 4: Node in between nodes

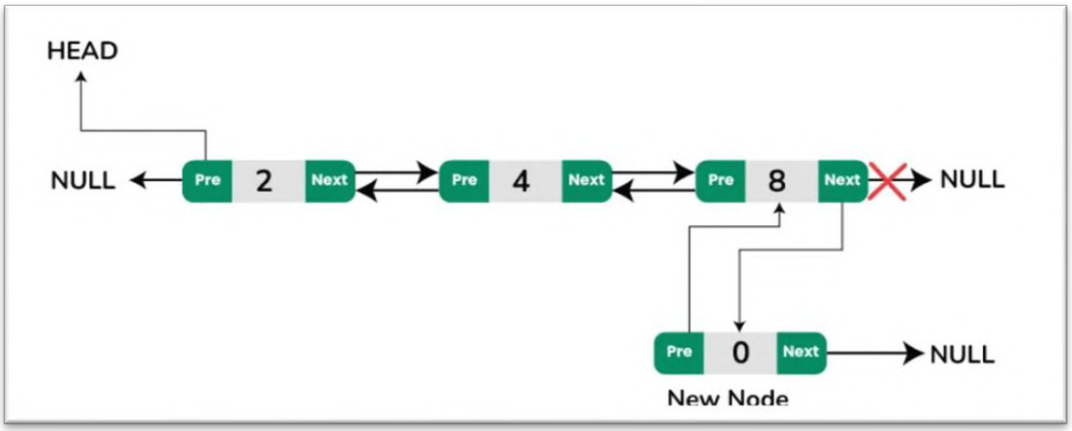

Figure 5: Block Diagram for Doubly Linked List

5. ACTIVITIES:

A1) Write a code for doubly linked lists and also Add a node after a given node in a Doubly Linked List.

A2) Write a code for doubly linked list and also Add a node before a given node in a Doubly Linked List.

A3) Write a code for doubly linked list for 7 nodes and also loop through the linked list to show its next node address and previous node address.

A4) Write a python program to get the n number of nodes from the end of a doubly linked list.

IMPORTANT: All the activities` code should be attached to the manual before summary section.

Lab Exercise and Summary

Summary should cover Introduction, Procedure, Data Analysis and Evaluation.

Student's Signature:_____ **Date:**_____

LABORATORY SKILLS ASSESSMENT (Psychomotor)

Total Marks:100

Criteria (Max Marks)	Level 1 0% ≤ S < 50%	Level 2 50% ≤ S< 70%	Level 3 70% ≤ S< 90%	Level 4 90%≤ S ≤100%	Score (S)
Procedural Awareness (30)	Selects inappropriate skills and/or strategies Required by the task.	Selects and applies appropriate skills and/or strategies required by the task with major errors.	Selects and applies the appropriate strategies and/or skills specific to the task without significant errors.	Selects and applies appropriate strategies and/or skills specific to the task without any error.	
Practical Implementation (30)	Makes major critical errors in applying procedural knowledge related to python doubly linked list.	Makes numerous critical errors in applying procedural knowledge related to python doubly linked list.	Makes some non-critical errors in applying procedural knowledge related to python doubly linked list.	Applies the procedural knowledge in optimized ways related to python Lists, doubly linked list.	
Use of Tool/Equipment (30)	Uses tools, equipment and materials with limited competence.	Uses tools, equipment and materials with some competence.	Uses tools, equipment and materials with considerable competence.	Uses tools, equipment and materials with a high degree of competence.	
Safety (10)	Requires constant reminders to follow safety procedures.	Requires some reminders to follow safety procedures.	Follows safety procedures with only minimal reminders.	Routinely follows safety procedures.	
				Marks Obtained	

Instructor Name:_____ **Sign:** _____

LABORATORY SKILLS ASSESSMENT (Affective)

Total Marks: 40

Criteria (Max. Marks)	Level 1 0% ≤ S < 50%	Level 2 50% ≤ S < 70%	Level 3 70% ≤ S < 90%	Level 4 90% ≤ S ≤ 100%	Score
Introduction (5)	Very little background information provided or information is incorrect	Introduction is brief with some minor mistakes	Introduction is nearly complete, missing some minor points	Introduction complete and well-written; provides all necessary background principles for the experiment	
Procedure (5)	Many stages of the procedure are not entered on the lab report.	Many stages of the procedure are entered on the lab report.	The procedure could be more efficiently designed but most stages of the procedure are entered on the lab report.	The procedure is well designed and all stages of the procedure are entered on the lab report.	
Data Record (10)	Data is brief and missing significant pieces of information.	Data provides some significant information and has few critical mistakes.	Data is almost complete but has some minor mistakes.	Data is complete and relevant. Tables with units are provided. Graphs are labeled. All questions are answered correctly.	
Data Analysis (10)	Data is presented in very unclear manner.	Data is presented in ways that are not clear enough.	Data is presented in ways that can be understood and interpreted.	Data is presented in ways that best facilitate understanding and interpretation.	
Report Quality (10)	Report contains many errors.	Report is somewhat organized with some spelling or grammatical errors.	Report is well organized and cohesive but contains some grammatical errors.	Report is well organized and cohesive and contains no grammatical errors. Presentation seems polished.	
				Marks Obtained	

LABORATORY SKILLS ASSESSMENT (Cognitive)

Total Marks: 10

(If any) **Marks Obtained**	

Instructor's Signature:_____ **Date:**_____