LAB-06

Name: K V Jaya Harsha

Roll no: CS23B1034 Date: 04-08-2024

Q1. Circular Linked List. (in cpp)

```cpp
// CS23B1034
// K V Jaya Harsha
#include <iostream>
using namespace std;

struct Node{
    int data;
    Node *next;
};

Node *create_l_l(int n){
    Node *head = nullptr;
    Node *temp = nullptr;
    Node *last = nullptr;
    int value;

    for (int i = 0; i < n; i++)
    {
        cout << "Element " << (i + 1) << " : ";
        cin >> value;
        Node *newNode = new Node;
        newNode->data = value;
        newNode->next = nullptr;

        if (head == nullptr){
            head = newNode;
            last = head;
        }
        else{
            last->next = newNode;
            last = newNode;
        }
    }

    if (last != nullptr){
        last->next = head;
    }
    return head;
}
```

```cpp
void display_l_l(Node *head){
    if (head == nullptr)
        return;

    Node *temp = head;
    do{
        cout << temp->data << " -> ";
        temp = temp->next;
    } while (temp != head);

    cout << "(head)" << endl;
}

int main(){
    cout << "Number of nodes: ";
    int n;
    cin >> n;
    Node *head = create_l_l(n);
    cout << "Circular Linked List: ";
    display_l_l(head);
    return 0;
}
```

```
Number of nodes: 5
Element 1 : 9
Element 2 : 2
Element 3 : 6
Element 4 : 34
Element 5 : 7
Circular Linked List: 9 -> 2 -> 6 -> 34 -> 7 -> (head)
```

Q2. Interchange kth and k+1th node of circular linked list. (in cpp)

```cpp
// CS23B1034
// K V Jaya Harsha
#include <iostream>
using namespace std;
struct Node{
    int data;
    Node *next;
};
Node *create_l_l(int n){
    Node *head = nullptr;
    Node *temp = nullptr;
    Node *last = nullptr;
    int value;
    for (int i = 0; i < n; i++){
        cout << "Element " << (i + 1) << " : ";
        cin >> value;
        Node *newNode = new Node;
        newNode->data = value;
        newNode->next = nullptr;

        if (head == nullptr){
            head = newNode;
            last = head;
        }
        else{
            last->next = newNode;
            last = newNode;
        }
    }

    if (last != nullptr){
        last->next = head;
    }
    return head;
}
void display_l_l(Node *head){
    if (head == nullptr)
        return;
    Node *temp = head;
    do{
        cout << temp->data << " -> ";
        temp = temp->next;
    } while (temp != head);
    cout << "(HEAD)" << endl;
}
```

```cpp
void interchange(Node *&head, int value){
    cout << "You are interchanging " << value << " and " << value + 1 << " . " << endl;
    if (head==NULL||head->next==head||value<1){
        return;
    }
    Node *a = NULL;
    Node *b = head;
    for (int i = 1; i < value; ++i){
        a = b;
        b = b->next;
        if (b == head)
            return;
    }
    Node *bb = b->next;
    if (bb == head)
        return;
    if (a != NULL){
        a->next = bb;
    }
    else{
        head = bb;
    }
    b->next = bb->next;
    bb->next = b;
    if (b->next == head){
        Node *temp = head;
        while (temp->next != b){
            temp = temp->next;
        }
        temp->next = b;
    }
}
```

```
linked-list } ; if ($?) { .\k-interchange-circular-linked-list }
Number of nodes: 4
Element 1 : 1
Element 2 : 3
Element 3 : 4
Element 4 : 6
Circular Linked List: 1 -> 3 -> 4 -> 6 -> (HEAD)
Enter the no of node to interchange 2
You are interchanging 2 and 3.
The interchanged Circular Linked List: 1 -> 4 -> 3 -> 6 -> (HEAD)
PS C:\Users\harsh\OneDrive\Documents\Desktop\challenge\dsa\labs\lab-6>
```

Q3. Doubly linked list with function to count no of non zero nodes. (in cpp)

```cpp
// CS23B1034
// K V Jaya Harsha
#include <iostream>
using namespace std;
struct Node{
    int data;
    Node *next;
    Node *prev;
};

Node *create_dll(int n){
    Node *head = nullptr;
    Node *temp = nullptr;
    Node *newNode = nullptr;
    int value;

    for (int i = 0; i < n; i++){
        cout << "Element " << (i + 1) << " : ";
        cin >> value;

        newNode = new Node;
        newNode->data = value;
        newNode->next = nullptr;
        newNode->prev = nullptr;

        if (head == nullptr){
            head = newNode;
        }
        else{
            temp = head;
            while (temp->next != nullptr){
                temp = temp->next;
            }
            temp->next = newNode;
            newNode->prev = temp;
        }
    }
    return head;
}

void display_forward(Node *head){
    Node *temp = head;
    while (temp != nullptr){
        cout << temp->data << " -> ";
        temp = temp->next;
    }
    cout << "NULL" << endl;
}

void display_backward(Node *tail){
    Node *temp = tail;
    while (temp != nullptr){
        cout << temp->data << " -> ";
        temp = temp->prev;
    }
    cout << "NULL" << endl;
}
```

```cpp
int count_non_zero(Node *head){
    int count = 0;
    Node *temp = head;
    while (temp != nullptr){
        if (temp->data != 0){
            count++;
        }
        temp = temp->next;
    }
    return count;
}

int main(){
    cout << "Number of nodes: ";
    int n;
    cin >> n;
    Node *head = create_dll(n);
    cout << "DLL_Forword: ";
    display_forward(head);
    Node *tail = head;
    while (tail != nullptr && tail->next != nullptr){
        tail = tail->next;
    }
    cout << "DLL_Backside: ";
    display_backward(tail);
    int non_zero_count = count_non_zero(head);
    cout << "non-zero nodes: " << non_zero_count << endl;

    return 0;
}
```

```
Number of nodes: 4
Element 1 : 1
Element 2 : 2
Element 3 : 3
Element 4 : 4
DLL_Forword: 1 -> 2 -> 3 -> 4 -> NULL
DLL_Backside: 4 -> 3 -> 2 -> 1 -> NULL
non-zero nodes: 4
```