\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Data Structure Lab

CEN-391

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Program 9

Code :-

#include <iostream>

using namespace std;

struct Node

{

    int data;

    Node \*next;

};

void

isEmpty(int size)

{

    cout << "isEmpty...\n";

    if (size == 0)

        cout << "Empty" << endl;

    else

        cout << "Not Empty" << endl;

}

void Display(Node \*head, int size)

{

    cout << "Display...\n";

    if (size == 0)

    {

        cout << "Queue Is Empty" << endl;

        return;

    }

    while (head != nullptr)

    {

        cout << head->data << " ";

        head = head->next;

    }

    cout << endl;

}

void Enqueue(Node \*&head, Node \*&tail, int &size)

{

    cout << "Enqueue...\n";

    size++;

    Node \*newnode = (Node \*)malloc(1 \* sizeof(Node));

    if (newnode == nullptr)

    {

        cout << "Memory Not Assigned" << endl;

        return;

    }

    cout << "Enter The Element : ";

    int val;

    cin >> val;

    newnode->data = val;

    newnode->next=nullptr;

    if (head != nullptr)

    {

        tail->next = newnode;

        tail = tail->next;

    }

    else

    {

        head = newnode;

        tail = newnode;

    }

    Display(head, size);

}

void Dequeue(Node \*&head, int &size)

{

    cout << "Dequeue...\n";

    if (size == 0)

    {

        cout << "Queue Underflow" << endl;

        return;

    }

    cout << head->data << endl;

    size--;

    Node \*todelete = head;

    head = head->next;

    delete todelete;

    Display(head, size);

}

void Front\_Rear(Node \*head, Node \*tail, int size)

{

    cout << "Front And Rear...\n";

    if (size == 0)

    {

        cout << "Queue Is Empty" << endl;

        return;

    }

    cout << "Front : " << head->data << endl;

    cout << "Rear : " << tail->data << endl;

}

void Total\_Element(int size)

{

    cout << "Total Elements In Queue : " << size << endl;

}

void Bars()

{

    cout << "---------------------------------------------------------------\n";

}

bool Options(Node \*&head, Node \*&tail, int &size)

{

    int opt;

    cin >> opt;

    Bars();

    switch (opt)

    {

    case 1:

        Enqueue(head, tail, size);

        break;

    case 2:

        Dequeue(head, size);

        break;

    case 3:

        Front\_Rear(head, tail, size);

        break;

    case 4:

        isEmpty(size);

        break;

    case 5:

        Total\_Element(size);

        break;

    case 6:

        Display(head, size);

        break;

    case 7:

        cout << "Exit...\n";

        return 0;

    default:

        cout << "Invalid Input!\nTry Again!\n";

    }

    Bars();

    return 1;

}

void Menu()

{

    cout << "\_\_\_\_\_Operations\_On\_Queue\_\_\_\_\_ \n";

    cout << "1.Enqueue \n";

    cout << "2.Dequeue \n";

    cout << "3.Front And Rear Element \n";

    cout << "4.isEmpty \n";

    cout << "5.Total Elements \n";

    cout << "6:Display \n";

    cout << "7.Exit \n";

    cout << "Enter Your Choice : ";

}

int main()

{

    system("cls");

    cout << "\_\_\_\_\_Vicky\_Gupta\_20BCS070\_\_\_\_\_\n\n";

    int size = 0;

    Node \*head = nullptr, \*tail = nullptr;

    while (true)

    {

        Menu();

        if (!Options(head,tail,size))

            break;

    }

    cout << "Exiting...\n";

    Bars();

    return 0;

}

Output :-

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated