**Logo, company name

Description automatically generated**

**Department of (Computer Science)**

**Pak-Austria Fachhochschule: Institute of Applied Sciences and Technology, Haripur, Pakistan**

**COMP-112L Data Structure** **& Algorithm Lab**

**Lab Journal**

**Class: BS Computer Science**

**Name: Ahmed Raza**

**Registration No.: B20F0436CS031**

**Semester: 4th**

**Submission Date: 26th April 2022**

**Submitted to: Engr. Rafi-Ullah**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Instructor Signature**

**Lab No. 07**

**Circular Linked List**

**Objectives:**

In this lab we will be discussing about Circular Linked List in detail. This is one of the most important concepts in C++ language.

Lab Objective is:

* To understand Circular link lists.
* To Implement link lists in C++

**Tools/Software Required:**

* All the tasks are implemented on DEV C++.

**Introduction:**

**Circular Link List**

Circular Linked List is a variation of Linked list in which the first element points to the last element and the last element points to the first element. Both Singly Linked List and Doubly Linked List can be made into a circular linked list.

**Lab Tasks:**

**Task 1:**

Write a program to insert/append, prepend, insert Node After, Delete node by Key, Update node by key operation in Circular link List.

**Code:**

**#include<iostream>**

**using namespace std;**

**class Node**

**{**

**public:**

**int data;**

**int key;**

**Node\* next;**

**//initialize**

**Node()**

**{**

**key=0;**

**data=0;**

**next=NULL;**

**}**

**Node(int key, int data)**

**{**

**this->key=key;**

**this->data=data;**

**}**

**};**

**class CircularLinkedList**

**{**

**public:**

**Node\* head;**

**CircularLinkedList()**

**{**

**head=NULL;**

**}**

**SinglyLinkedList(Node\* n)**

**{**

**head=n;**

**}**

**//Check Node exist by using key**

**Node\* nodeExist(int k)**

**{**

**Node\* temp =NULL;**

**Node\* ptr =head;**

**if(ptr==NULL)**

**{**

**return temp;**

**}**

**else**

**{**

**do**

**{**

**if(ptr->key==k)**

**{**

**temp=ptr;**

**}**

**ptr=ptr->next;**

**}**

**while(ptr!=head);**

**return temp;**

**}**

**//return temp;**

**}**

**// 2-Append a node to a list**

**void appendNode(Node \*new\_node)**

**{**

**if(nodeExist(new\_node->key)!=NULL)**

**{**

**cout<<"Node Already exits with the key value: "<<new\_node->key<<". Append another node with the diifferent Key value"**

**<<endl;**

**}**

**else**

**{**

**if(head==NULL)**

**{**

**head=new\_node;**

**new\_node->next=head;**

**cout<<"Node Appended at first Head position"<<endl;**

**}**

**else**

**{**

**Node\* ptr=head;**

**while(ptr->next!=head)**

**{**

**ptr=ptr->next;**

**}**

**ptr->next=new\_node;**

**new\_node->next=head;**

**cout<<"Node Appended"<<endl;**

**}**

**}**

**}**

**// 3. Prepend Node- Attach a Node at Start**

**void prependedNode(Node \*new\_node)**

**{**

**if(nodeExist(new\_node->key)!=NULL)**

**{**

**cout<<"Node Already exits with the key value: "<<new\_node->key<<". Append another node with the diifferent Key value"**

**<<endl;**

**}**

**else**

**{**

**Node\* ptr=head;**

**while(ptr->next!=head)**

**{**

**ptr=ptr->next;**

**}**

**ptr->next=new\_node;**

**new\_node->next=head;**

**head=new\_node;**

**cout<<"Node Appended"<<endl;**

**}**

**}**

**//4. Insert a node after particular node in the list**

**void insertNodeAfter( int k, Node\* new\_node)**

**{**

**Node\* ptr=nodeExist(k);**

**if(ptr==NULL)**

**{**

**cout<<"No Node Existswith the key value OF: "<<k<<endl;**

**}**

**else**

**{**

**if( nodeExist(new\_node->key)!=NULL)**

**{**

**cout<<"Node Already exits with the key value: "<<new\_node->key<<". Append another node with the diifferent Key value"**

**<<endl;**

**}**

**else**

**{**

**if(ptr->next==head)**

**{**

**new\_node->next=head;**

**ptr->next=new\_node;**

**cout<<"Node Inserted At the End "<<endl;**

**}**

**else**

**{**

**new\_node->next=ptr->next;**

**ptr->next=new\_node;**

**cout<<"Node Inserted in between"<<endl;**

**}**

**}**

**}**

**}**

**// 5. Delete a Node by Unique Key**

**void deleteNodeByKey(int k)**

**{**

**Node\* ptr= nodeExist(k);**

**if(ptr==head)**

**{**

**if(head->next==NULL)**

**{**

**head=NULL;**

**cout<<"Head node Unliked...List Empty";**

**}**

**else**

**{**

**Node\* ptr1 =head;**

**while(ptr->next!=head)**

**{**

**ptr1=ptr->next;**

**}**

**ptr1->next=head->next;**

**head=head->next;**

**cout<<"Node UNLIKED with the Keys Value"<<k<<endl;**

**}**

**}**

**else**

**{**

**Node\* temp=NULL;**

**Node\* prevptr = head;**

**Node\* currentptr = head->next;**

**while(currentptr!=NULL)**

**{**

**if(currentptr->key==k)**

**{**

**temp = currentptr;**

**currentptr=NULL;**

**}**

**else**

**{**

**prevptr = prevptr->next;**

**currentptr = currentptr->next;**

**}**

**}**

**prevptr->next = temp->next;**

**cout<<"Node UNLIKED with the Keys Value : "<<k<<endl;**

**}**

**}**

**// 6th Update Node by Specific Key**

**void updateNodeByKey(int k, int new\_data)**

**{**

**Node\* ptr= nodeExist(k);**

**if(ptr!=NULL)**

**{**

**ptr->data = new\_data;**

**cout<<"Node Data Updated Successfully...";**

**}**

**else**

**{**

**cout<<"Node Doesn't Exist with the Key Value: "<<k<<endl;**

**}**

**}**

**// 7th Printing**

**void printList()**

**{**

**if(head==NULL)**

**{**

**cout<<"No Nodes in Singly Linked List";**

**}**

**else**

**{**

**cout<<"\nHead Address: "<<head<<endl;**

**cout<<"Singly Linked List Values: "<<endl;**

**Node\* temp = head;**

**do**

**{**

**cout<<"("<<temp->key<<","<<temp->data<<","<<temp->next<<") --> ";**

**temp = temp->next;**

**}**

**while(temp!=head);**

**}**

**}**

**};**

**// Main functoin**

**int main()**

**{**

**CircularLinkedList obj;**

**int option;**

**int key1, k1,data1;**

**do**

**{**

**cout<<"\nWhat operation you want to perform? Select option number. Enter 0 to exit."<<endl;**

**cout<<"1. appendNode()"<<endl;**

**cout<<"2. prependNode()"<<endl;**

**cout<<"3. insertNodeAfter()"<<endl;**

**cout<<"4. deleteNodeByKey()"<<endl;**

**cout<<"5. updateNodeByKey()"<<endl;**

**cout<<"6. print()"<<endl;**

**cout<<"7. Clear Screen"<<endl<<endl;**

**cin>>option;**

**Node\* n1 = new Node();**

**// Node n1**

**switch(option)**

**{**

**case 0:**

**break;**

**case 1:**

**cout<<"Append Node Operation \nEnter key & data of the Node to be Appended"<<endl;**

**cin>>key1;**

**cin>>data1;**

**n1->key=key1;**

**n1->data=data1;**

**obj.appendNode(n1);**

**// cout<<n1.key<<" = "<<n1.data<<endl;**

**break;**

**case 2:**

**cout<<"Prepend Node Operation \nEnter key & data of the Node to be Prepended"<<endl;**

**cin>>key1;**

**cin>>data1;**

**n1->key=key1;**

**n1->data=data1;**

**obj.prependedNode(n1);**

**// cout<<n1.key<<" = "<<n1.data<<endl;**

**break;**

**case 3:**

**cout<<"Insert Node after Operation \nEnter key of existing Node after which you want to Insert this node: "<<endl;**

**cin>>k1;**

**cout<<"Enter key & data of the New Node first: "<<endl;**

**cin>>key1;**

**cin>>data1;**

**n1->key=key1;**

**n1->data=data1;**

**obj.insertNodeAfter(k1,n1);**

**break;**

**case 4:**

**cout<<"Delete Node By Key Operation \nEnter key of the Node to be deleted: "<<endl;**

**cin>>k1;**

**obj.deleteNodeByKey(k1);**

**break;**

**case 5:**

**cout<<"Update Node By Key Operation \nEnter key & New data to be updated: "<<endl;**

**cin>>key1;**

**cin>>data1;**

**obj.updateNodeByKey(key1,data1);**

**break;**

**case 6:**

**obj.printList();**

**break;**

**case 7:**

**system("cls");**

**break;**

**default:**

**cout<<"Enter Proper Option number "<<endl;**

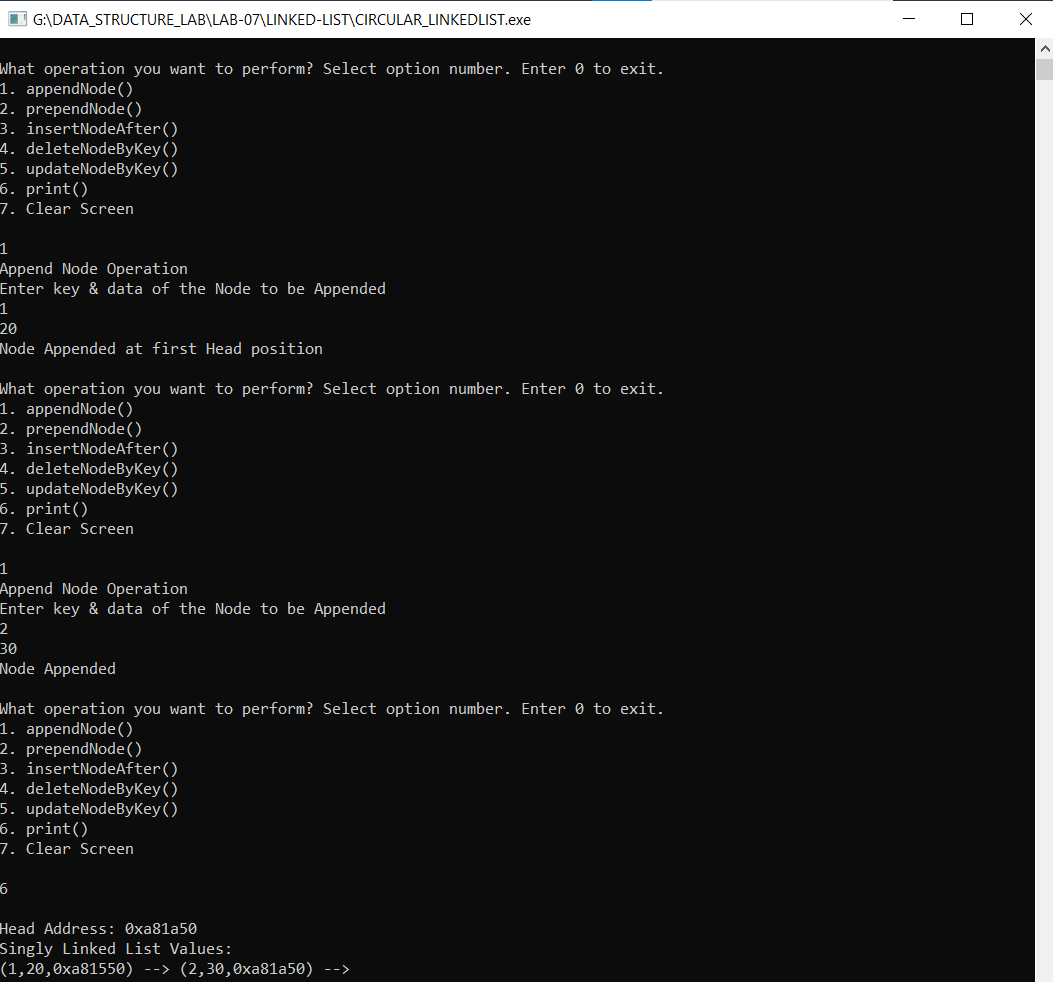
**}**

**}while(option!=0);**

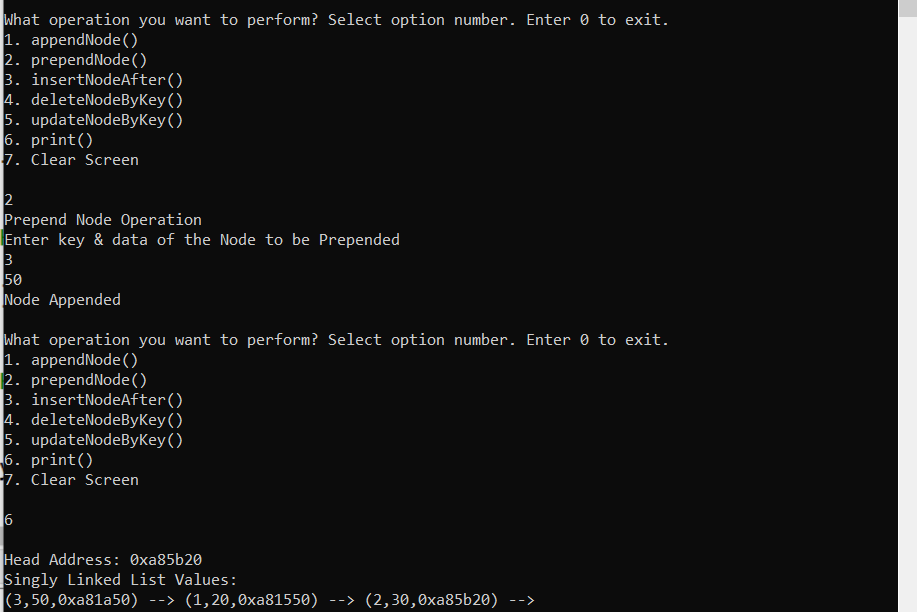
**}**

**Output:**

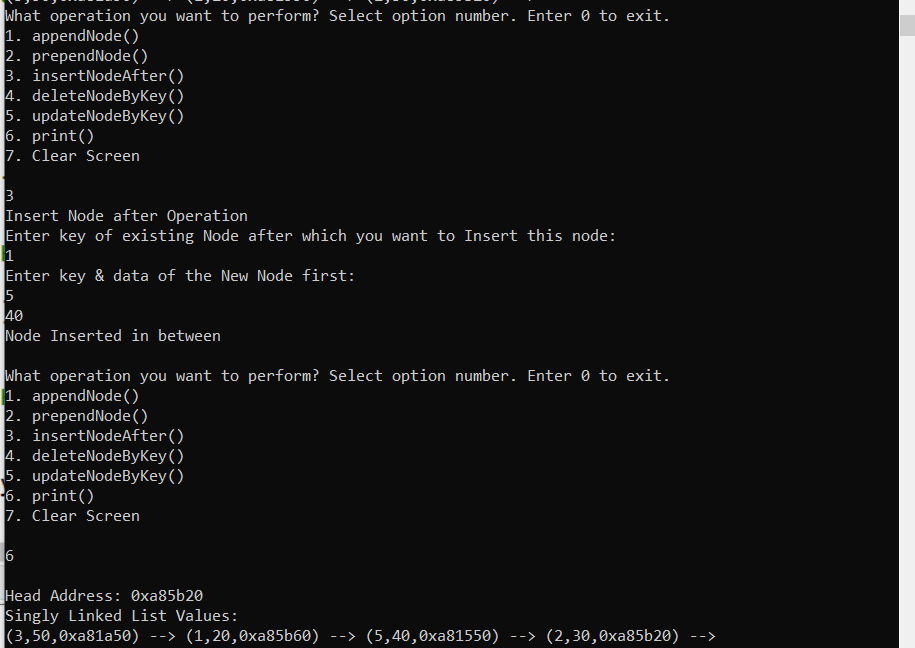
1. **Append Node:**

****

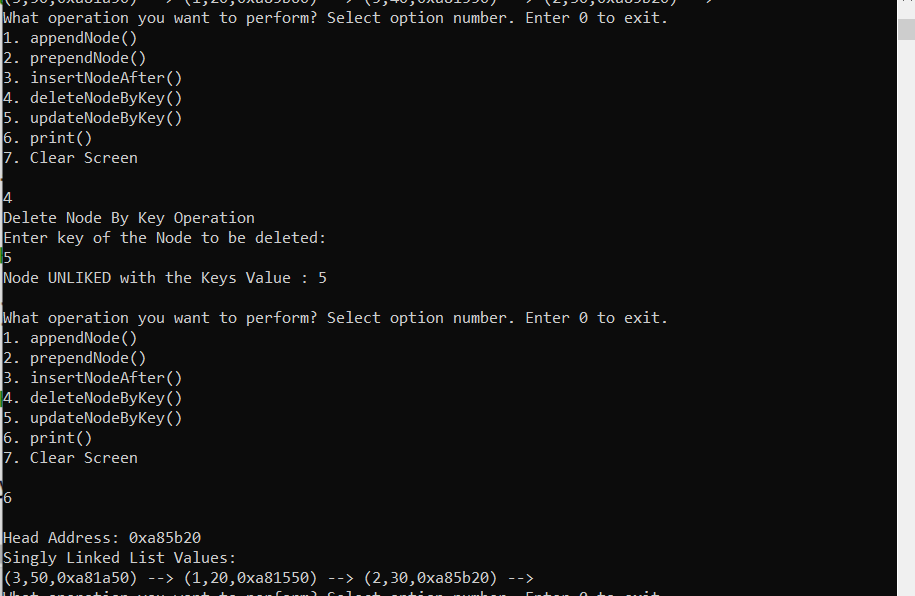
1. **Prepend Node:**

****

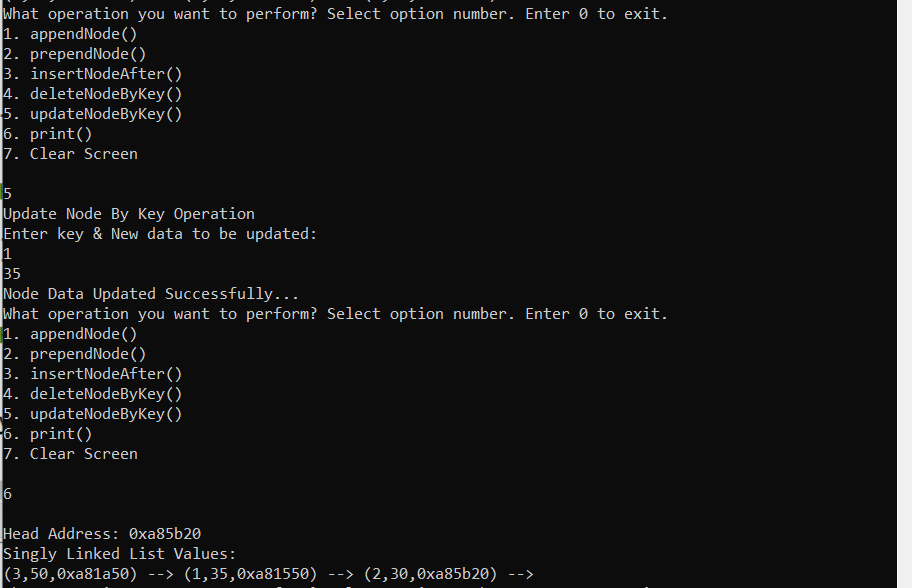
1. **Insert Node After Key:**

****

1. **Delete Node by Key:**

****

1. **Update Node by Key:**

****

**Results & Observations:**

In this Lab I've learned about the concept of Circular Link List & also understand that how to create node and how to link the node one another, I also understand singly and doubly linked list. Further I understand the Circular linked that how to link the tail to the head and then I learned that how to append, prepend a node, how to insert a node after specific node by using key and also update and delate the node by using key.