|  |  |
| --- | --- |
| download | COMSATS University Islamabad, Vehari Campus Department of Computer Science |

**Class: BCS-SP22-4A Submission Deadline: 9 oct 2023**

**Subject: Data Structures and Algorithms-Lab**

**Name: Muhammad Abdullah**

**Instructor: Yasmeen Jana Max Marks: 10 Reg. No: SP22-BCS-036**

**Email:** [**yasmeenjana@cuivehari.edu.pk**](mailto:yasmeenjana@cuivehari.edu.pk)

**You can ask queries related to Lab Activities on the above email.**

**Activity1:**

#include <iostream>

struct Node {

int data;

Node\* next;

};

void displayLinkedListInfo(Node\* head) {

Node\* ptr = head;

std::cout << "\*\*head address:\*\* " << &head << std::endl;

std::cout << "head content:" << head << std::endl;

while (ptr !=NULL) {

std::cout << "\*\*ptr address :\*\*" << ptr << std::endl;

std::cout << " ptr content :" << ptr << std::endl;

std::cout << "ptr data :" << ptr->data << std::endl;

std::cout << "Ptr :" << ptr << std::endl;

if (ptr->next != NULL) {

std::cout << "Ptr - >next:" << ptr->next << std::endl;

std::cout << "Ptr - >data:" << ptr->next->data << std::endl;

}

ptr = ptr->next;

}

}

int main() {

Node\* head = new Node{1, NULL};

head->next = new Node{2, NULL};

head->next->next = new Node{20, NULL};

head->next->next->next = new Node{30, NULL};

displayLinkedListInfo(head);

while (head != NULL) {

Node\* temp = head;

head = head->next;

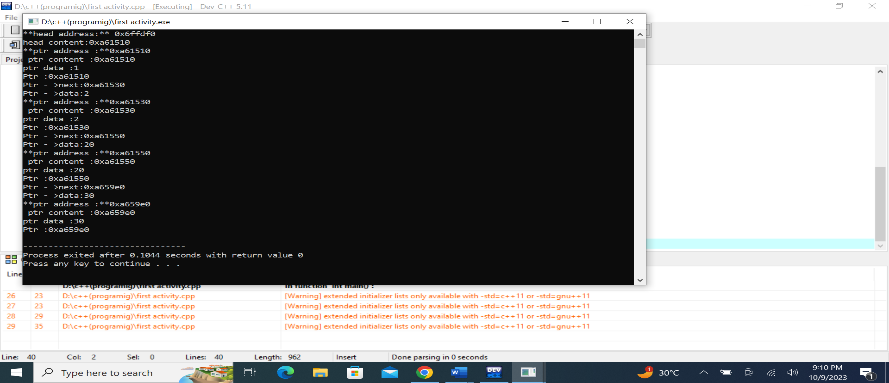
delete temp;

}

return 0;

}

Output:



**Activity 2:**

Write a program that will implement single, doubly, and circular linked link list operations by showing a menu to the user.

The menu should be:

**Which linked list you want:**

1: Single

2: Double

3: Circular

After the option is chosen by the user:

**Which operation you want to perform:**

1: Insertion

2: Deletion

3: Display

4: Reverse

4: Seek

5: Exit

**Let's suppose, the user has chosen the insertion option then the following menu should be displayed:**

1: insertion at beginning

2: insertion at end

3: insertion at the specific data node

A sample output screenshot is below:

**Code:**

#include<iostream>

using namespace std;

class Node {

public:

int data;

Node\* next;

Node\* prev; // for doubly linked list

Node(int val) {

data = val;

next = NULL;

prev = NULL;

}

};

class LinkedList {

public:

Node\* head;

Node\* tail; // for doubly linked list

int size;

LinkedList() {

head = NULL;

tail = NULL;

size = 0;

}

void insertAtBeginning(int data) {

Node\* newNode = new Node(data);

newNode->next = head;

head = newNode;

size++;

}

void insertAtEnd(int data) {

Node\* newNode = new Node(data);

if (head == NULL) {

head = newNode;

tail = newNode;

} else {

tail->next = newNode;

newNode->prev = tail;

tail = newNode;

}

size++;

}

void insertAtSpecific(int data, int target) {

Node\* newNode = new Node(data);

Node\* current = head;

while (current) {

if (current->data == target) {

newNode->next = current->next;

current->next = newNode;

size++;

return;

}

current = current->next;

}

}

void deleteNode(int data) {

Node\* current = head;

if (current == NULL) {

return;

}

if (current->data == data) {

head = current->next;

delete current;

size--;

return;

}

while (current->next) {

if (current->next->data == data) {

Node\* temp = current->next;

current->next = temp->next;

if (temp->next) {

temp->next->prev = current;

}

delete temp;

size--;

return;

}

current = current->next;

}

}

void display() {

Node\* current = head;

while (current) {

cout << current->data << " -> ";

current = current->next;

}

cout << "nullptr" << endl;

}

void reverse() {

Node\* prevNode = NULL;

Node\* current = head;

while (current) {

Node\* nextNode = current->next;

current->next = prevNode;

current->prev = nextNode; // for doubly linked list

prevNode = current;

current = nextNode;

}

head = prevNode;

}

int seek(int data) {

Node\* current = head;

int index = 0;

while (current) {

if (current->data == data) {

return index;

}

current = current->next;

index++;

}

return -1;

}

};

int main() {

LinkedList linkedList;

int choice;

cout << "Which linked list you want:" << endl;

cout << "1: Single" << endl;

cout << "2: Double" << endl;

cout << "3: Circular" << endl;

cin >> choice;

while (choice < 1 || choice > 3) {

cout << "Invalid choice. Please enter 1, 2, or 3: ";

cin >> choice;

}

while (true) {

cout << "\nWhich operation you want to perform:" << endl;

cout << "1: Insertion" << endl;

cout << "2: Deletion" << endl;

cout << "3: Display" << endl;

cout << "4: Reverse" << endl;

cout << "5: Seek" << endl;

cout << "6: Exit" << endl;

cin >> choice;

switch (choice) {

case 1: {

int insertChoice;

cout << "1: Insertion at beginning" << endl;

cout << "2: Insertion at end" << endl;

cout << "3: Insertion at specific data node" << endl;

cin >> insertChoice;

int data;

cout << "Enter data: ";

cin >> data;

if (insertChoice == 1) {

linkedList.insertAtBeginning(data);

} else if (insertChoice == 2) {

linkedList.insertAtEnd(data);

} else if (insertChoice == 3) {

int target;

cout << "Enter the target data: ";

cin >> target;

linkedList.insertAtSpecific(data, target);

} else {

cout << "Invalid choice" << endl;

}

break;

}

case 2: {

int data;

cout << "Enter data to delete: ";

cin >> data;

linkedList.deleteNode(data);

break;

}

case 3:

linkedList.display();

break;

case 4:

linkedList.reverse();

break;

case 5: {

int data;

cout << "Enter data to seek: ";

cin >> data;

int index = linkedList.seek(data);

if (index != -1) {

cout << "Data " << data << " found at index " << index << endl;

} else {

cout << "Data " << data << " not found in the linked list" << endl;

}

break;

}

case 6:

return 0;

default:

cout << "Invalid choice" << endl;

}

}

return 0;

Out put:

