

**Name: Muhammad Jahanzaib**

**Sap ID: 37878**

**Section: BSSE-3**

**Course: DSA**

**TASK NO:1**

// SingleLinkedList.cpp : This file contains the 'main' function. Program execution begins and ends there.

//

#include <iostream>

using namespace std;

class node

{

public:

string storyName;

int priority;

int teamNumber;

int storyNumber;

bool isDone;

node\* next;

node(string story)

{

this->storyName = story;

this->next = NULL;

}

//Destructor

~node()

{

}

};

class SingleLinkedList

{

private:

node\* head, \* tail;

public:

SingleLinkedList()

{

head = NULL;

tail = NULL;

}

//isEmpty

bool isEmpty()

{

if (head != NULL)

{

return true;

}

else

{

return false;

}

}

//insertStoryAtLast

void insertStoryAtLast(string n)

{

if (head == NULL)

{

node\* temp = new node(n);

temp->storyName = n;

temp->next = NULL;

if (head == NULL)

{

head = temp;

tail = temp;

}

else

{

tail->next = temp;

temp = tail;

}

}

else

{

node\* temp = new node(n);

temp->storyName = n;

tail->next = temp;

temp = tail;

}

}

//display

void printList()

{

node\* curr;

curr = head;

while (curr != NULL)

{

cout << curr->storyName<<endl;

curr = curr->next;

}

}

//Void removeStory(string)

void removeStory()

{

node\* curr;

curr = head;

curr = curr->next;

delete curr;

cout << "\nDelete RemoveStory\n";

}

//. Bool Search(string)

bool Search(string)

{

}

// Void searchKeyword(string)

void searchKeyword(string search)

{

node\* curr;

curr = head;

int found = 0;

while (curr != NULL)

{

if (curr->storyName == search)

{

cout << curr->storyName << endl;

found++;

break;

}

curr = curr->next;

}

if (found == 0)

{

cout << "\nNot Found!!!" << endl;

}

}

//void assignPrority( string, int)

void assignPrority(string s, int n)

{

}

//void assignTeamNumber( string, int)

void assignTeamNumber(string s, int n)

{

node\* newnode = new node(s);

newnode->teamNumber = n;

newnode->storyName = s;

newnode->next = NULL;

if (head == NULL)

{

head = newnode;

tail = newnode;

}

else

{

tail->next = newnode;

tail = newnode;

}

}

//void orberByPriority()

void orberByPriority()

{

}

//.void findByPriority(int)

void findByPriority(int n,int search)

{

node\* curr;

curr = head;

int found = 0;

while (curr != NULL)

{

if (curr->priority == search)

{

cout << curr->storyName << endl;

found++;

break;

}

curr = curr->next;

}

if (found == 0)

{

cout << "\nNot Found!!!" << endl;

}

}

//.void replaceStory( string, StoryNode)

void replaceStory( string s,node \*StoryNode )

{

StoryNode = head;

int found = 0;

while (StoryNode != NULL)

{

if (StoryNode->storyName== s)

{

cout<<"Update Story Name";

cin >> StoryNode->storyName;

found++;

break;

}

StoryNode = StoryNode->next;

}

if (found == 0)

{

cout << "\nNot Found!!!" << endl;

}

}

//.void addAtFirst()

void addAtFirst(string n)

{

node \*temp=new node(n);

temp->storyName=n;

temp->next=head;

head=temp;

}

//.void addAfter(string)

void addAfter(string n)

{

if (head == NULL)

{

node\* temp = new node(n);

temp->storyName = n;

temp->next = NULL;

if (head == NULL)

{

head = temp;

tail = temp;

}

else

{

tail->next = temp;

temp = tail;

}

}

else

{

node\* temp = new node(n);

temp->storyName = n;

tail->next = temp;

temp = tail;

}

}

void addBefore(string n)

{

node \*temp=new node(n);

temp->storyName=n;

temp->next=head;

head=temp;

}

};

int main()

{

SingleLinkedList s;

cout << s.isEmpty();

s.insertStoryAtLast("zaibi");

s.removeStory();

s.searchKeyword("fak");

s.assignTeamNumber("hal", 2);

s.addAfter("All") ;

s.addBefore("a");

s.printList();

}

**Task NO: 2**

#include <iostream>

using namespace std;

class node

{

public:

string storyName;

int priority;

int teamNumber;

int storyNumber;

bool isDone;

node\* next;

node\* previous;

node(string story)

{

this->storyName = story;

this->next = NULL;

}

//Destructor

~node()

{

}

};

class SingleLinkedList

{

private:

node\* head;

public:

SingleLinkedList()

{

head = NULL;

}

//isEmpty

bool isEmpty()

{

if (head != NULL)

{

return true;

}

else

{

return false;

}

}

//insertStoryAtLast

void insertStoryAtLast(string n)

{

if (head == NULL)

{

node \*curr;

node\* temp = new node(n);

temp->storyName = n;

temp->next=NULL;

if(head==NULL)

{

temp->previous=NULL;

head=temp;

}

else

{

curr=head;

while(curr->next!=NULL)

{

curr=curr->next;

}

curr->next=temp;

temp->previous=curr;

while(curr->next!=NULL){

curr=curr->next;

}

curr->next=temp;

temp->next=NULL;

temp->previous=curr;

}

}

else

{

node \*temp,\*current;

temp = new node(n);

temp->storyName = n;

current=head;

while(current->next!=NULL){

current=current->next;

}

current->next=temp;

temp->next=NULL;

temp->previous=current;

}

}

//display

void printList()

{

node\* curr;

curr = head;

while (curr != NULL)

{

cout << curr->storyName<<endl;

curr = curr->next;

}

}

//Void removeStory(string)

void removeStory()

{

node\* temp;

temp = head;

head = head->next;

head->previous = NULL;

delete temp;

cout << "\nDelete RemoveStory\n";

}

//. Bool Search(string)

bool Search(string)

{

}

// Void searchKeyword(string)

void searchKeyword(string search)

{

node\* curr;

curr = head;

int found = 0;

while (curr != NULL)

{

if (curr->storyName == search)

{

cout << curr->storyName << endl;

found++;

break;

}

curr = curr->next;

}

if (found == 0)

{

cout << "\nNot Found!!!" << endl;

}

}

//void assignPrority( string, int)

void assignPrority(string s, int n)

{

}

//void assignTeamNumber( string, int)

void assignTeamNumber(string s, int n)

{

node\* newnode = new node(s);

newnode->teamNumber = n;

newnode->storyName = s;

newnode->next = NULL;

if (head == NULL)

{

head = newnode;

}

else

{

}

}

//void orberByPriority()

void orberByPriority()

{

}

//.void findByPriority(int)

void findByPriority(int n,int search)

{

node\* curr;

curr = head;

int found = 0;

while (curr != NULL)

{

if (curr->priority == search)

{

cout << curr->storyName << endl;

found++;

break;

}

curr = curr->next;

}

if (found == 0)

{

cout << "\nNot Found!!!" << endl;

}

}

//.void replaceStory( string, StoryNode)

void replaceStory( string s,node \*StoryNode )

{

StoryNode = head;

int found = 0;

while (StoryNode != NULL)

{

if (StoryNode->storyName== s)

{

cout<<"Update Story Name";

cin >> StoryNode->storyName;

found++;

break;

}

StoryNode = StoryNode->next;

}

if (found == 0)

{

cout << "\nNot Found!!!" << endl;

}

}

//.void addAtFirst()

void addAtFirst(string n)

{

node \*temp;

temp = new node(n);

temp->previous = NULL;

temp->storyName =n;

temp->next = head;

head->previous = temp;

head = temp;

}

//.void addAfter(string)

void addAfter(string n)

{

node \*temp,\*current;

temp = new node(n);

temp->storyName = n;

current=head;

while(current->next!=NULL){

current=current->next;

}

current->next=temp;

temp->next=NULL;

temp->previous=current;

}

void addBefore(string n)

{

node \*temp;

temp = new node(n);

temp->previous = NULL;

temp->storyName =n;

temp->next = head;

head->previous = temp;

head = temp;

}

};

int main()

{

SingleLinkedList s;

cout << s.isEmpty();

s.insertStoryAtLast("zaibi");

s.removeStory();

s.searchKeyword("fak");

s.assignTeamNumber("hal", 2);

s.addAfter("All") ;

s.addBefore("a");

s.printList();

}