National University of Computer and Emerging Sciences



Laboratory Manual

for

Computer Programming

(CS 104)

|  |  |
| --- | --- |
| Course Instructor | Ms. Aamir |
| Lab Instructor(s) | Mr. Waqas Manzoor  Ms. Farwa Batool |
| Section | CS-C |
| Semester | Spring 2017 |

Department of Computer Science

FAST-NU, Lahore, Pakista

**Question # 1 (Struct)**

Design a structure Song that contains the track no, artist name and year it was released. In the main function your program should create an array of 100 pointers. Each pointer in the array should point to a variable of type Song. Moreover you need to write the following functions.

Insert : Adds a new item in the list of type Song.

Initialize: Which initializes the variable of type Song.

Display : Which prints the contents of any given Song.

Display Library: That prints all the items added to the list of songs so far.

Remove; It should remove an item from the list.

Ask index of song from user to remove.

Remove All: It should delete all the items in the list of Song type.

**Question # 2 (Link List)**

Create a Struct name as Node have two attributes, int x and Node next.

Use this Struct for your link list implementation. Create following functions

void DisplayLinkList(node \* root)

int getLength(node \* root)

void insertElement(node \* root) // take input, value should +ve integer

bool deleteElement(node \* root)

void insertElementAt(node \* root , int position) // position of root element is, take input

bool deleteElementAt(node \* root , int position) // position of root element is, take input

void deleteLinkList(node \* root)

To access the data of Node (it is a struct) pointer use -> operator.

int main() //This is only for illustration.

{

Node \* root = NULL;

insertElement(root);

DisplayLinkList(root);

if(deleteElement(root)

{

cout << “element deleted successfully\n”;

}

else

{

cout << “Element deletion fail\n”;

}

deleteLinkList(root);

return 0;

}

There should be no memory leakage and no dangling pointer

**Question # 3 (Pointers Array of Link List)**

Extend the question-2 and make a pointer array just like:

Node \*\* nodesPtr = new Node \* [size];

Where size can be any number greater than 10. Now we have an array in which every single index can make a new Linked\_List. So be careful. Use the same functions given in Question-2 and complete the task writing a new main() function which will be approximately like this:

Node \*\* initNodePtr()

{

Node \*\* nodePtr = new Node \* [size];

//Now initialize a node on every index.

}

int main() //This is only for illustration.

{

Node \*\* nodePtr = initNodePtr();

new Node \* [size];

nodePtr[0]= new Node;

insertElement(nodePtr[0]);

DisplayLinkList(nodePtr[0]);

if(deleteElement(nodePtr[0])

{

cout << “element deleted successfully\n”;

}

else

{

cout << “Element deletion fail\n”;

}

deleteLinkList(nodePtr[0]);

return 0;

}

“Imagination is more important than Knowledge” ***[ …*** Albert Einstein ***… ]***

***l16-4078@***