TECHNICAL GUIDE

Data Structures Learning Software

Contents

1. LOGIN AND SIGNUP

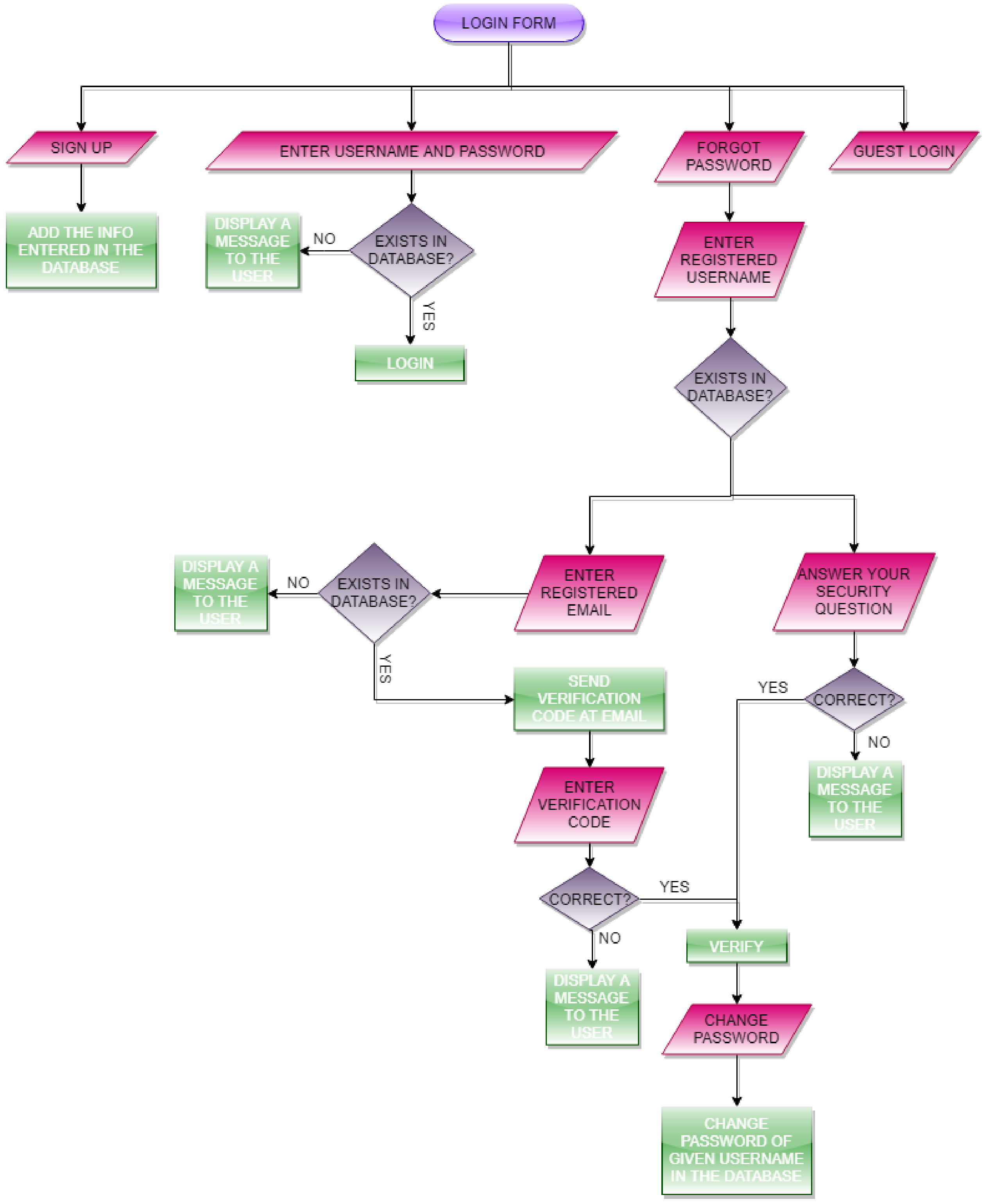
1. LoginForm:
This is the form where users can login or signup. The **loginpanel** is the main panel of this form. It takes in your username and password using **UsernameTextBox** and **PasswordTextBox** respectively. The checkbox **LoginPagePasswordShowcheckBox** will show/hide password depending on its state. The **LoginButton** will log the user in.

If the user has forgotten their password, they click on **ForgotPasswordButton** or if they want to sign up, they click on **RegisterButton** or on **GuestLoginButton** if they don't want to login using an account.

If the user clicks on **RegisterButton**, we show the **Registerpanel** where they are asked the basic info such as desired username, password, name, email, security question and their account type.

If the user clicks on **ForgotPasswordButton**, we show the **ForgotPasswordpanel**, where the user is asked their registered username in **ForgotPasswordUsernamepanel**, after which they're given two options in the panel **ForgotPasswordVerificationpanel** whether they have to verify their identity: security question or email verification. Depending on the radio button they checked, they're shown either of two panels **ForgotPasswordEmailpanel** and **ForgotPasswordQuestionpanel**. Here they either enter their registered email or answered the security question. If they chose email, they have to enter the verification code sent to their email on **ForgotPasswordEmailCodepanel**, and once that is verified, they're prompted to enter new password on **ChangePasswordpanel**.

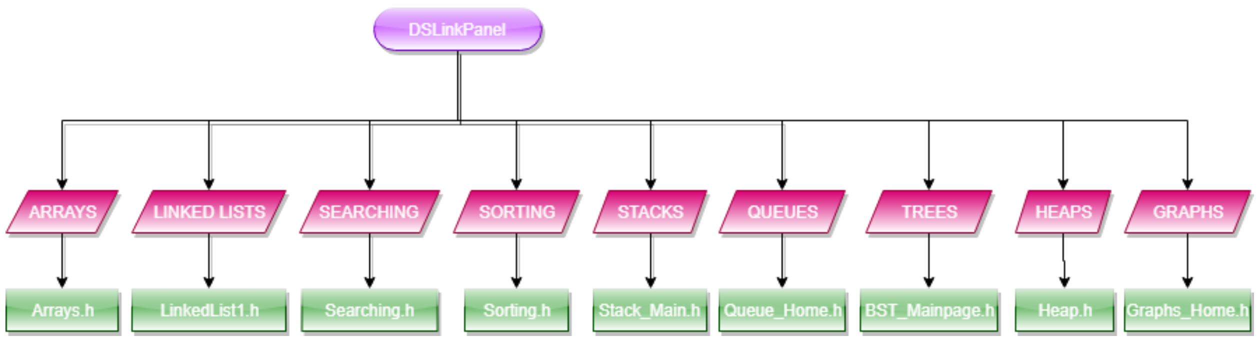
(REFER FLOWCHART BELOW FOR MORE CLARITY)


LOGIN FORM
SIGN UP
ENTER USERNAME AND PASSWORD
FORGOT PASSWORD
GUEST LOGIN
ADD THE INFO ENTERED IN THE DATABASE
DISPLAY A MESSAGE TO THE USER
NO
EXISTS IN DATABASE?
YES
LOGIN
ENTER REGISTERED USERNAME
EXISTS IN DATABASE?
DISPLAY A MESSAGE TO THE USER
NO
EXISTS IN DATABASE?
ENTER REGISTERED EMAIL
ANSWER YOUR SECURITY QUESTION
YES
SEND VERIFICATION CODE AT EMAIL
YES
CORRECT?
NO
DISPLAY A MESSAGE TO THE USER
ENTER VERIFICATION CODE
CORRECT?
YES
NO
VERIFY
DISPLAY A MESSAGE TO THE USER
CHANGE PASSWORD
CHANGE PASSWORD OF GIVEN USERNAME IN THE DATABASE

2. HOMEPAGE

Homepage:

This is the homepage of the user. Here, in the panel **DSLinkPanel**, there are buttons which lead to the specific data structure.

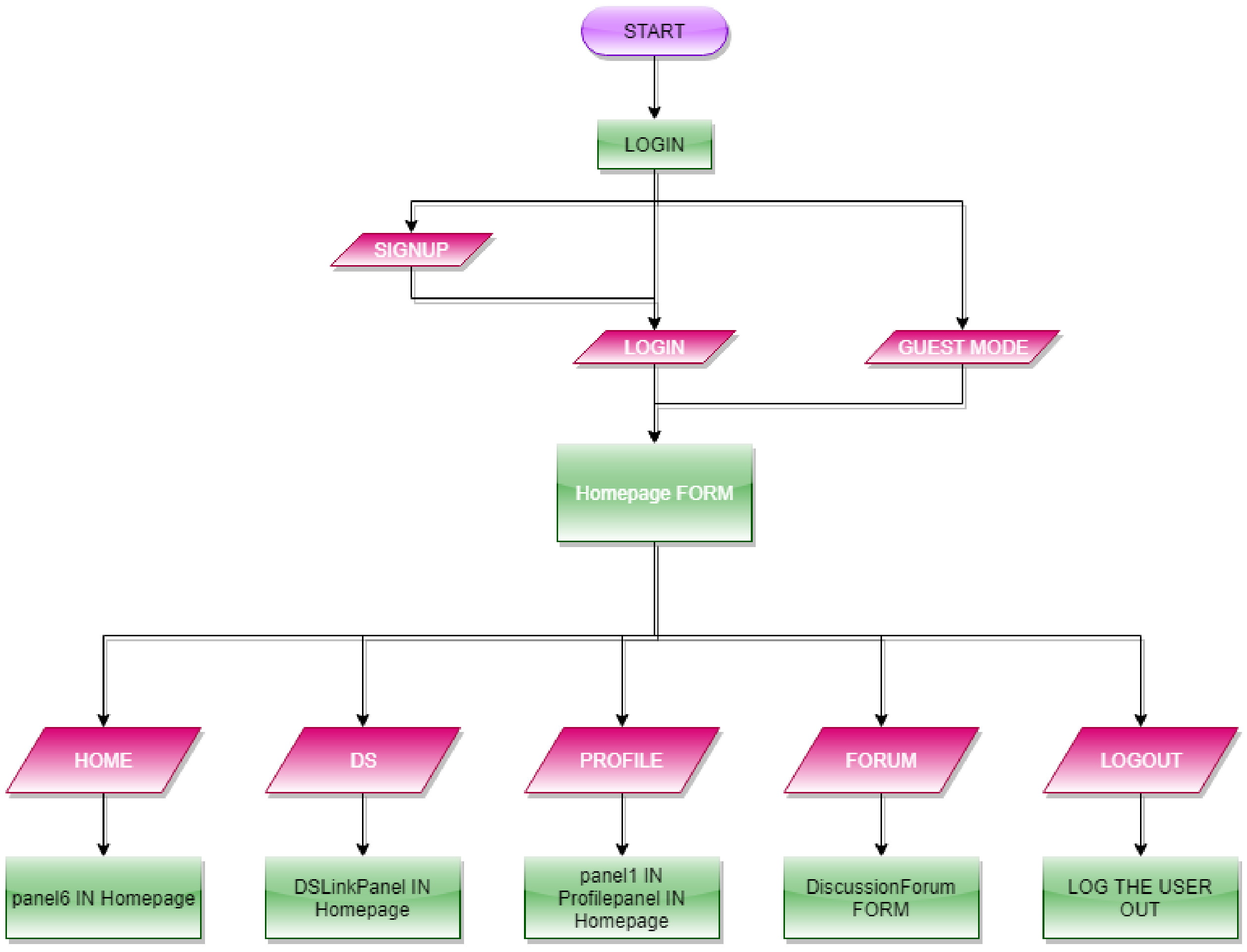


It also has a **DSSuggestionsButton** which brings the other panel **SuggestionPanel** to the front. In this panel, the user can suggest databases to be included in the software.

The panel called **Profilepanel**, which contains the basic info about the user profile.
In this panel, **panel1** is where the user can view their basic info of their profile.
The text box called **ProfilecodetextBox** which is where the user enters the verification code sent to their email. In the **panel3**, the user is able to upload their photo as the profile picture using the **ProfilePictureChangebutton.**

Below these two are the options to change password and change the security question.
Panels **Profilepanel1** through **Profilepanel6** act as lines.

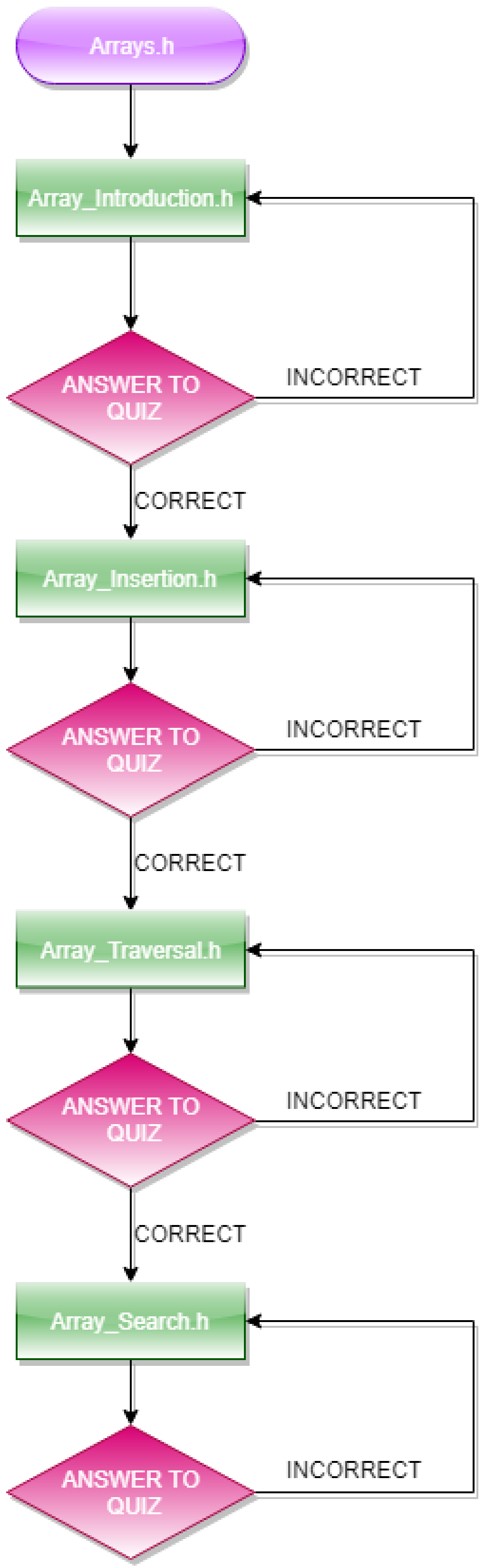
The panel **Homepanel** has the **panel6** which shows the user's progress through each of the data structures using progress bars.
Database is accessed using *System::Data::OleDb* which can read and write data in the database.



3. ARRAY

1. Array_Insertion:

This form has a panel named **insert_animation**. This panel has the part where the user is able to use the interactive software to learn arrays.
pictureBox1 is the picture box that contains some basic info about array insertion.
insertvalue and **insertindex** are the text boxes that take the user input and insert the appropriate value at the appropriate position.



This form has one timer called **insertimer**. Every tick of the insertimer if there is a value to be inserted, at the appropriate index and the textboxes are cleared.
The rest of the form includes labels at their proper positions and showing any relevant information.

2. Array_Introduction:

This form contains only the basic introduction, only picture boxes and labels.

3. Array_Search:

panel1 contains the box that has the labels **label_rec_1** through **label_rec_7** that contain the numbers in the array. This panel also contains the picture box **arrow** that is the arrow that points to the current element while searching, animated according to **timer1**.
Input is the text box that takes in user input for the element to search in the array. **button3** is the "Start" button and **Reset_Button** resets the input text box. **timer1** is the timer started when **button3** is pressed. At every tick, every label is compared to the input linearly searching for the input. If the element is found(or not), the timer is disabled and the result is shown.

4. Array_Traversal:

This form contains the panel **panelForTraversal** which includes the interactive teaching animation. It has three buttons, **btnStart**, **btnPause** and **btnPlay** which start, pause and play the animation respectively.
It also has a label **Llabel** which highlights a given label, labels **LB1** through **LB7** which

contain the contents of the array. It also contains labels **lbel1** through **lbel7**
This form contains the timer **timerTraversal**. Every tick **Llabel** is moved forward till we reach a LB. When all LB's are traversed the timer is disabled.

5. Arrays:

This form is the backbone for the entire part of Arrays, as it contains the buttons for various modules regarding arrays.

---

4. LINKED LIST

1. LinkedList:

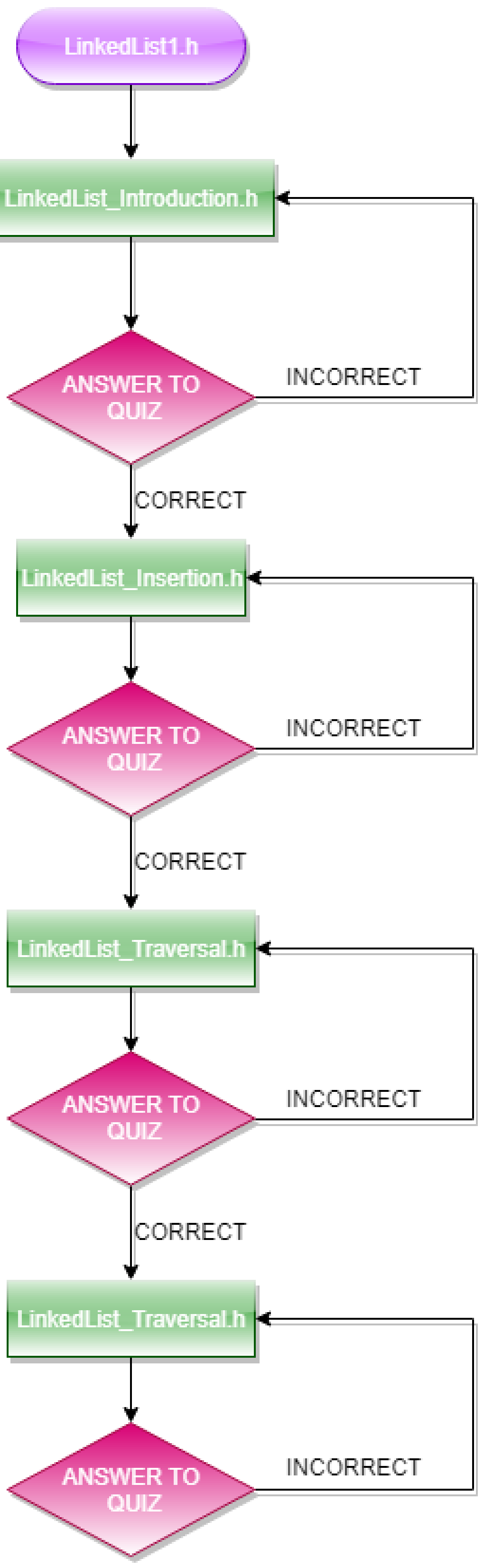
This form is about linear search in a linked list. This form should give output “yes” if the number is in the linked list and “no” if it isn't.
The interactive part is **panel1** with timer **timer1**. The user can add a value using **richTextBox1** <why a rich text box?> and **button3**, while search using **textBox1** and **button4**. The buttons **button5, button6** and **button7** respectively pause the animation, resume the animation, and reset the linked list.
The rest of the form is **quizPanel**, and the labels and picture-boxes containing relevant information.

2. LinkedList1:

This is the form that is the backbone for all the forms regarding linked list as it contains the buttons that point to the different modules about linked list.

3. LinkedList_Deletion:

Here the interactive panel is the **panel2**. This time, the animation has three timers **deletetimer**, **deletefixtimer** and **tailfixtimer**.
The user enters the value to delete in the **deletevalue** textbox and clicks on the **delete_ll_but** button. This starts an animation to find and delete the value in the linked list.
The rest of the form contains labels and picture-boxes that contain the relevant information.



4. LinkedList_Insertion:

The interactive part in this form is in the panel **insert_animation_ll** which uses two timers **insertimerll** and **tailtimer**. The user can input their values using **insert_ll_tf** and **insert_but_ll**. The panel contains labels **ll1** through **ll7** for the linked list. **insert_ll_label** shows the value to be inserted. The rest of the form is labels and picture-boxes containing relevant information.

5. LinkedList_Introduction:

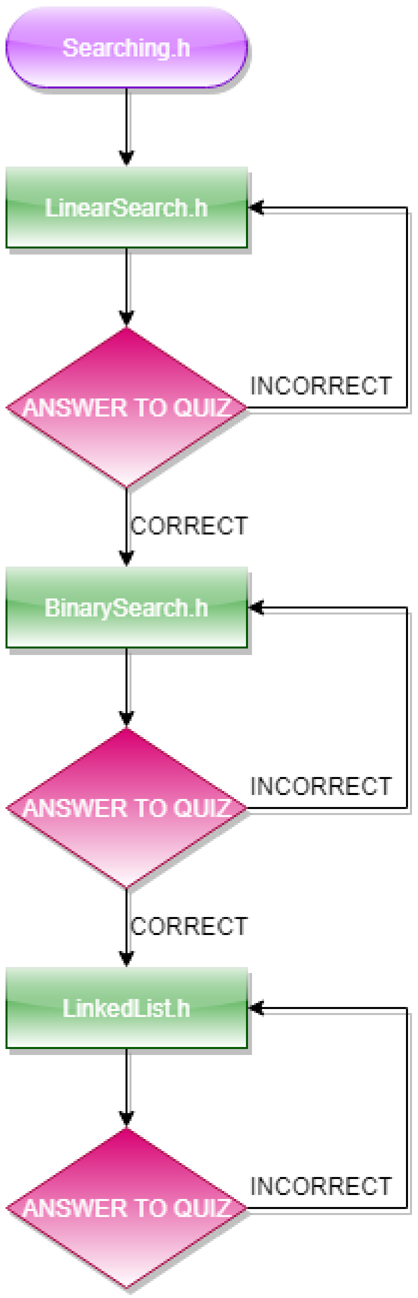
Gives an introduction to the linked lists. Has only information-containing labels and picture-boxes.

6. LinkedList_Traversal:

The interactive part of this form is in the panel **panel3**, which contains the animations which run on timer **timerLTrav**. The labels **lli1** through **lli7** contains the numbers in the linked list. The label **arrowlli** is the arrow that points to each element while traversing along the linked list <right?>
The buttons **but_start**, **but_play** and **but_pause** respectively start, play and pause the animation. The rest of the form is labels explaining the implementation.

5. SEARCHING

This form is the backbone for all the modules regarding Searching Algorithms.



1. LinearSearch:

The interactive part of this panel is the **AnimationPanel** which contains the animations to the timer **timer1**.

The user can input a value using **txtInput** and **btnEnterInput** and enter a value in **txtSearch** and click on **btnSearch** to search it. **btnAnimate**, **btnPause** and **btnReset** respectively start the animation, pause the animation and reset the array.
The rest of the form contains the **quizPanel** for the quiz question, and the labels and picture-boxes containing the relevant information.

2. BinarySearch:

The binary search contains a panel regarding interactive teaching called **AnimationPanel.**

Textboxes **txtInput** and **txtSearch** respectively take input from the user and the value to search in the array. **btnAnimate**, **btnPause** and **btnReset** respectively animate, pause and reset the array, and the animations are according to the timer **timer1**.
In every tick of the **timer1** the variables **si**(start index), **ei**(end index) and **mid** are updated. This is followed by modifying the **label_si,label_ei** and **label_mid.** Then value in **label_mid** is compared and the indices are updated.

quizPanel is the panel that will have the quiz question, which is chosen randomly by using the **moduleQuiz()**

from **quiz.h.** The rest of the form is labels and picture-boxes containing relevant information.

---

6. SORTING

1. Sorting:

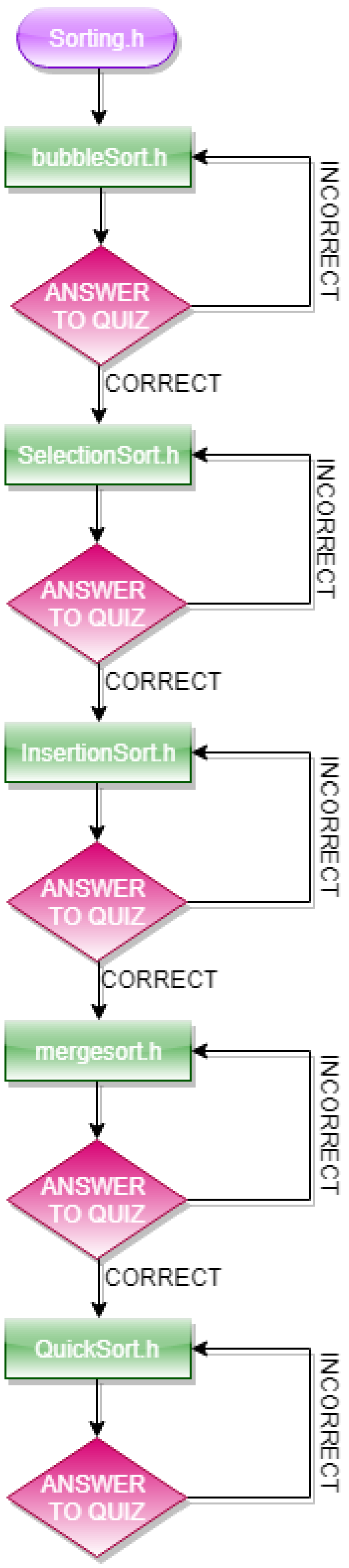
This is the backbone for all the modules regarding Sorting, as it contains all the respective buttons.

2. bubbleSort:

The interactive panel in this form is named as **panel1**. It has picture-boxes **p1** through **p15** that contain arrows and labels **l1** through **l15** that have each picture-box corresponding to them. Button **btnAdd** adds the number taken in as input from the text box **txt1**. The button **btnBS** starts the bs, and the buttons **btnIncreaseSpeed** and **btnDecreaseSpeed** respectively increase and decrease the animation speed, which is regulated according to the timer **timer1**, while **btnPause** and **btnReset** respectively pause and reset the animation.
The **quizPanel** contains the quiz question <incomplete?> and the rest of the form is labels and picture-boxes containing relevant information.

3. InsertionSort:

The interactive part in this form is the **panel1** which, as the others, contains the array and the animations to sort is as needed. It contains fifteen picture boxes named **p1** through **p15**, each with their labels **l1** through **l15**. The labels contain the numbers in the array, while the picture boxes contains arrows <why?>
The user can add a value using **txt1** and **btnAdd**. The button **btnIS** starts the animation and the Insertion Sort, while **btnIncreaseSpeed**, **btnDecreaseSpeed**, **btnPause** and **btnReset** respectively increase animation speed, decrease animation speed, pause the animation and reset the array. The animations are to the timer **timer1**. <explain>
The rest of the form is **quizPanel**, where the quiz question appears, labels and picture-boxes containing the relevant information.



4. QuickSort:

The interactive part of this form is called **panel1** and it has three timers **timer1**, **timer2** and **timer3**. The user can input using **txtInput** and **button3**. After entering all the values, sorting begins by **button1**. **button5**, **buttonresume** and **button7** respectively pause, resume the animation and reset the array.

Now, in quicksort, we also show the pivot element in **label11**, and show the elements we are currently comparing in **label20** and **label22**, and show the elements we are swapping in **label10** and **label24**.

The rest of the form contains **quizPanel** for the quiz question, a picture-box explaining quicksort and various labels that show the implementation of quicksort.

5. mergesort:

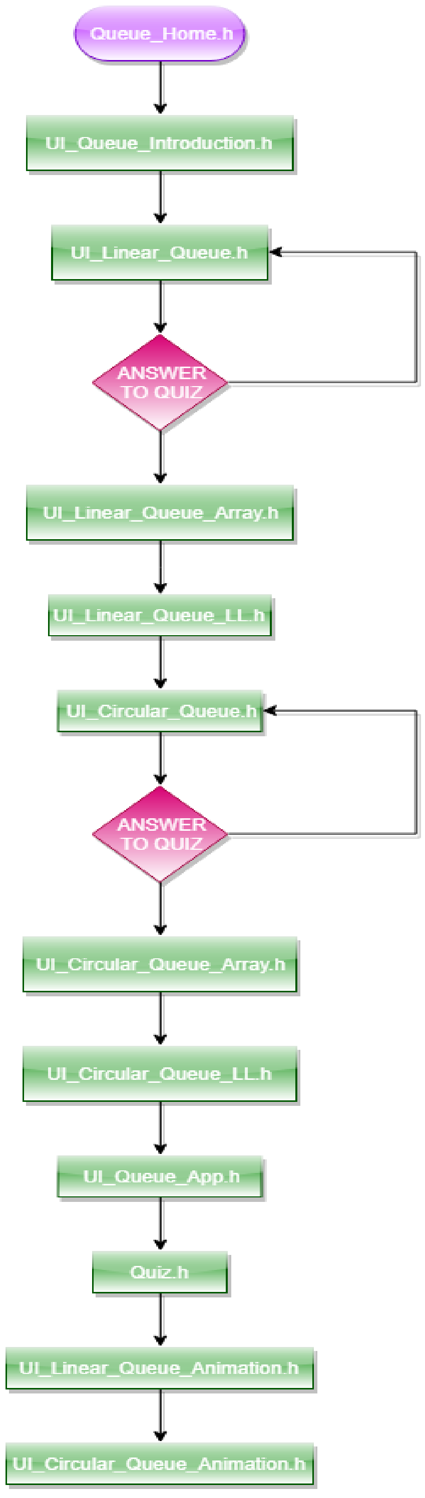
The interactive part of this form is the **panel1**, and has two timers **timer2** and **timer1**.
<their functions>
The user can enter values using **richTextBox1** and **button3**. The animation starts on pressing **button4**. **button5**, **button6** and **button7** respectively pause, resume the animation and reset the array.
The rest of the form is **quizPanel** which has the quiz question, picture-box explaining the code, and labels explaining the implementation.

6. SelectionSort:

The interactive part of this form is **panel1** and it has a timer **timer1**.
In the **panel1**, there are fifteen labels **l1** through **l15**, each with their respective arrows in labels **p1** through **p15**.

The user can enter values using **txt1** and **btnAdd**. The sorting starts with **btnSS**.

btnIncreaseSpeed and **btnDecreaseSpeed** increase and decrease the animation speed respectively, while **btnPause** and **btnReset** respectively pause the animation and reset the array.
The rest of the form contains **quizPanel** for the quiz question, a picture box showing the flowchart of selection sort algorithm, and various labels containing relevant information.



7. QUEUE

1. Queue_Home:

This is the backbone for the forms regarding Queue as it contains the buttons pointing to all the different modules in Queue.

2. UI_Circular_Queue:

This contains rich text boxes and picture boxes containing information regarding circular queues, and a **quizPanel** which has a quiz question.

3. UI_Circular_Queue_Animation:

Here the user is given two choices: array and linked lists, and depending on which of the two radio buttons **arraybtn** or **llbtn** the user selects, the user is shows either **arrayAnimationPanel** or **llAnimationPanel**. The user can enter the element to enqueue in the **enqtext** text box and click on **enqbtn** to enqueue, or click on **dqbtn** to dequeue.

4. UI_Circular_Queue_Array:

This contains a **richTextBox1** containing the implementation of circular queues using arrays.

5. UI_Circular_Queue_LL:

This contains a **richTextBox1** containing the implementation of circular queues using linked lists.

6. UI_Linear_Queue:

This contains information about linear queues and a **quizPanel** that contains a quiz question.

7. UI_Linear_Queue_Animation:

Here, the user is given two choices, array and linked lists, and depending on which of the two **arraybtn** or **llbtn** the user checks, the user is shown either **arrayAnimationPanel** or **llAnimationPanel**. The user can enter the element to enqueue in **enqtext** and enqueue using **enqbtn** or dequeue using **dqbtn**.

8. UI_Linear_Queue_Array:

Contains a **richTextBox1** which has the implementation of linear queues using arrays.

9. UI_Linear_Queue_LL:

Contains a **richTextBox1** which has the implementations of linear queues using linked lists.

10. UI_Queue_App:

Contains rich text boxes explaining applications of queues.

11. UI_Queue_Introduction:

Contains rich text boxes and picture boxes introducing queues.

---

8. STACKS

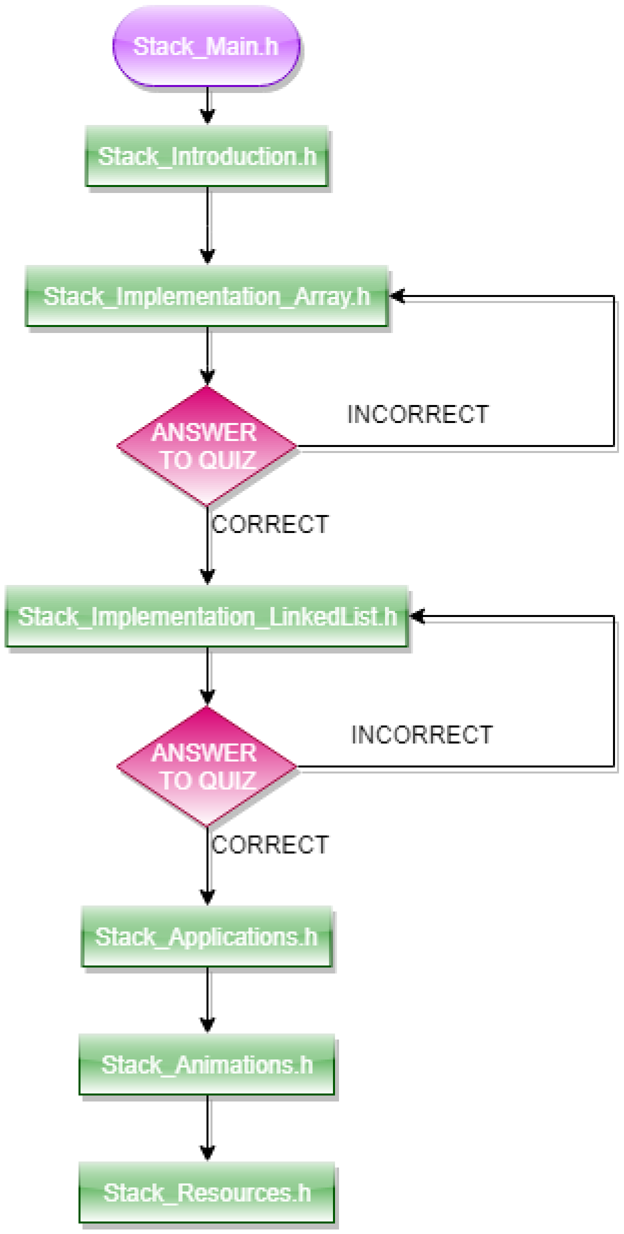
1. Stack_Animations:

Here, we are given two choices, either to implement using arrays or linked lists, and by checking either of the respective radio buttons **Stack_Array** or **radioButton1**, the user is shown either **panel1** or **panel2** respectively.

The user can then use **textBox1** to write a value, **button1** to push, **button2** to pop, and **button3** to clear the stack.

2. Stack_Applications:

Contains a rich text box explaining applications of stacks.



3. Stack_Implementation_Array:

This contains a rich text box explaining how arrays can be used to implement stacks.

It also contains a **quizPanel** containing a quiz question.

4. Stack_Implementation_LL:

This contains a couple of rich text boxes explaining how linked lists are used to implement stacks.

It also contains a **quizPanel** which has a quiz question.

5. Stack_Introduction:

Contains a **richTextBox1** introducing stacks.

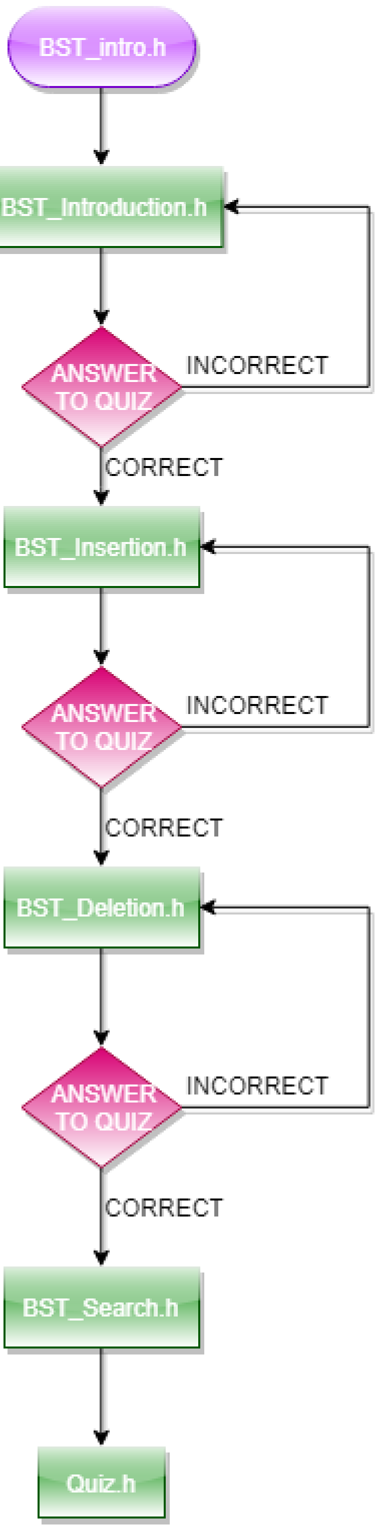
6. Stack_Main:

This is the backbone for all module regarding Stacks.

7. Stack_Resources:

This contains some links for learning stacks.

8. TREES



1. BST_Intro:

This is the backbone for all the modules for Trees, it contains buttons linking to all the modules.

2. BST_Introduction:

This form contains information about Trees in labels and picture-boxes, and a **quizPanel** for quiz question.

3. BST_Insertion:

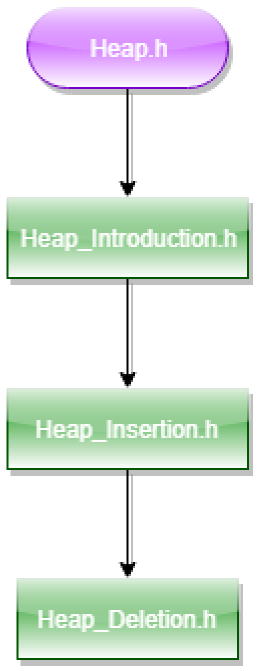
In this form, the user can insert the value in **BTinsert_value** and **InsertBT**, the tree is represented by labels from **BT1** through **BT7**.It contains a **quizPanel** for a quiz question. The animations are according to **BTInsertTimer**.

4. BST_Deletion:

This form contains a panel **delete_animation1**, you can choose either of **RB1**, **RB2** or **RB3** from a group box **groupBox1**, **BTree_delete_but** and **BTree_stop_but** respectively start and stop the animation. This form contains a **quizPanel** for quiz questions.

5. BST_Search:

This form shows different form of traversals, such as inorder, preorder and postorder. **btn_in**, **btn_pre**, **btn_post** and **btn_res** respectively give inorder traversal, preorder traversal, postorder traversal and reset the tree respectively. **bl1** through **bl7** are the labels that show the transversed elements. It also contains the **quizPanel**.



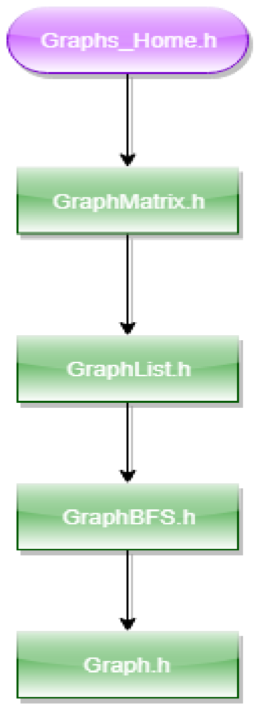
10. HEAP

1. Heap:
This is the backbone for all the modules for heaps.

2. Heap_Introduction:
This contains the labels and picture boxes that contain the relevant information.

3. Heap_Insertion:
In this form, the user can insert the value using **richTextBox1** and **button3**, while **button5**, **button6** and **button7** respectively pause, resume and reset the heap.

4. Heap_Deletion:
In this form, user can add elements using **richTextBox1** and **button3**, while delete using **richTextBox2** and **button1**, while **button5**, **button6** and **button7** respectively pause, resume and reset.



11. GRAPHS

1. Graphs_Home:
This form contains the backbone for all modules for Graphs.

2. GraphMatrix:
The user can enter the **txtFrom** and **txtTo** along with **btnAdd** to add, **btnReset** to reset.

3. GraphList:
The user can enter the **txtFrom** and **txtTo** along with **btnAdd** to add, **btnReset** to reset.

4. GraphBFS:

The user can enter the **txtFrom** and **txtTo** along with **btnAdd** to add, **btnBFS** to animate, **btnPause** to pause, **btnDecrease** and **btnIncrease** to decrease and increase animation speed, while **button1** to reset.

5. Graph:

The user can enter the **txtFrom** and **txtTo** along with **btnAdd** to add, **btnDFS** to animate, **btnPause** to pause, **btnDecrease** and **btnIncrease** to decrease and increase animation speed, while **button1** to reset.

---

12. DISCUSSION FORUM

This has the panel called **message_detail.** Below this panel are the panels **msgpanel** and **toppanel** which contains checkboxes that act as a way to filter different doubts according to the data structure they are regarding.

The **replypanel** contains **replytxt**, a rich text box, a **replybtn** which reads “Post” and a combobox called **tagcombo**.

13.CONTRIBUTION

1. Lavish Gulati
    a. Linear Search
    b. Binary Search
    c. Quizzes
    d. Integration and GUI

2. Arpit Gupta
    a. Stacks
    b. Discussion Forum
    c. Integration
    d. GUI of Graphs , BST , Heaps

3. Shivang Dalal
    a. Queues
    b. Discussion Forum
    c. Integrating quizzes in each module

4. Siddharth Agrawal
    a. Login, Sign up form
    b. Home page
    c. Mentor Panel
    d. Admin Panel

5. Devaishi Tiwari
    a. Added content in Stack
    b. Admin Documentation
    c. User Documentation

6. Ayush Patni
    a. Added content in Queue

7. Vivek Kumar
    a. Insertion Sort, Bubble sort, Selection Sort
    b. Graph representations of two types list and matrix
    c. Graph BFS and DFS

8. Rakesh Gupta
    a. Merge Sort, Quick Sort
    b. Heaps (Insertion and Deletion)
    c. Linear search in linked list

9. Abhishek Jaiswal
    a. Completed arrays, linked lists pages
    b. Completed Trees traversal
    c. Debugging

10. Ravi Shankar
    a. Animation of insertion in arrays
    b. Linked List insertion and deletion
    c. Completed Trees Insertion
    d. Debugging

11. Sanchit
    a. Animation of search in arrays
    b. Linked list and trees traversal
    c. Completed Trees Deletion
    d. Debugging

12. Ajinkya
    a. Technical Documentation

13. Sayal
    a. Technical Documentation

14. Naveen
    a.Technical Documentation

15. Sachin Giri
    a. Rigorous testing of all modules
    b. Debugging in all modules