There is some knowledge that related to basic binary tree function. And we’ll used these words for basic function in binary tree.

**add tree node function:** Get new node and root note of tree then add new node to binary treewhich compare with E-mail. If there are same E-mail, it should print error.

**search tree function:** Find node that want to know in binary treewhich find with E-mail. If found data, return address of user back. If not, return NULL.

**delete tree node function:** Get node that want to delete and root node of tree then find node that want to delete in tree and deleted it. If can delete, return 1, If can’t, return 0.

**Add New User Data**

Ask data from user

Send E-mail to hash function

Get position of hash function

If there isn’t any data

Set that position in hash table to new data

End if

Else if there is some data

Get root node tree of that position in hash table

Send new data and root node tree to **add tree node** **function**

End if

**Find User Data**

Set found user to NULL

Get E-mail from user

Send E-mail to hash function

Get position of hash function

If that position of hash table is NULL

Print not found message

Return found user

End if

Else

Get root node tree of that position in hash table

Send E-mail and root node tree to **search tree function**

Set found user to receive address of user after send to search

If found user equal NULL

Print not found message

Return found user

End if

Else if found user not equal NULL

Print data profile

Return found user

End if

End if

**Add Friend**

Set current user to user that login

\*After find another user\*

Set another user to user that we found

Set found user to NULL

Set friend relation to 0

Send root node of friend of current user and E-mail of another user to **search tree function**

Set found user to receive address of user after send to search

If found user is NULL

Set friend relation to 0

Else if found user is not NULL

Set friend relation to 1

Get command from user

If user want to be friend

If friend relation equal to 0

Create new linked list of pending requests

Set user in new linked list to current user

Set next node of new linked list to head node of friend pending request of found user

Set head node of friend pending request of found user to new linked list

End if

Else if friend relation equal to 1

Print error message

End if

End if

**Friend Pending Request**

Set current user to user that login

Set temporary request to NULL

Get Command from user

If user want to see friend pending request

While friend pending request not equal to NULL

Print head node of friend pending request

Ask command for user

If accept friend

Send head node of friend pending request and root node of friend to **add tree node** **function**

Send current user and root node of friend of head node of pending request to **add tree node** **function**

Set head node of pending request to next node of head node

End if

If deny friend

Set temporary request to next list of head node

Free linked list of head node

Set head node of friend pending request to temporary friend

End if

**Delete Friend**

Set current user to user that login

\*After find another user\*

Set another user to user that we found

Set found user to NULL

Set friend relation to 0

Send root node of friend of current user and E-mail of another user to **search tree function**

Set found user to receive address of user after send to search

If found user is NULL

Set friend relation to 0

Else if found user is not NULL

Set friend relation to 1

Get command from user

If user want to delete friend

If friend relation equal to 1

Send found user and root node of friend of current user to **delete tree node function.**

Send current user and root node of friend of found user to **delete tree node function.**

If value that return after send to delete equal to 0

Print error message.

End if

End if

Else if friend relation equal to 0

Print error message

End if

End if