

SPL-1 Project Report, [2019]

STUDENT INFORMATION MANAGEMENT SYSTEM

Course no. :

Course name:

Submitted By:

Name: Sudip Kumar Sah

BSSE Roll no. : 1040

BSSE Session:2019

Supervised By:

Md. Nurul Ahad Towhid

Assistant Professor

Institute of Information Technology



Date:29.May.2019

Tables of Content

1. Introduction.....	1
1.1 Background Study.....	2-6
1.1.1 B-tree	
1.1.2 Text color	
1.2Challenges.....	7
2. Project Overview.....	8-11
2.1 For insertion	
2.2 For deletion	
2.3 For searching	
3. User Manual.....	12-14
4. Conclusion.....	15
5.References.....	15

1.Introduction:

The project named “Student Information Management System” clarifies its objectives and importance. It will keep the records of students enrolled in IIT with basic (personal and academic) information. Student information systems provide capabilities for registering students in courses. Student Information System is a management information system for education establishments to manage student data. Keeping of records had started from a very long time but people used to keep manually i.e. hand-written. They used to keep records in hardcopy. They were updating records manually. A lot of background systematic study has to be maintained while updating. A bulky and heavy register used to carry while working with related information. Searching for specific person related information, sometimes they may have kept record with lexicographically or randomly. With lexicographically, it has become a little simpler but although it used to take a lot of time to retrieve information. Previously, register need to leave some spaces to update information at a instant time and also used to occupy a large space. We used to have bundles of register at working places which seems bulky and hard to maintain.

To overcome this problem, different online or offline database for keeping record has introduced with user manual. They provide information how to work with it. The project performs same work but the way this project works is quite different. The project opens with some recorded information and you can perform search and delete operations as well as when user wants to keep new records which will be updated to file for future reference at a time of need. It is very easy to use and work can performed with a little information. It automatically provides what to do next.

1.1 Background Study

1.1.1B-tree:

It is primarily used by relational databases or file systems. It provides an efficient way to implement common database features. It is ideal to work with large blocks of data. B-tree is a tree data structure that keeps data sorted and allows searches, sequential access and insertions in logarithmic time. The B-tree is a generalization of a binary search tree in that a node can have more than two children. B-tree is a multi-way search tree. A node in B-tree of order n can have at most $n-1$ values and n children.

All values that appear on the left sub-tree are smaller than left most value in the parent node. All values that appear on the right sub-tree are greater than right most value in the parent node. All values that appear on the middle sub-tree are greater than leftmost value in parent node and smaller than right most value in parent node.

Examples of B-tree

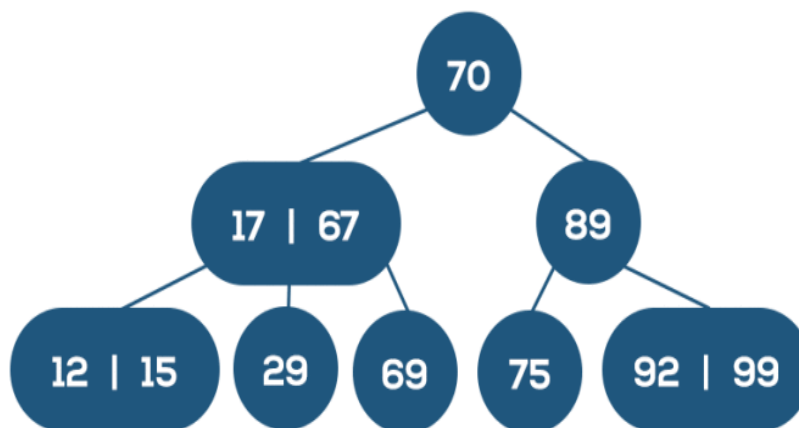


Figure-1: Examples of B-tree

Three operations performed by B-tree:

1. Insertion
2. Deletion
3. Searching

Insertion

Insert the value 17 to the above B-Tree.

Find the appropriate position to insert the given value in B-Tree.

$17 < 70$ – Search in left sub-tree of 70.

$17 \leq 17$ – Search in left sub-tree of 17.

17 needs to be inserted in the left child of 17.

The tree will results into

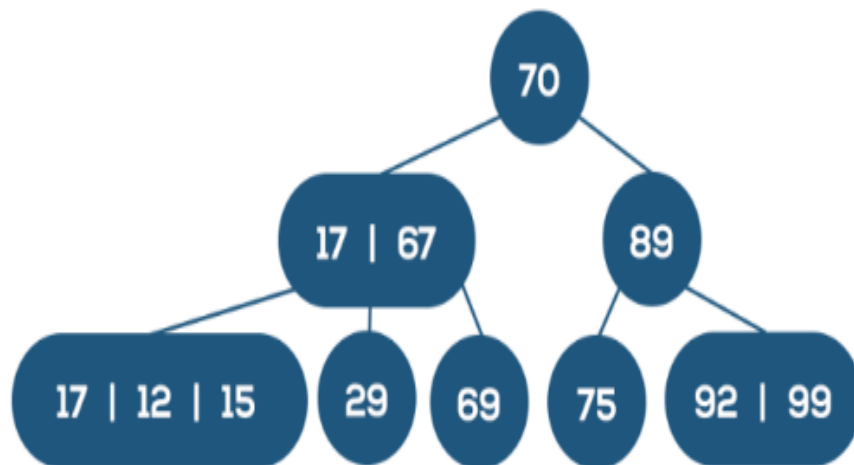


Figure-2(a): Inserting node (17) of B-tree

Now, the modified node has 3 values 17, 12 and 15. But, it violates the rule in B-Tree (any node in B-Tree of order can have at most $n-1$ value).

To restore B-Tree, middle value of 17, 12 and 15 is moved to parent node. Then, split the resultant node containing 17 and 15 into two nodes forming left and right sub-tree containing the value 17 and 15 correspondingly.

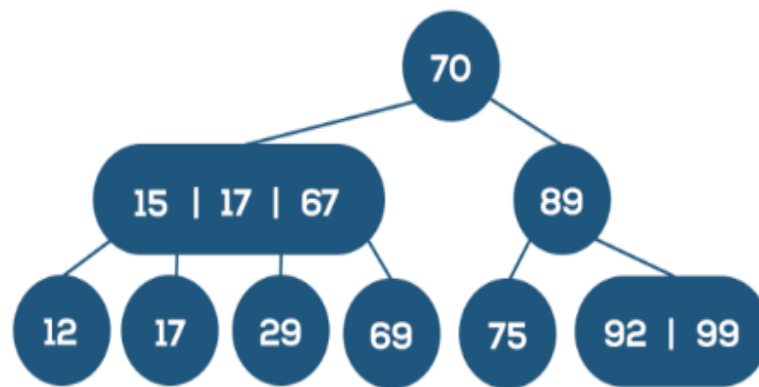


Figure-2(b): modified form (1)-Inserting node (17) of B-tree

Now, the parent node violates B-Tree definition. So, restore it.

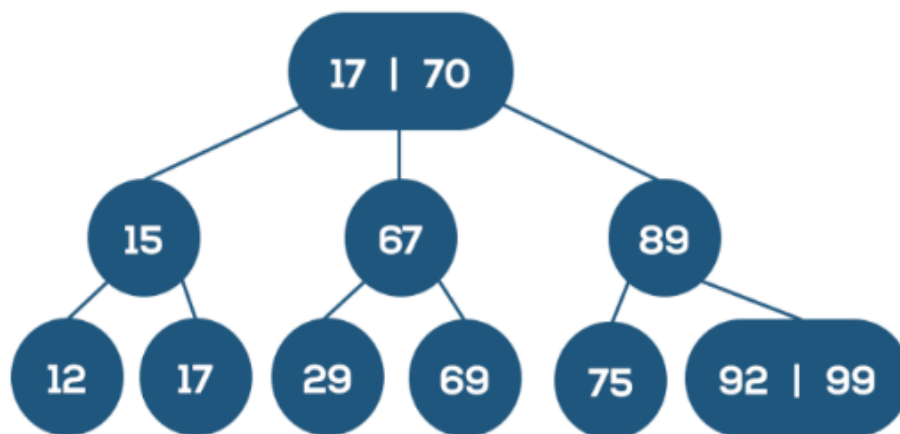


Figure-2(c): modified form (2) -Inserting node (17) of B-tree

Deletion

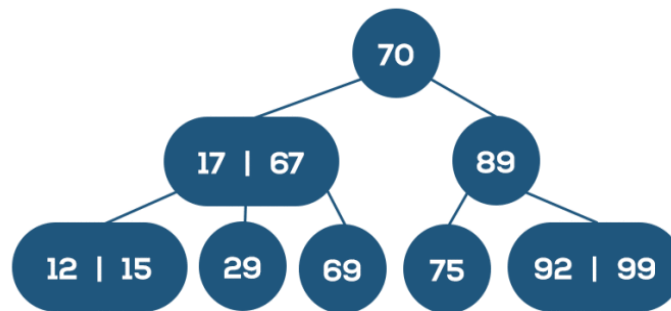


Figure-3(a): Deleting node (69) of B-tree

Delete 69 from the above B-Tree. Search the position of 69 to delete.

69 17 – Compare 69 with right most value in the search node.

69 > 67 – Search in right sub-tree of 67

69 is the right child of 67.

In case the node which we are trying to delete has only one value(69), then find the predecessor(29) for it(69) and merge the predecessor with the sibling(29) of the node to be deleted. Then, delete the desired node.

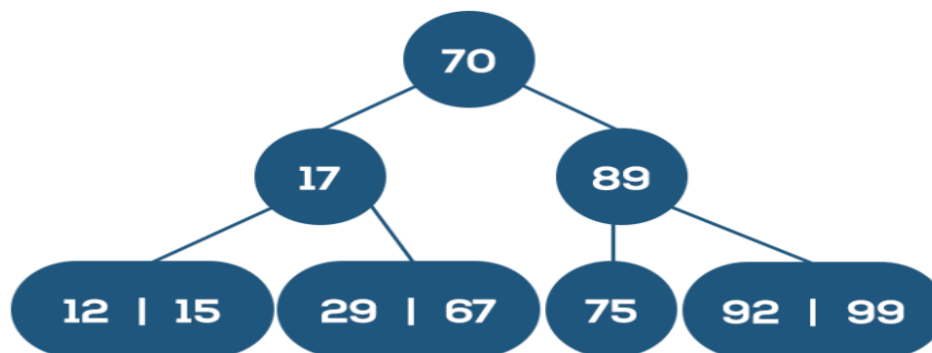


Figure-3(b): Deleting node (69) of B-tree

Searching

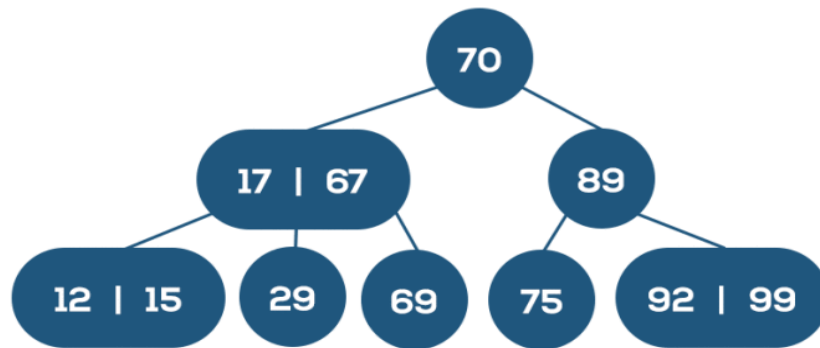


Figure-4: Searching node (29) of B-tree

Search the value 29 in above B-Tree.

29 17 && 29 < 67 – So, search in middle sub-tree.

29 is the middle child of 17 & 67.

1.1.2 Text Color

Color attributes are specified by TWO hex digits -- the first corresponds to the background; the second the foreground. Each digit can be any of the following values:

0 = Black	8 = Gray
1 = Blue	9 = Light Blue
2 = Green	A = Light Green
3 = Aqua	B = Light Aqua
4 = Red	C = Light Red
5 = Purple	D = Light Purple
6 = Yellow	E = Light Yellow
7 = White	F = Bright White

1.2 Challenges:

Implementing a new software solution carries with it a number of challenges. The process can be overwhelming, confusing and lengthy. The project has to handle huge amount of data and b-tree algorithm for insertion, deletion and searching results quite challenging with this data.

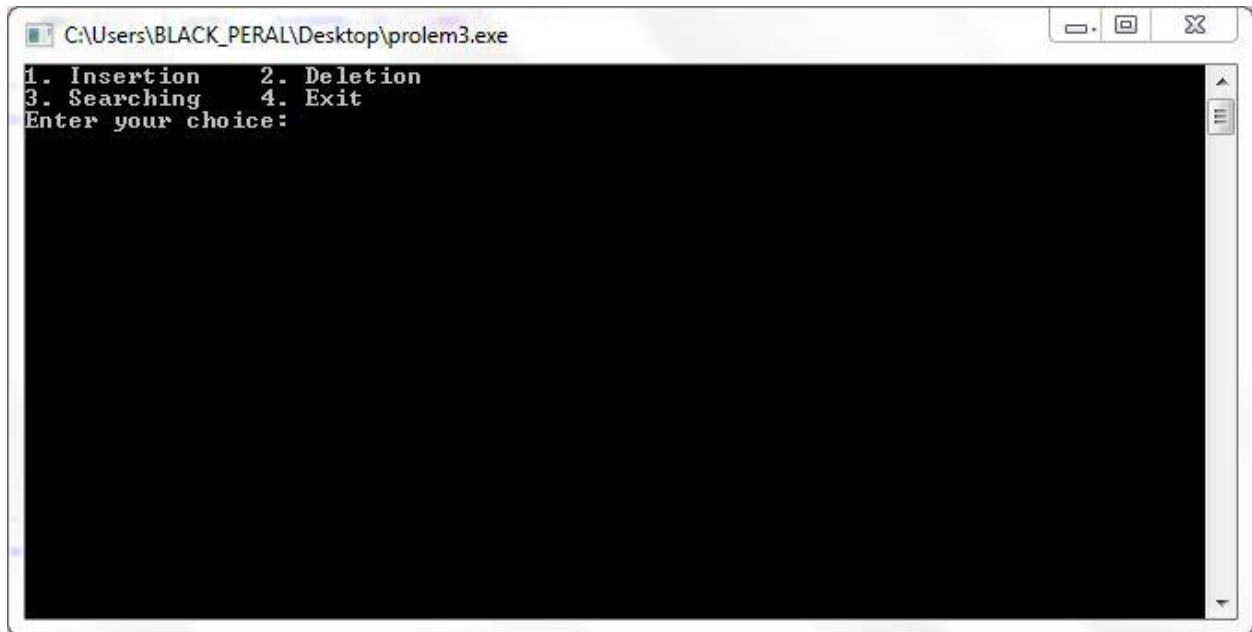
Challenges are enlisted below:

- **Inputting a huge information from console**
- **Handling those information with the help of inserting algorithm within readable text file**
- **Parsing those information from file when user perform search with particular student roll number**
- **Deleting roll number associated data from file**
- **Implementing algorithm for handling node as roll number in left and right in tree**

2. Project Overview

The project has mostly 3 parts included; the project takes input as roll number of students in the beginning.

When we run the project, the below windows will appear and ask for what kind of operation want to perform.



When we choose option (1), it asks for your roll number as well as it takes entered roll number as one of its node of tree. After entering roll number, it will ask for roll number related data from user. Now this data given by user will be saved to text file.

2.1 For insertion:

- 1) Struct type structure is defined for creating node for tree and when user enters their roll number then it takes all roll number into an array (val);
- 2) check whether val at index 0 is less or greater or equal to the val at index 1

```

/* sets the value val in the node */
int setValueInNode(int val, int *parentVal, btreeNode *node, btreeNode **child) {

    int pos;
    if (!node) {
        *parentVal = val;
        *child = NULL;
        return 1;
    }

    if (val < node->val[1]) {
        pos = 0;
    }
    else {
        for (pos = node->count;
            (val < node->val[pos] && pos > 1); pos--);
        if (val == node->val[pos]) {
            cout<<"Duplicates not allowed\n";
            return 0;
        }
    }
    if (setValueInNode(val, parentVal, node->link[pos], child)) {
        if (node->count < MAX) {
            addValToNode(*parentVal, pos, node, *child);
        }
        else {
            splitNode(*parentVal, parentVal, pos, node, *child, child);
            return 1;
        }
    }
}

```

3) now addValToNode() functions, places their child node at appropriate places

```

void addValToNode(int val, int pos, btreeNode *node, btreeNode *child) {
    int j = node->count;
    while (j > pos) {
        node->val[j + 1] = node->val[j];
        node->link[j + 1] = node->link[j];
        j--;
    }
    node->val[j + 1] = val;
    node->link[j + 1] = child;
    node->count++;
}

```

4) if tree has more than four nodes then it splits using function splitNode();

2.2 For deletion:

- 1) When user makes choice for deletion of student's data, he/she must has to enter respective roll number to the program on which it sent to function called deletion();

```
void deletion(int val, btreeNode *myNode) {
    btreeNode *tmp;
    if (!delValFromNode(val, myNode)) {
        cout<<"Entered value related data is not present in B-Tree\n";
        return;
    }
    else {
        if (myNode->count == 0) {
            tmp = myNode;
            myNode = myNode->link[0];
            cout<<"roll number with data is found to deleted";
        }
    }
}
```

- 2) After deleting, copy successor for the value to be deleted.

```
void copySuccessor(btreeNode *myNode, int pos) {
    btreeNode *dummy;
    dummy = myNode->link[pos];

    for (; dummy->link[0] != NULL;)
        dummy = dummy->link[0];
    myNode->val[pos] = dummy->val[0];
}

/* removes the value from the given node and rear
void removeVal(btreeNode *myNode, int pos) {
    int i = pos + 1;
    while (i <= myNode->count) {
        myNode->val[i - 1] = myNode->val[i];
        myNode->link[i - 1] = myNode->link[i];
        i++;
    }
    myNode->count--;
}
```

- 3) Finally, adjusts all the nodes by doing right and left shift.
- 4) Then, finally all the data related user entered roll will be removed.

2.3 Searching:

It is performed using function called searching();

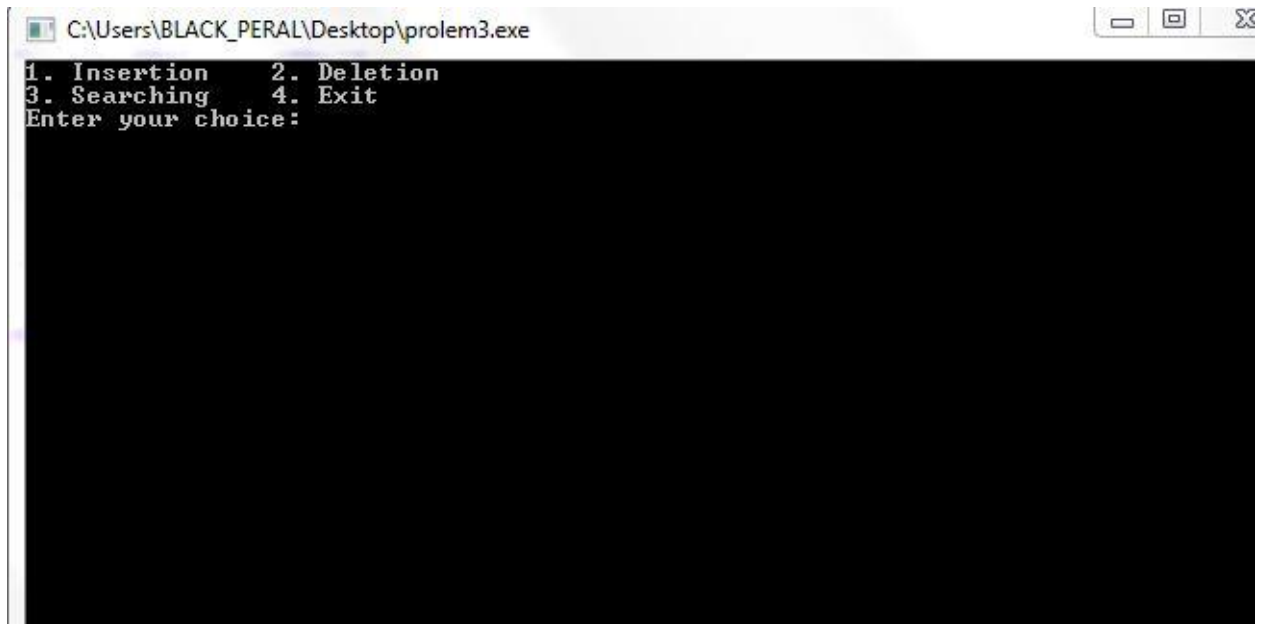
```
void searching(int val, int *pos, btreeNode *myNode) {
    if (!myNode) {
        return;
    }

    if (val < myNode->val[1]) {
        *pos = 0;

    }
    else {
        for (*pos = myNode->count;
            (val < myNode->val[*pos] && *pos > 1); (*pos)--);
        if (val == myNode->val[*pos]) {
            cout << "Given data is Found\n";
            return;
        }
    }
    searching(val, pos, myNode->link[*pos]);
    cout << "Given data is not Found\n";
    return;
}
```

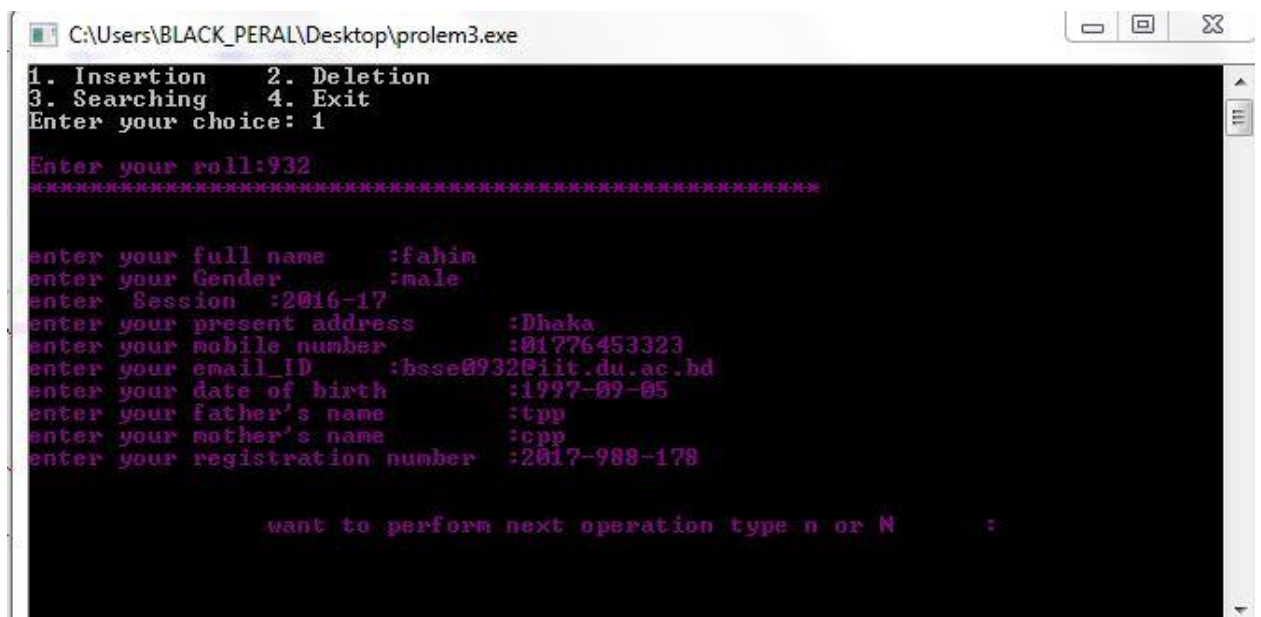
3. User manual

As the program begins, the windows open like



```
C:\Users\BLACK_PERAL\Desktop\prolem3.exe
1. Insertion    2. Deletion
3. Searching    4. Exit
Enter your choice:
```

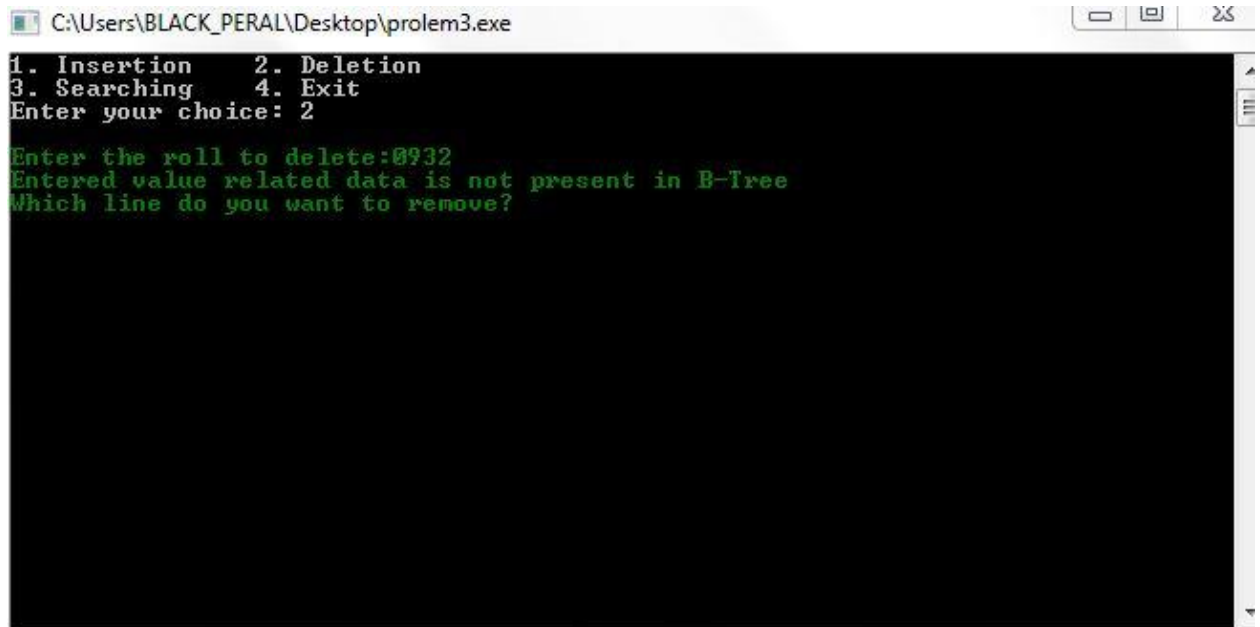
Then, when we enter for inserting data for tree and then it turns into purple. It helps user to identify easily.



```
C:\Users\BLACK_PERAL\Desktop\prolem3.exe
1. Insertion    2. Deletion
3. Searching    4. Exit
Enter your choice: 1
Enter your roll:932
*****
enter your full name      :fahim
enter your Gender        :male
enter Session            :2016-17
enter your present address :Dhaka
enter your mobile number  :01776453323
enter your email_ID       :bsse0932@iit.du.ac.bd
enter your date of birth  :1997-09-05
enter your father's name  :tpp
enter your mother's name  :cpp
enter your registration number :2017-988-178

want to perform next operation type n or N :
```

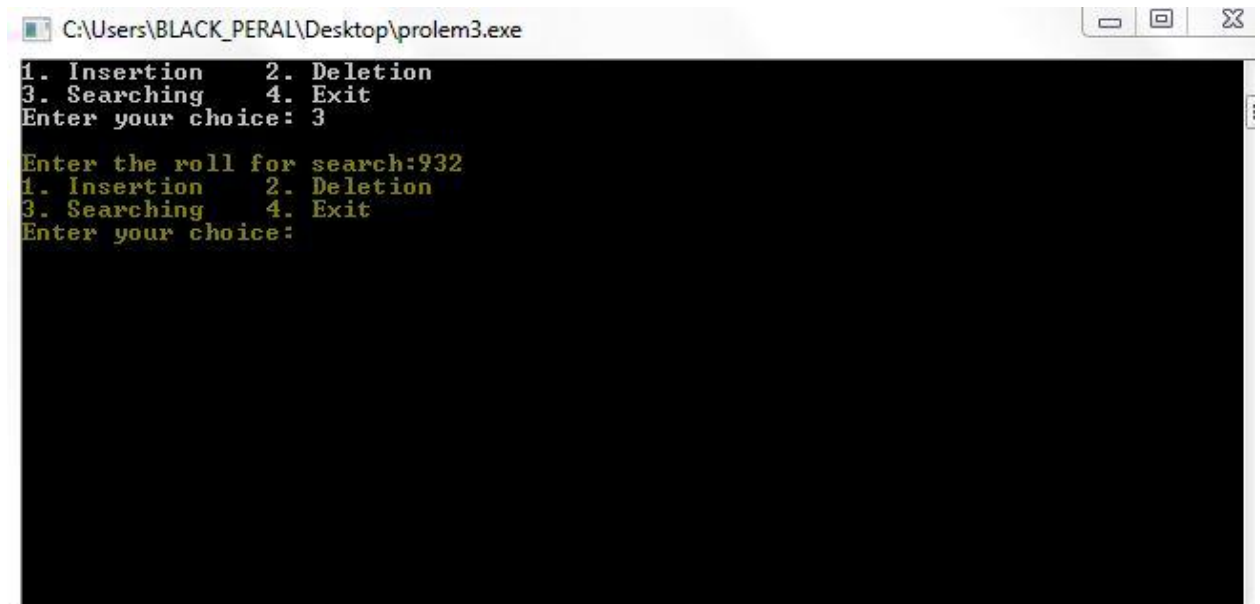
When we enter into delete option, it turns green



```
C:\Users\BLACK_PERAL\Desktop\prolem3.exe
1. Insertion    2. Deletion
3. Searching    4. Exit
Enter your choice: 2

Enter the roll to delete:0932
Entered value related data is not present in B-Tree
Which line do you want to remove?
```

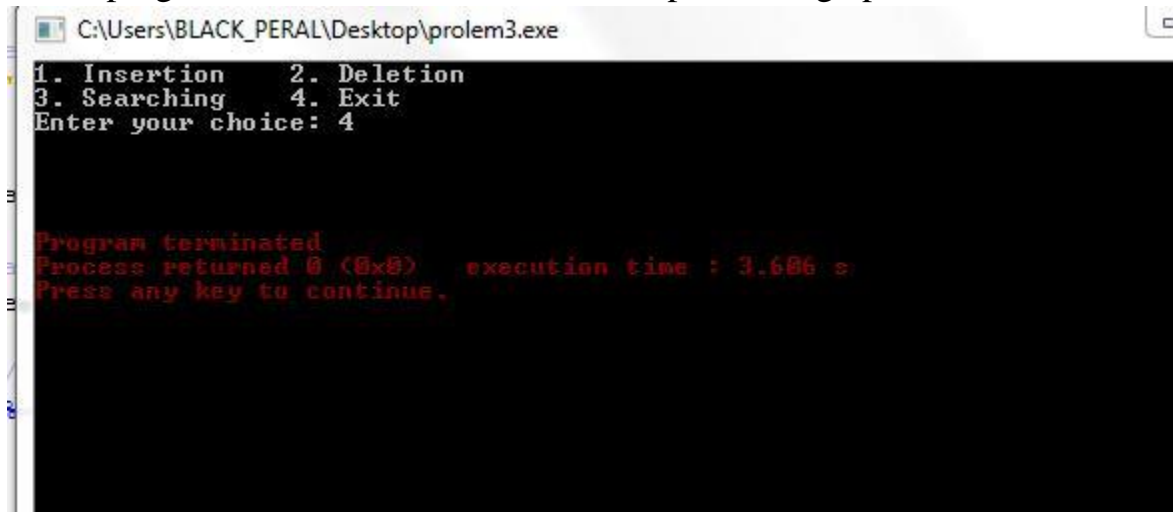
When we enter into search option, it turns yellow



```
C:\Users\BLACK_PERAL\Desktop\prolem3.exe
1. Insertion    2. Deletion
3. Searching    4. Exit
Enter your choice: 3

Enter the roll for search:932
1. Insertion    2. Deletion
3. Searching    4. Exit
Enter your choice:
```

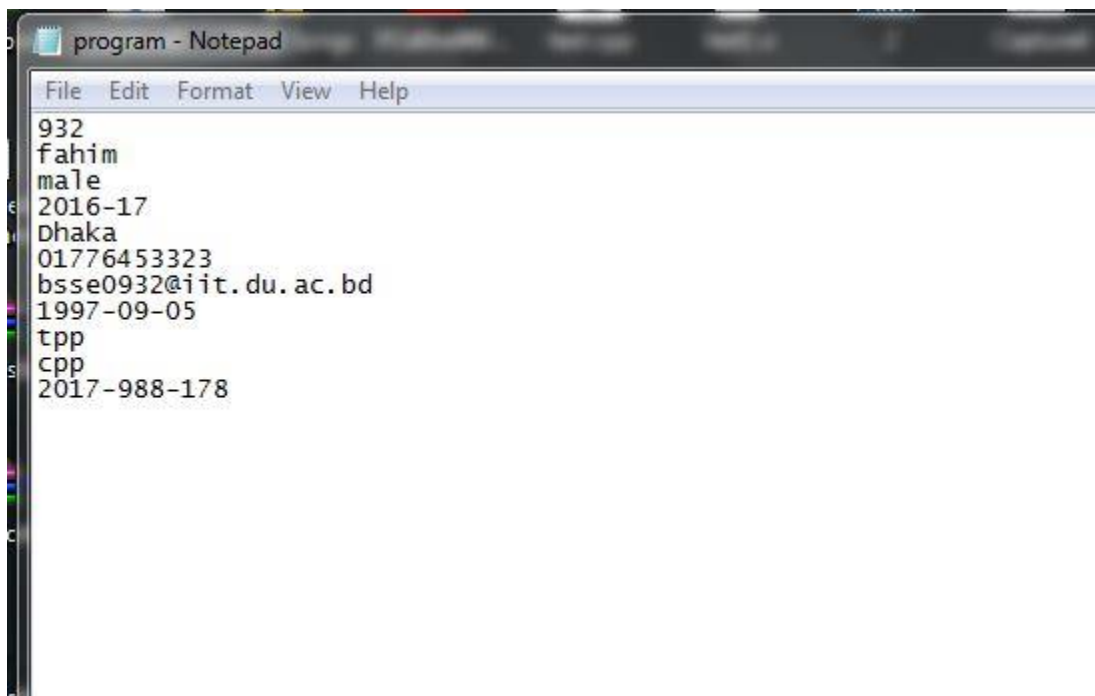

When program terminated, it turns red after performing operations;



```
C:\Users\BLACK_PERAL\Desktop\prolem3.exe
1. Insertion    2. Deletion
3. Searching    4. Exit
Enter your choice: 4

Program terminated
Process returned 0 (0x0)   execution time : 3.686 s
Press any key to continue.
```

Finally, the data written to text file named “program.txt” file.



```
program - Notepad
File Edit Format View Help
932
fahim
male
2016-17
Dhaka
01776453323
bsse0932@iit.du.ac.bd
1997-09-05
tpp
cpp
2017-988-178
```

4. Conclusion

Implementing B-tree algorithm for inserting, deleting and searching helps me to improve my coding skill and I have learned to handle large code for the first time. I hope it will help me to deal with difficulties in future. This project was quiet challenging and I gained a lot of experience from it. I want to thank my supervisor for guiding me a lot during this project.

5. References:

1. <http://gousios.org/courses/algo-ds/book/b-trees-b-trees.html>
2. <https://uxmankabir.wordpress.com/2017/05/08/cpp-program-to-perform-insertion-deletion-and-traversal-in-b-tree/>
3. <https://stackoverflow.com/questions/14707742/how-to-indent-my-code-in-codeblocks>
4. <https://webdocs.cs.ualberta.ca/~holte/T26/ins-b-tree.html>