

BLG 381E

Advanced Data Structure

2011 Summer School

Report of Homework 3

Date of Submission : 10.08.2011

Student Name : Ozan Arkan Can

Student Number : 040090573

Instructor : Arzucan Özgür Türkmen

Teaching assistant: Ahmet Aycan Atak

CRN: 30621

INDEX

INTRODUCTION	3
Development and Operating Environments	3
Ms Windows.....	3
Unix.....	3
UML Class Diagram.....	4
BUILDING B-TREE.....	4
Getting An Article	5
Splitting Article Into Sentences	5
Getting Words	7
Searching And Inserting.....	8
SEARCHING KEYS	8
CONCLUSION	11

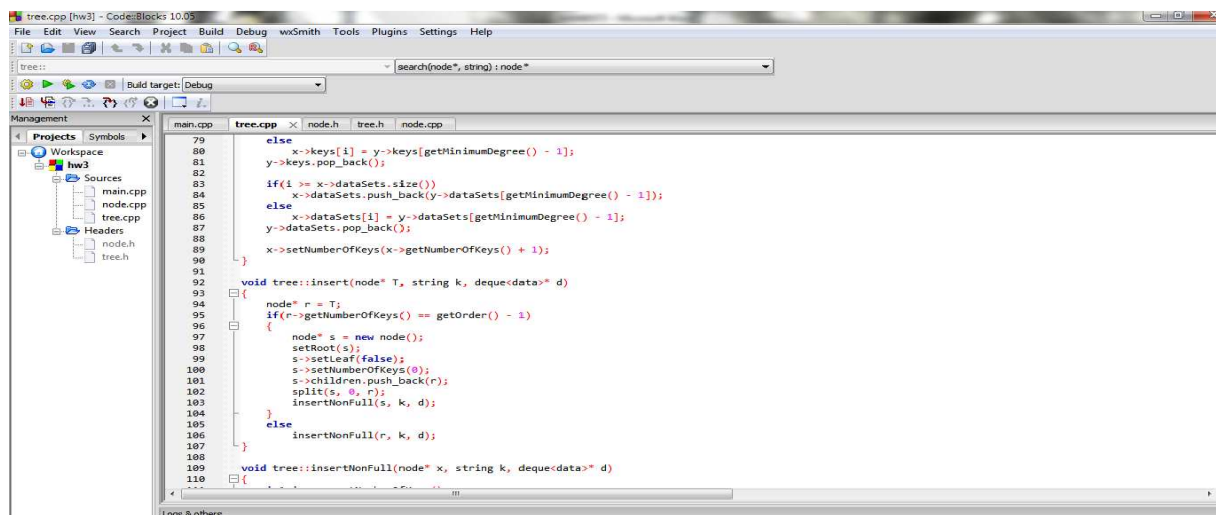
INTRODUCTION

In this homework, it is aim that showing difference between execution time with B-tree data structure with different order values (m) for searching a key. B-tree is build for m values which is obtained from command prompt. B-tree is built using Reuters dataset. Words that obtained from Reuters dataset, is used as keys for b-tree implementation. Node n has keys and dataset that has x value, word in which article and y value word in which sentence article x, for every key. After building b-tree is completed searching in b-tree n keys which is obtained from command prompt is executed. Finally, program outputs the total execution time for searching work.

Development and Operating Environments

Ms Windows

The Code::Blocks IDE has been used to write source code, compile and run the code.



Unix

The source code has been also copied to Unix, then compiled and tested with the GNU C++ Compiler. The following is the commands used:

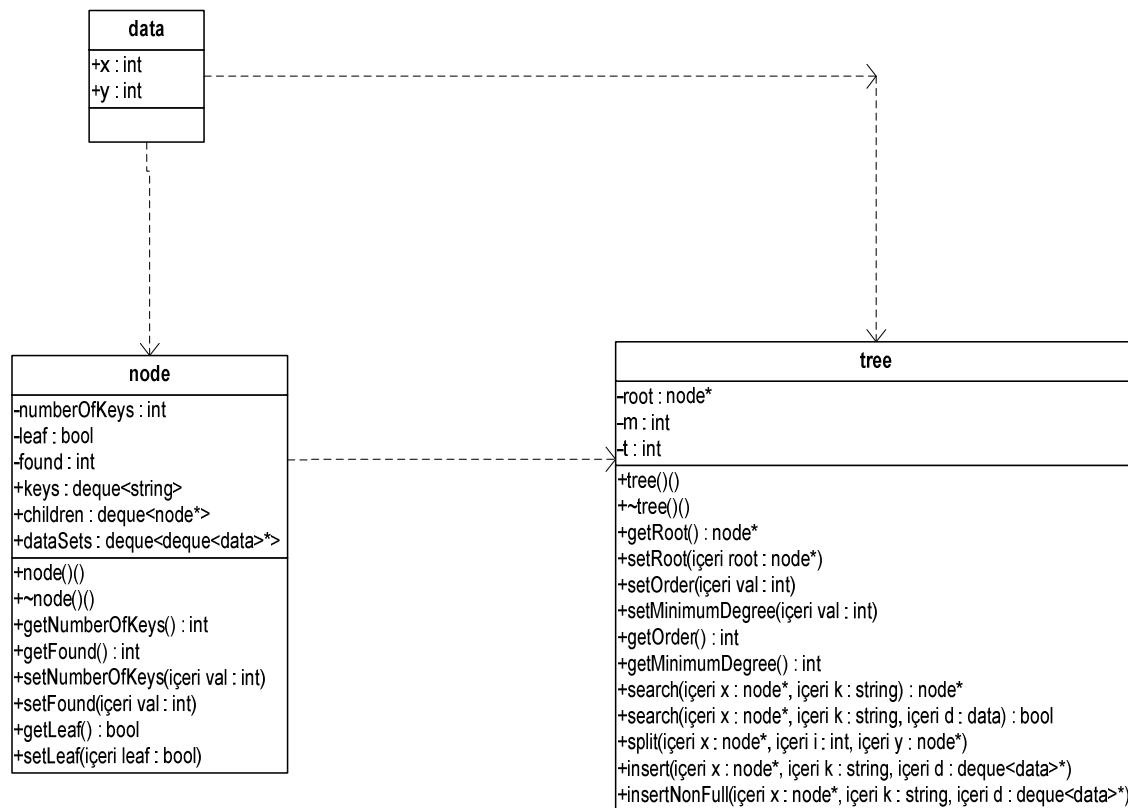
To compile :

```
g++ main.cpp tree.cpp tree.h node.cpp node.h -o hw3
```

To Run: ./hw3 4

UML Class Diagram

Program UML Diagram is like following chart.



BUILDING B-TREE

Building b-tree has following steps:

- Get an article from Reuters dataset
- Split this article into sentences
- Split sentence(that is obtained previous step) into words
- Search that word is it in tree

True: Add data in dataset

False: Add key in tree

Getting An Article

Reuters dataset contains article between <BODY> and </BODY> tags. getArticle function does this work. Also it moves pointer that points place in a file to forward.

```
string getArticle(ifstream& inputfile)
{
    string line, article("");
    bool check = false;
    int index, index2;
    while(!inputfile.eof())
    {
        getline (inputfile,line);
        index = line.find("<BODY>");
        index2 = line.find("</BODY>");
        if(check)
        {
            if(index2 != string::npos)
            {
                check = false;
                article.erase(article.size() - 1);
                return article;
            }
            else
                article += line + "\n";
        }
        else
        {
            if(index != string::npos)
            {
                check = true;
                article += line.substr(index+6);
                article += "\n";
            }
        }
    }
}
```

Splitting Article Into Sentences

In this work articles are represented as string. In this situation sentences are substring of article. Finding end of sentence is main idea for getting a sentence from article. Following situations are assumed as end of sentence:

-If a character is '!' or '?'

-If a character is '.'

 If character after that is not alphanumeric and ','

 If previous 2 character that is not '.'

-Sequential 4 characters are space

This method is enough for much sentence, but if sentence has table it causes getting wrong sentences.

getSentences function get a string and returns list of sentences.

Getting Words

Sentence splits into words from spaces. After obtaining a word following operations is applied on it.

- All characters are made lowercase

- All characters are eliminated except letters

In program, getWords function and cleanString function are used for these jobs.

```
deque<string> getWords(string& sentence)
{
    deque<string> wordList;
    int wordLength = 0;
    for(int i = 0; i < sentence.size(); i++)
    {
        if(isspace((int)sentence[i]))
        {
            if(wordLength != 0)
            {
                string str = sentence.substr(i - wordLength, wordLength);
                cleanString(str);
                if(str.length() != 0)
                    wordList.push_back(str);
            }
            wordLength = 0;
        }
        else
            wordLength++;
    }
    return wordList;
}

void cleanString(string& word)
{
    for(int i = 0; i < word.length(); i++)
    {
        if(!isalpha((int)word[i]))
        {
            string::iterator it = word.begin() + i;
            i--;
            word.erase(it);
        }
        else
            word[i] = tolower((int)word[i]);
    }
}
```

Searching And Inserting

For search and insert methods algorithms that are learned in lecture, are used.

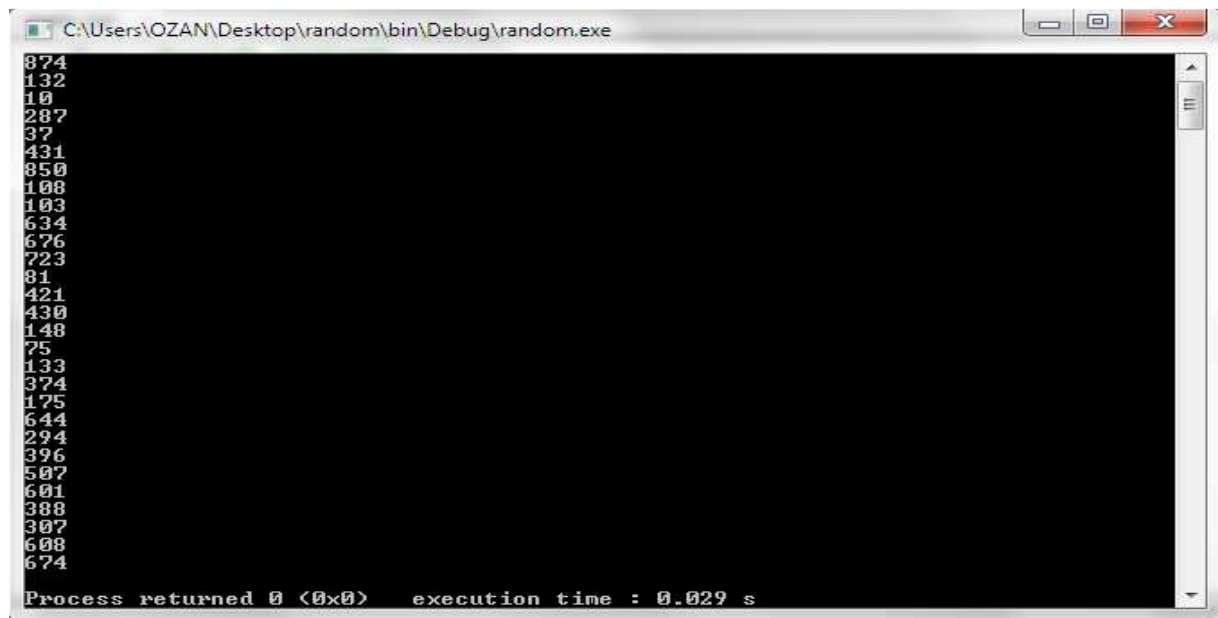
```
node* search(node*, string);  
bool search(node*, string, data);  
void split(node*, int, node*);  
void insert(node*, string, deque<data>*);  
void insertNonFull(node*, string, deque<data>*);
```

SEARCHING KEYS

After building b-tree with m order value, it is expected that searching n keys that are obtained from command prompt in tree. If key is in tree, program outputs its data, otherwise program should warn about this issue.

Keys that will be searched must be chosen randomly. Choosing keys randomly this method is followed.

Choosing a page from Oxford Thesaurus Dictionary and pick a word from that page.



```
C:\Users\OZAN\Desktop\random\bin\Debug\random.exe  
874  
132  
10  
287  
37  
431  
850  
108  
103  
634  
676  
723  
81  
421  
430  
148  
75  
133  
374  
175  
644  
294  
396  
507  
601  
388  
307  
608  
674  
Process returned 0 (0x0)   execution time : 0.029 s
```


When this method is applied following words are obtained.

“unfamiliar”, “coffee”, “excuse”, “expunge”, “appointed”, “individual”,
“transcend”, “car”, “bypass”, “possibility”, “rapidly”, “sample”, “board”,
“improve”, “indicate”, “configuration”, “blame”, “collapse”, “handicap”,
“crash”, “pretend”, “fake”, “holdings”, “luxury”, “passenger”, “hence”, “fight”,
“penalty”, “raise”

```
C:\Users\OZAN\Desktop>odev3.exe 4
B Tree is built

Program Searcher Part
Enter 29 string with a blank each two string
unfamiliar coffee excuse expunge appointed individual transcend car bypass possi
bility rapidly sample board improve indicate configuration blame collapse handic
ap crash pretend fake holdings luxury passenger hence fight penalty raise
```

```
'bypass' is not found in tree
'possibility' is found in tree
Datas:
Article: 74 Sentence: 11
Article: 214 Sentence: 1
Article: 287 Sentence: 17
Article: 508 Sentence: 2
Article: 533 Sentence: 1
Article: 681 Sentence: 1
Article: 792 Sentence: 10
Article: 792 Sentence: 29
Article: 874 Sentence: 3
Article: 896 Sentence: 10
Article: 896 Sentence: 29
```

```
'rapidly' is found in tree
Datas:
Article: 287 Sentence: 11
Article: 310 Sentence: 12
Article: 340 Sentence: 1
Article: 824 Sentence: 1
Article: 848 Sentence: 17
Article: 889 Sentence: 1
```

```
'sample' is found in tree
Datas:
Article: 595 Sentence: 2
```

```
'board' is found in tree
Datas:
Article: 6 Sentence: 1
Article: 6 Sentence: 8
Article: 9 Sentence: 1
Article: 9 Sentence: 2
Article: 23 Sentence: 1
Article: 28 Sentence: 3
```

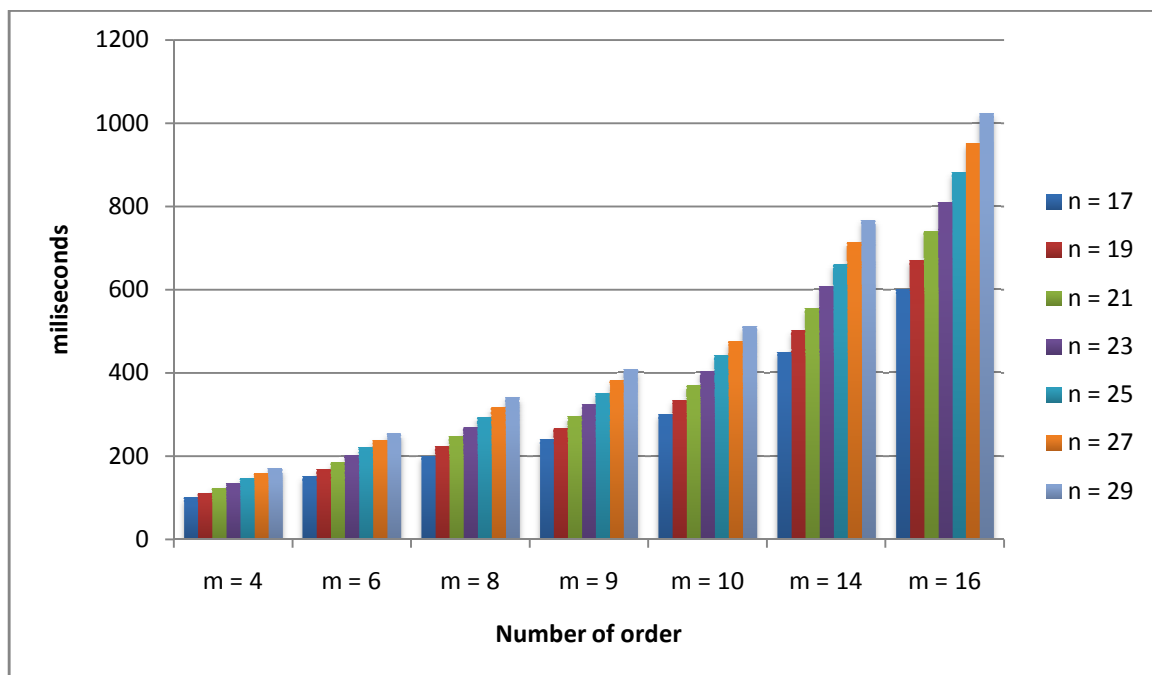
Program could not calculate the total time for searching n keys in b-tree, because its time is so little.

```
Running time for n = 17 : 0 milliseconds
Press a key to continue

'collapse' is found in tree
Datan:
Article: 558 Sentence: 9
Article: 613 Sentence: 6
Article: 694 Sentence: 1
Article: 694 Sentence: 2
Article: 694 Sentence: 13

'handicap' is not found in tree
Running time for n = 19 : 0 milliseconds
Press a key to continue
_
```

In this program nodes are not written in a file. And all works are done on memory. Searching a key is $O(\log n)$ for little m, because b-tree is more like binary tree. When m value is greater, execution time closes $O(n)$. Height of tree comes low and number of nodes is very small. This situation keys in these nodes with same layer. When execution time is 100 milliseconds for $m = 4$ and $n = 17$, following histogram can be drawn.



CONCLUSION

In this homework,

- properties of b-tree
 - implementing a b-tree with augmented features
 - Searching a key with different m values
 - Text processing
- are learned.