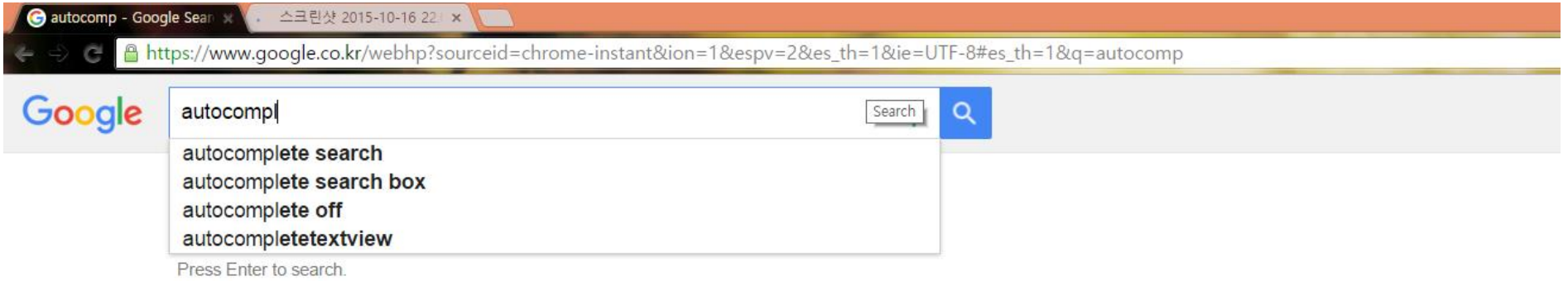
430.217
Introduction to Data Structures

Assignment 3. TRIE

Seoul National University
Advanced Computing Laboratory

ACL ADVANCED COMPUTING LABORATORY

실습과제 : Autocomplete 구현

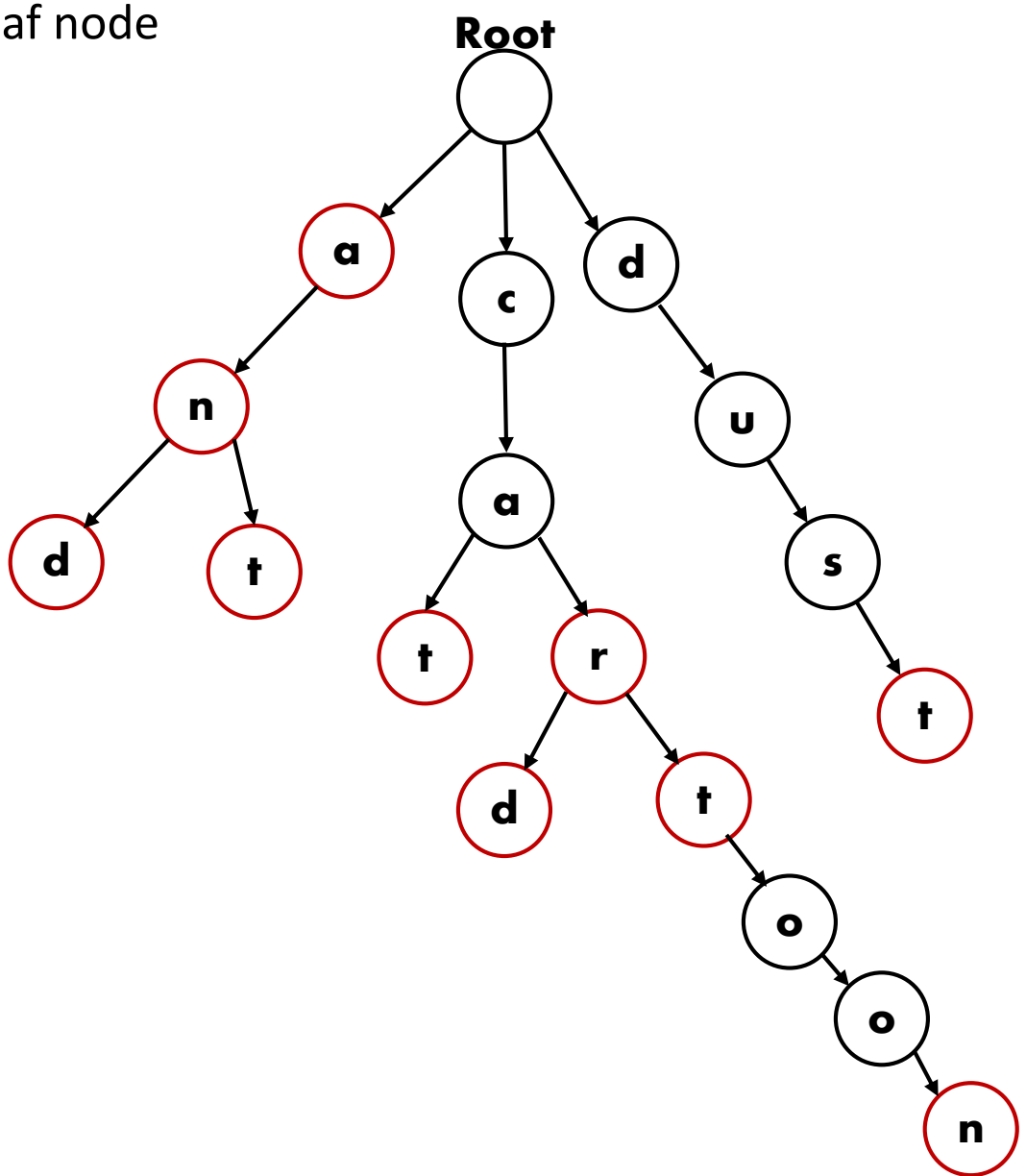


실습과제 : Autocomplete 구현

- Calling every child node to leaf node

Input

Suggested words



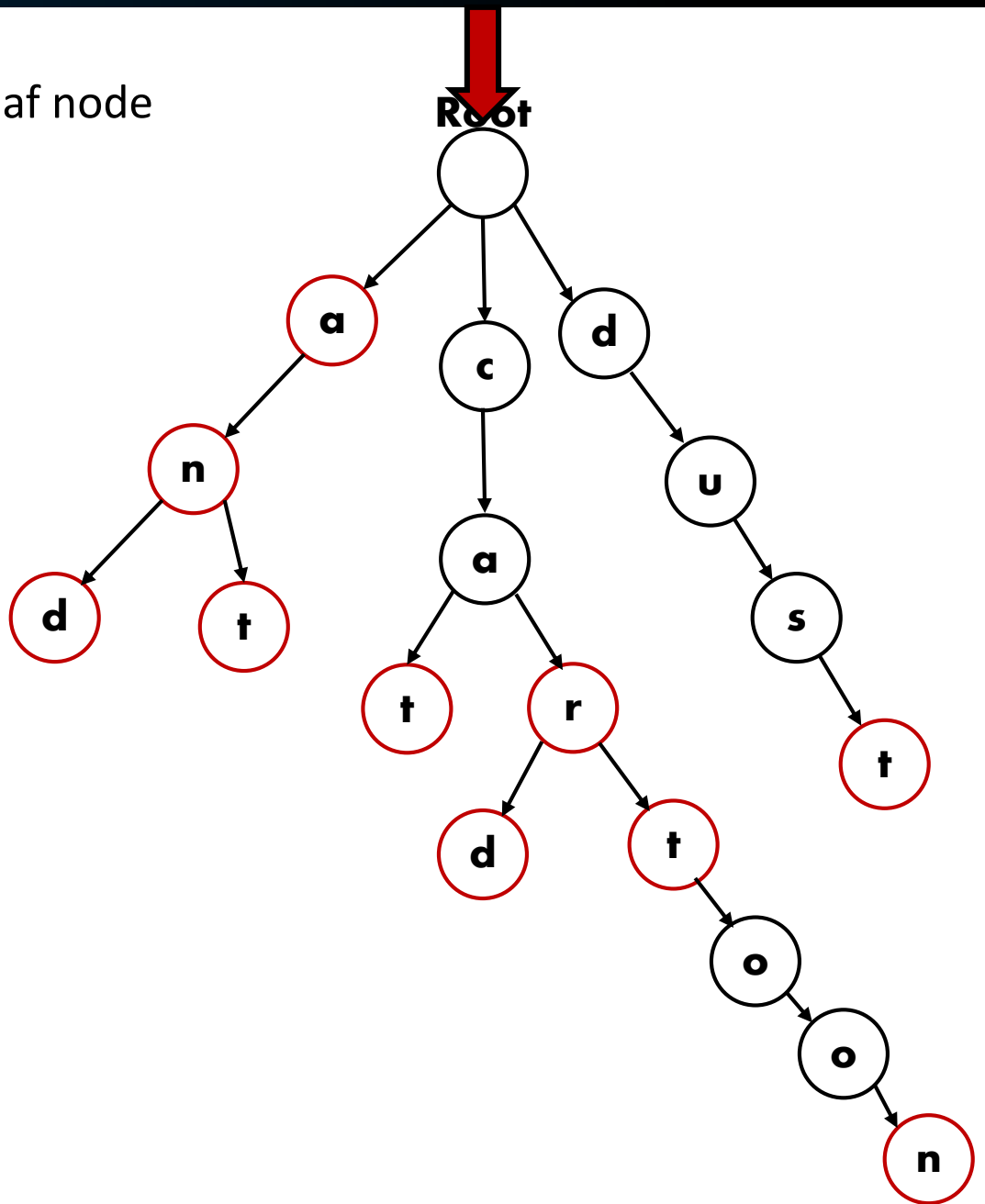
실습과제 : Autocomplete 구현

- Calling every child node to leaf node

Input

Suggested words

and
ant
cat
car
card
cart
cartoon
dust



실습과제 : Autocomplete 구현

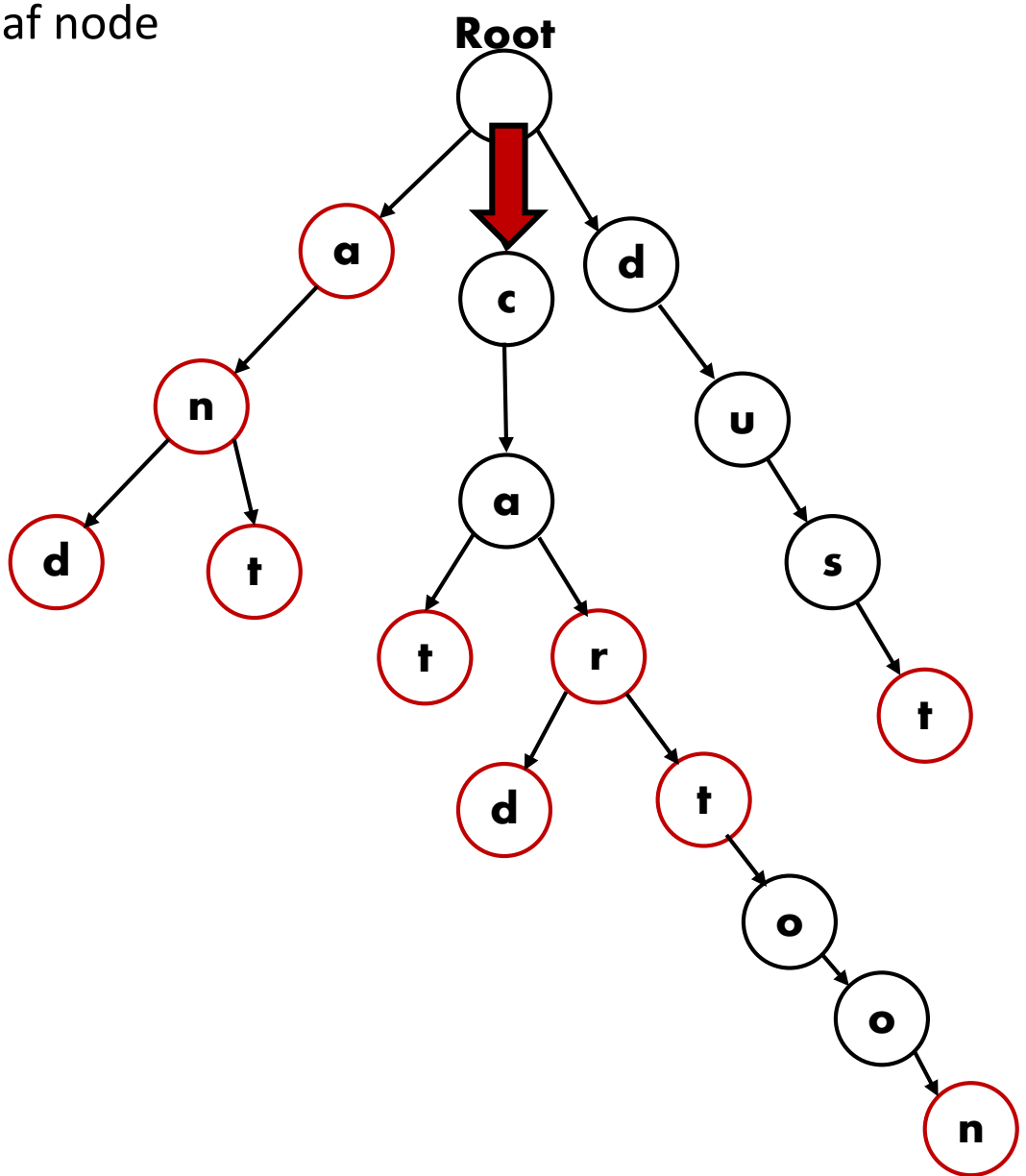
- Calling every child node to leaf node

Input

c

Suggested words

cat
car
card
cart
cartoon



실습과제 : Autocomplete 구현

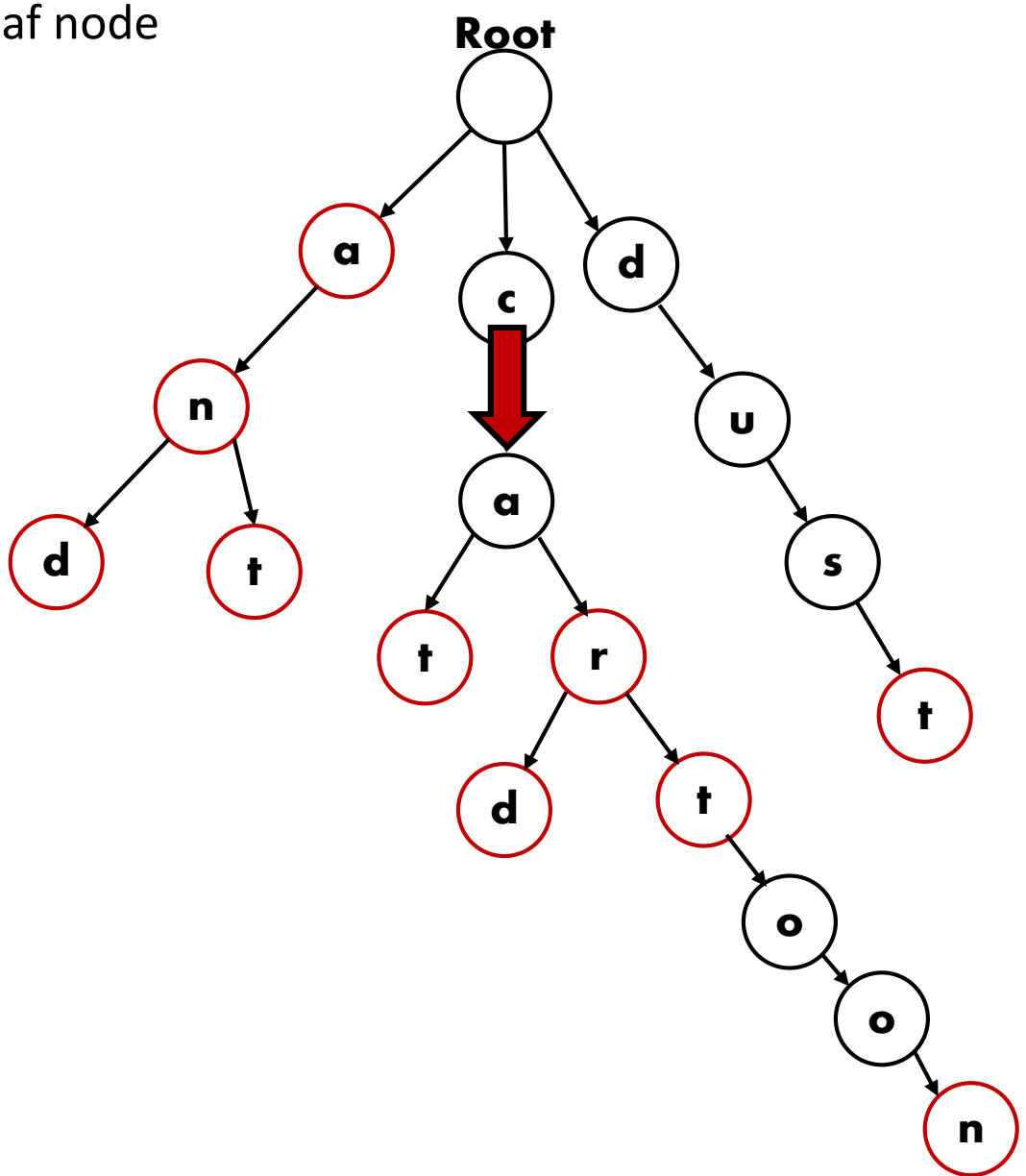
- Calling every child node to leaf node

Input

ca

Suggested words

cat
car
card
cart
cartoon



실습과제 : Autocomplete 구현

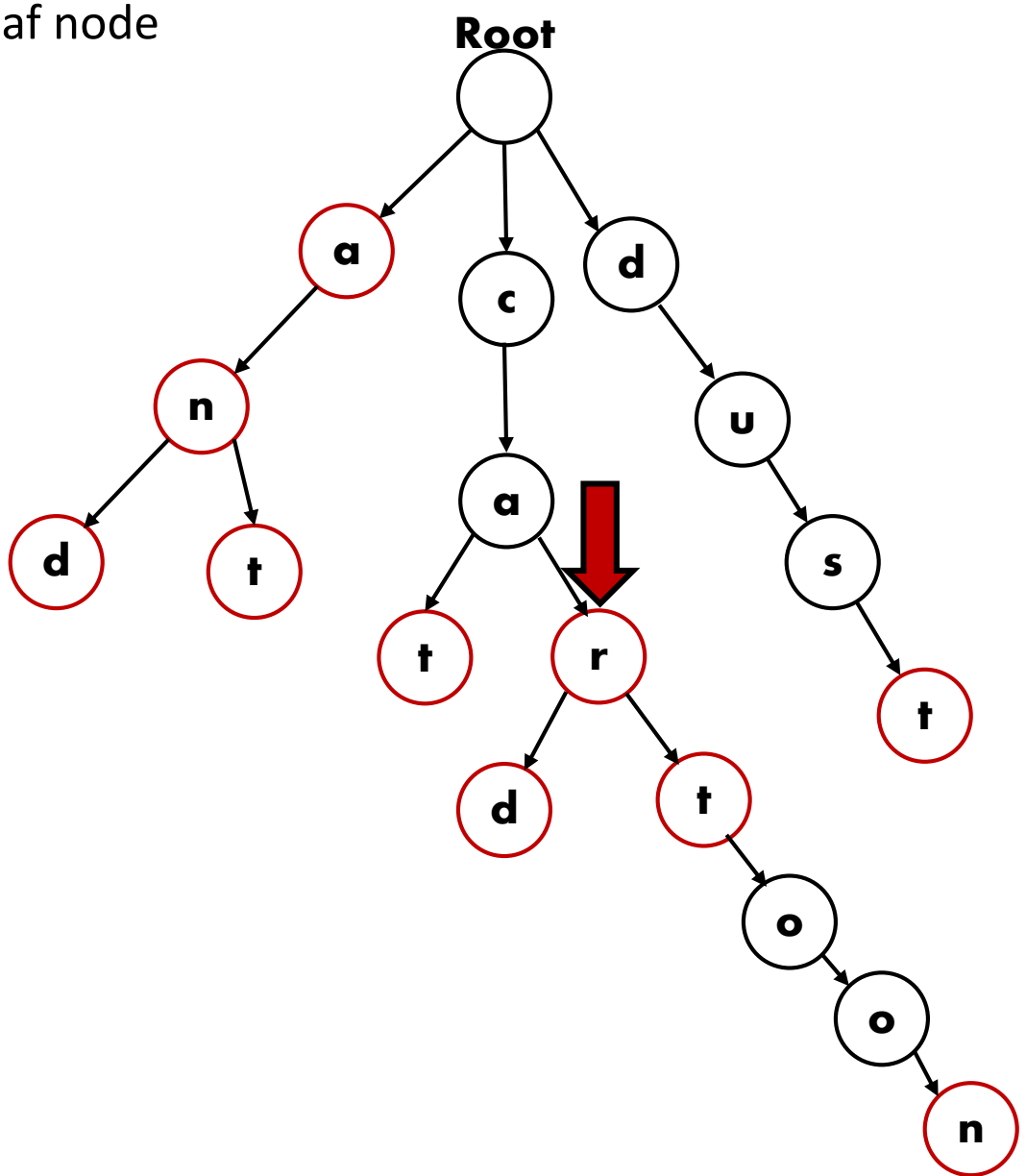
- Calling every child node to leaf node

Input

car

Suggested words

car
card
cart
cartoon



실습과제 : Autocomplete 구현

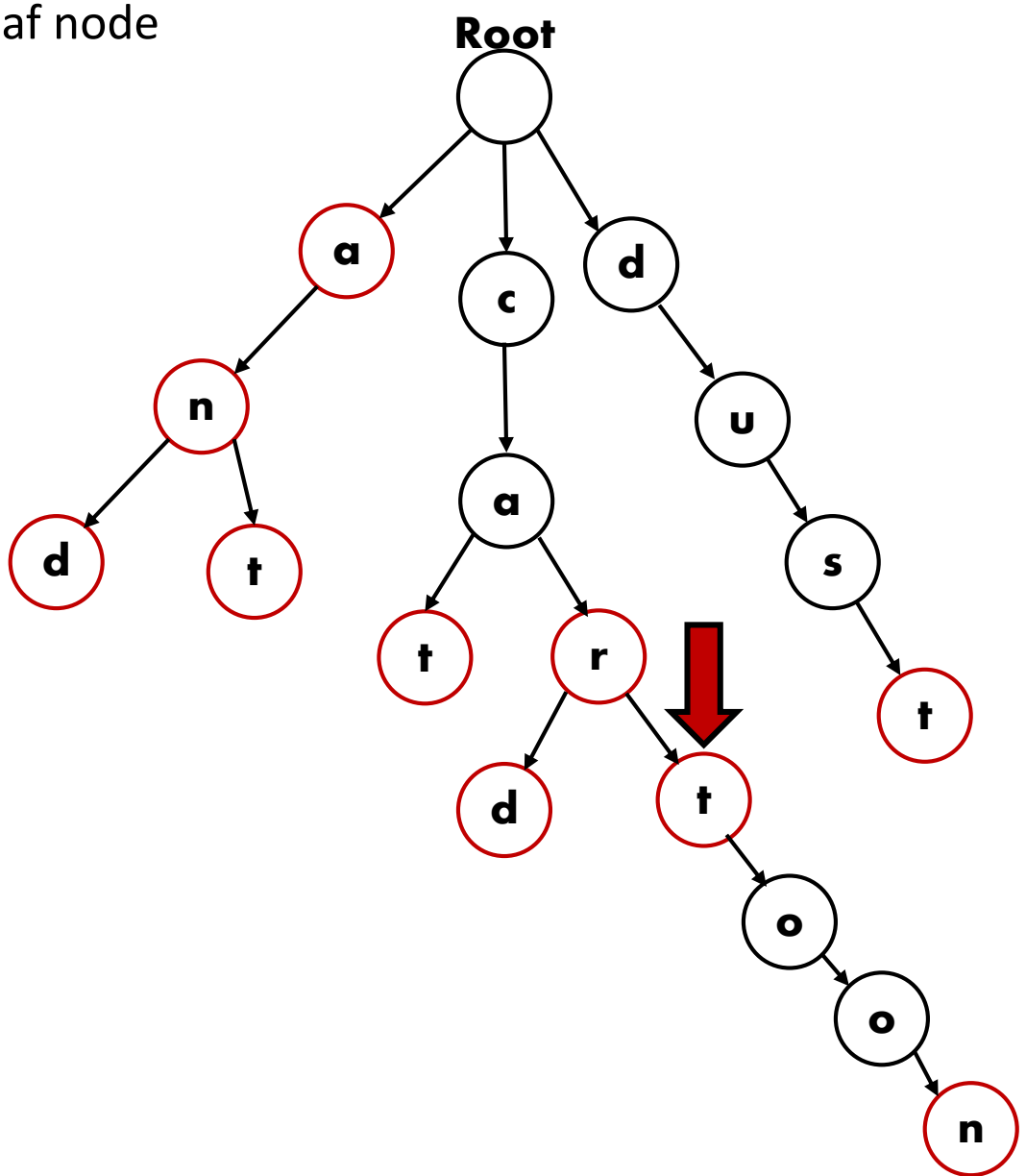
- Calling every child node to leaf node

Input

cart

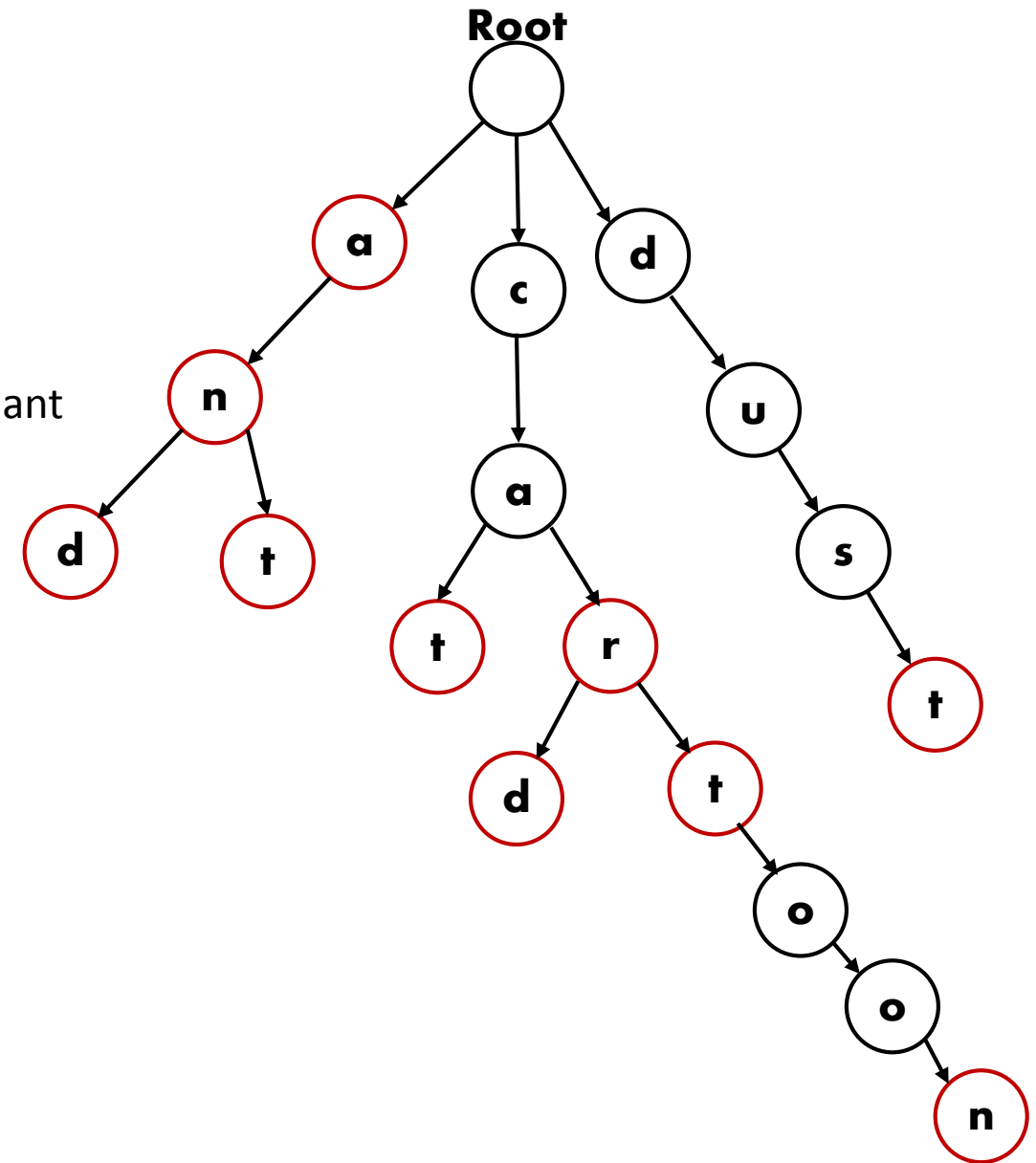
Suggested words

cart
cartoon



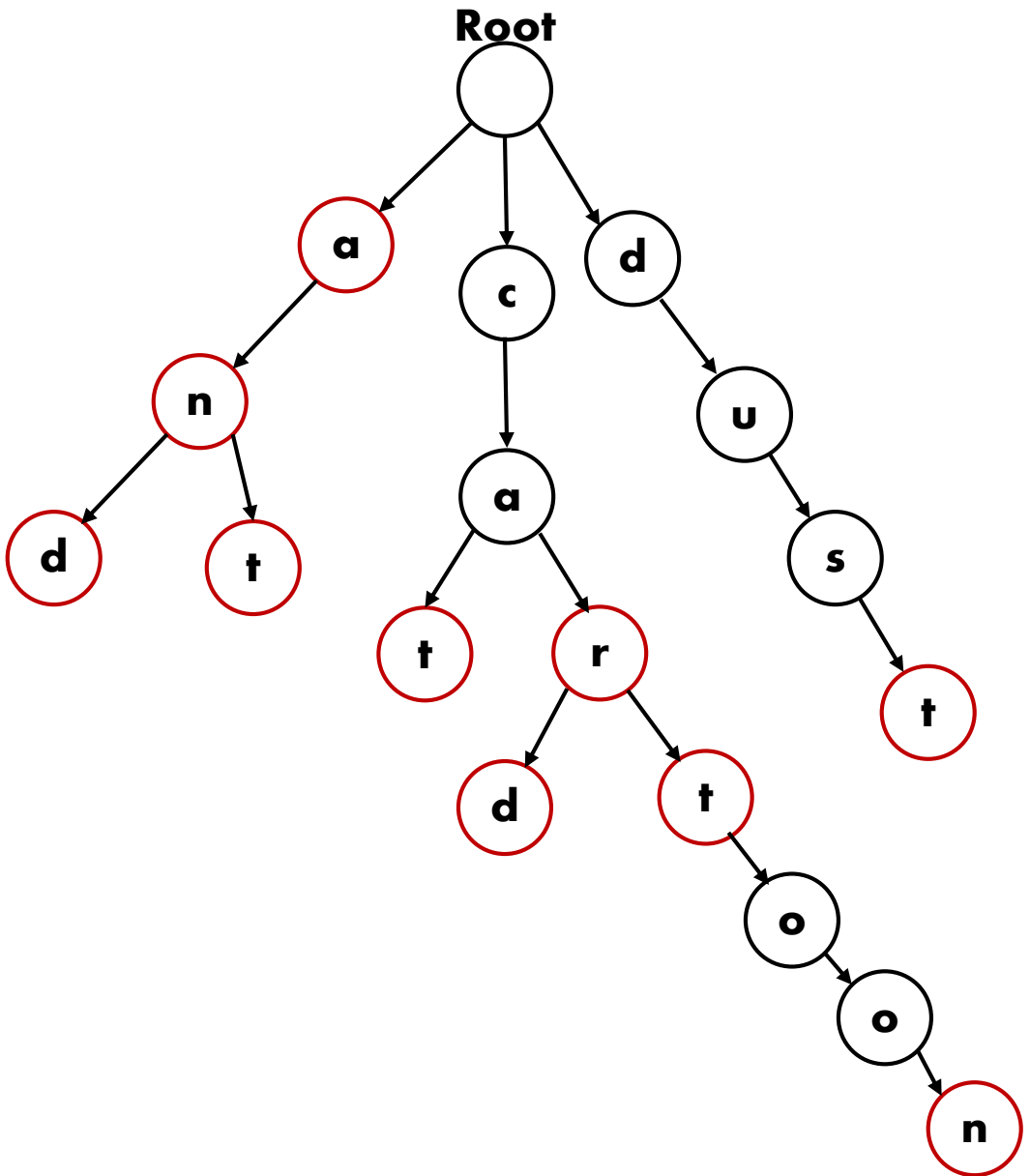
TRIE (edited)

- Data structure
  - From retrieval
  - Digital tree, prefix tree, …
  - Node data : char
  - Path from root to descendant forms a word
- Analysis
  - word length : M
  - Insertion: O(M)
  - Search : O(M)



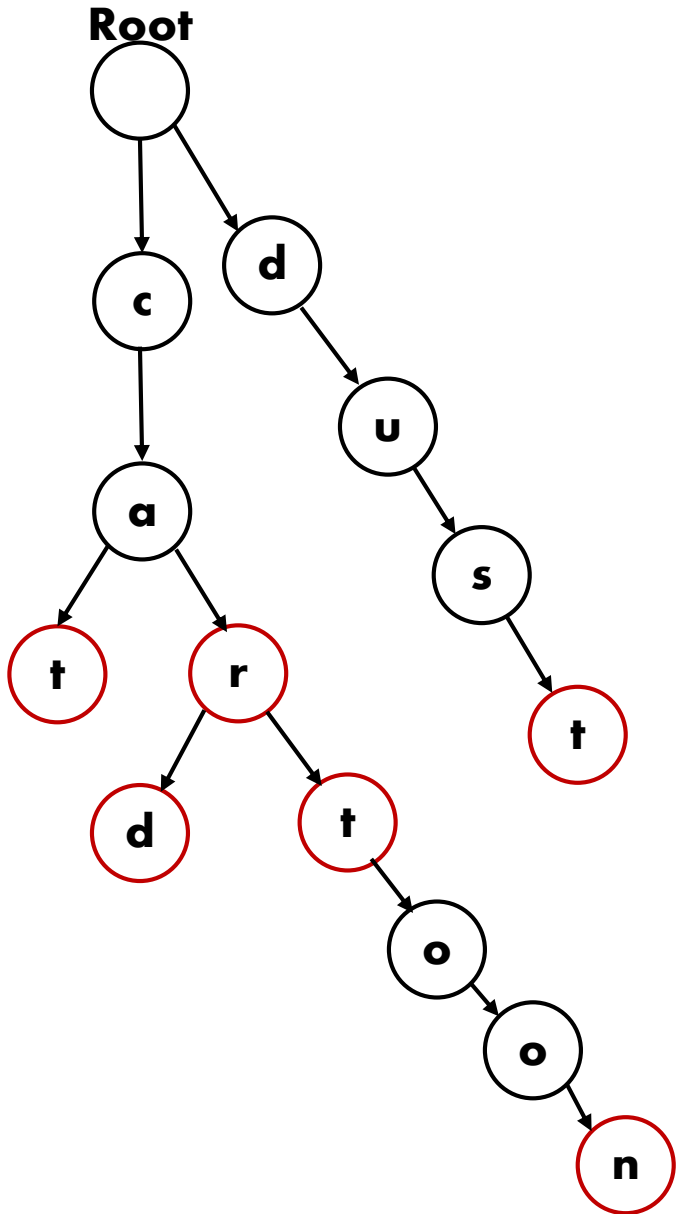
TRIE (edited)

- Functions
  - Insertion
  - Deletion
  - Print
  - Find



TRIE function : Insert

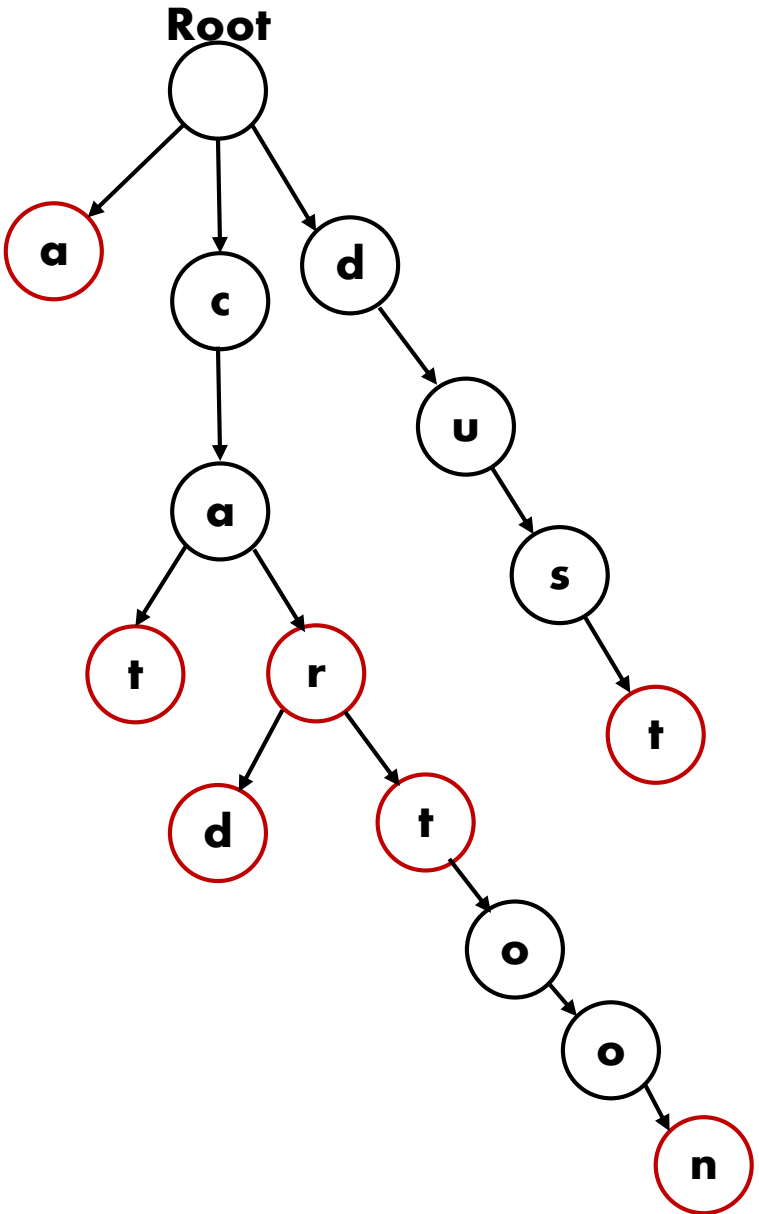
Insert



TRIE function : Insert

Insert

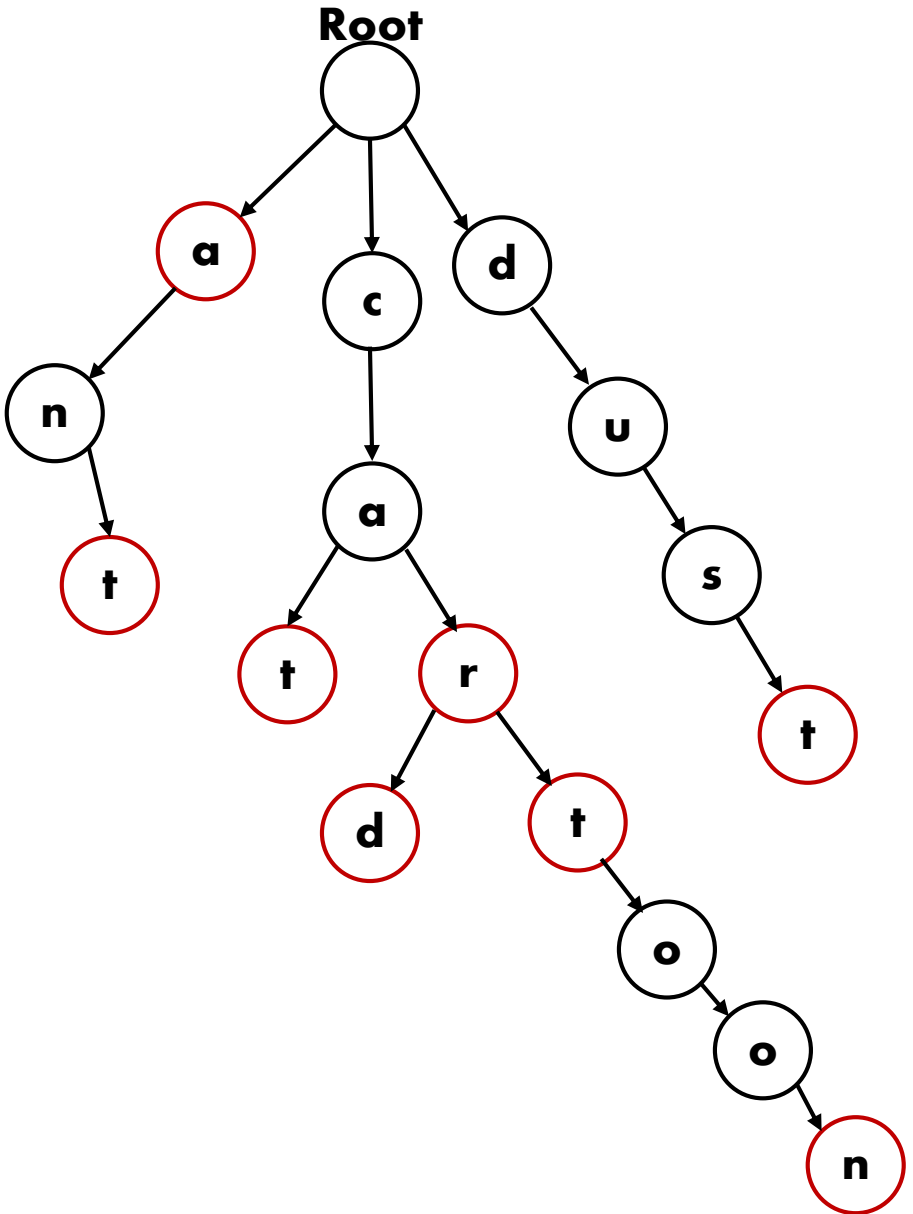
a



TRIE function : Insert

Insert

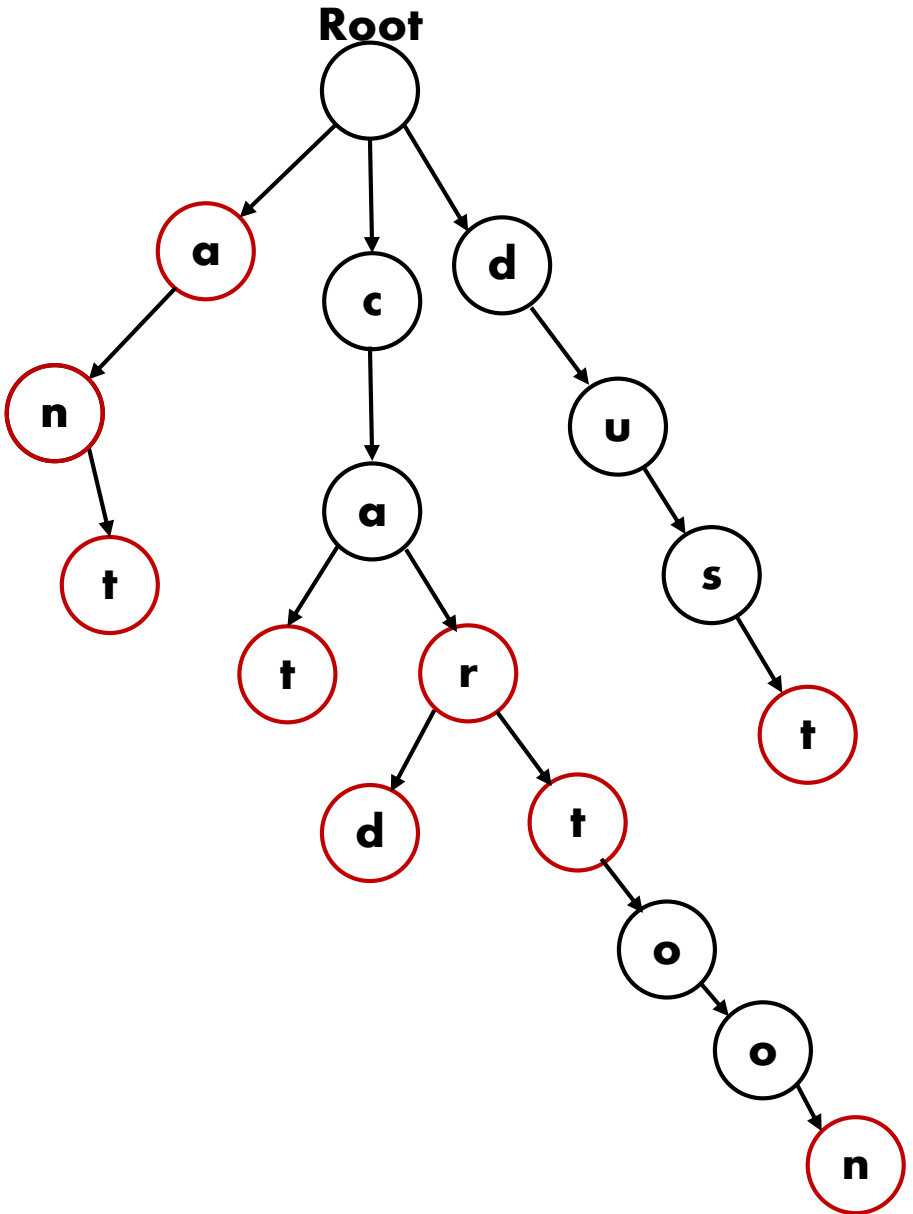
ant



TRIE function : Insert

Insert

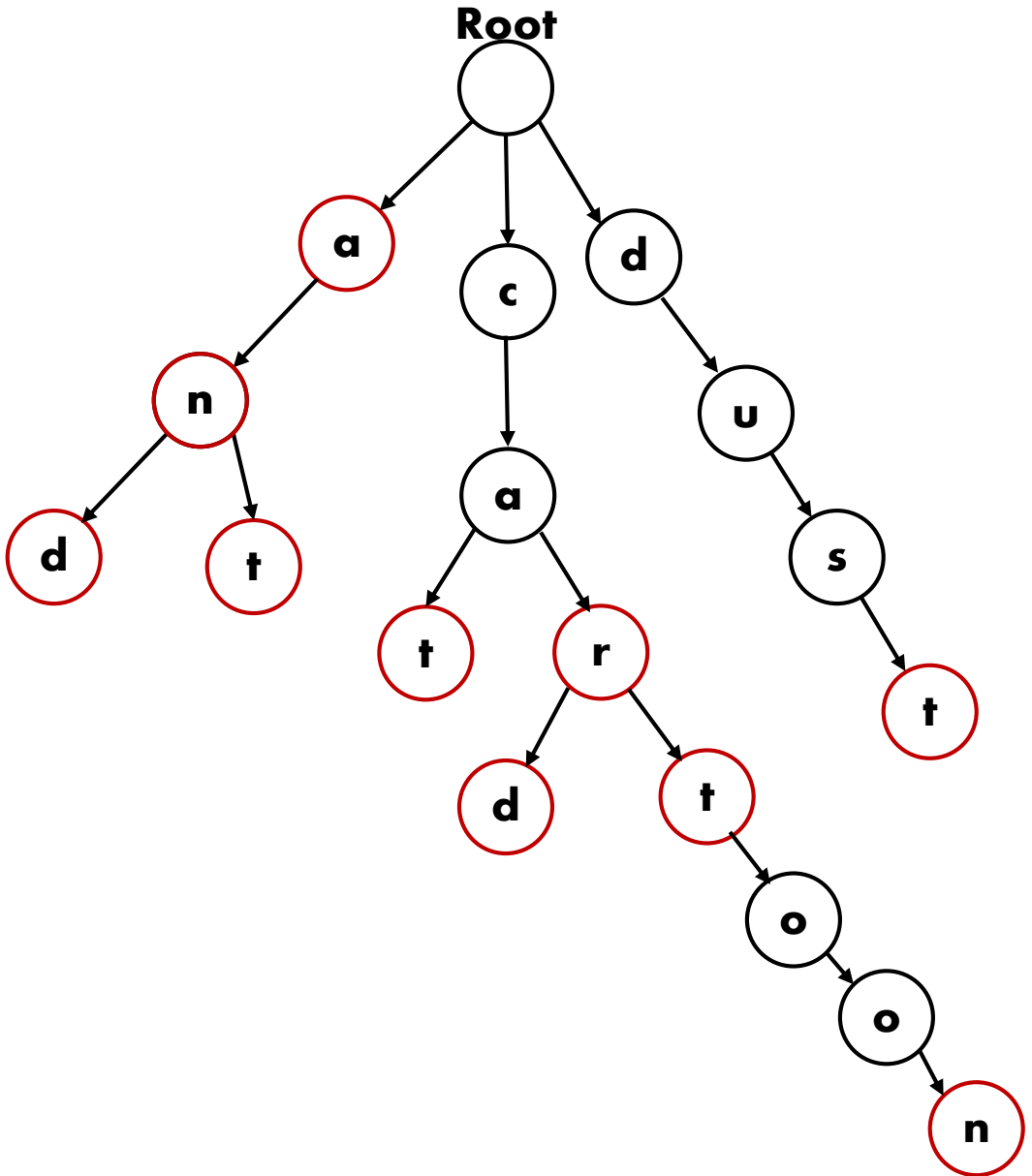
an



TRIE function : Insert

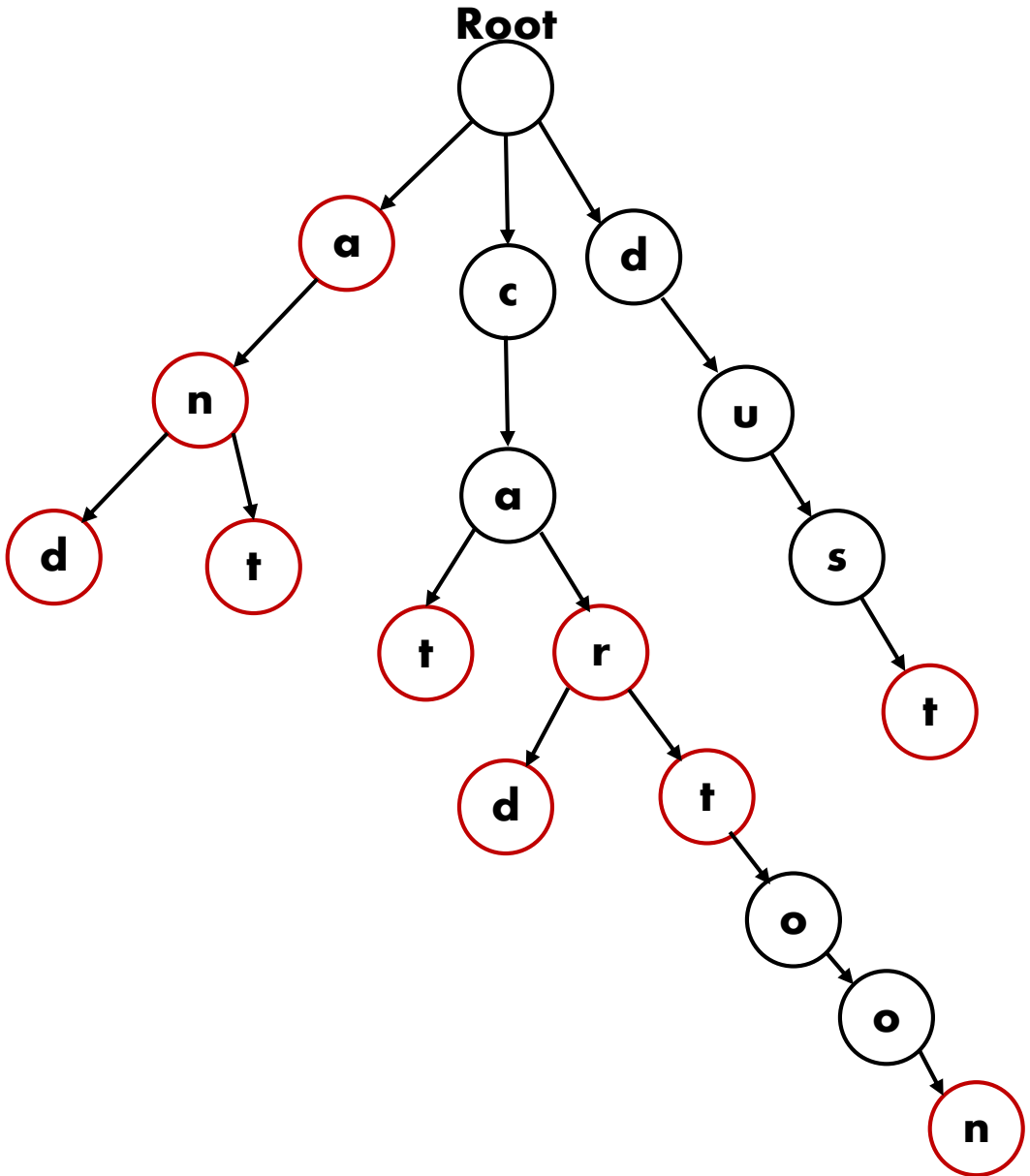
Insert

and



TRIE function : Delete

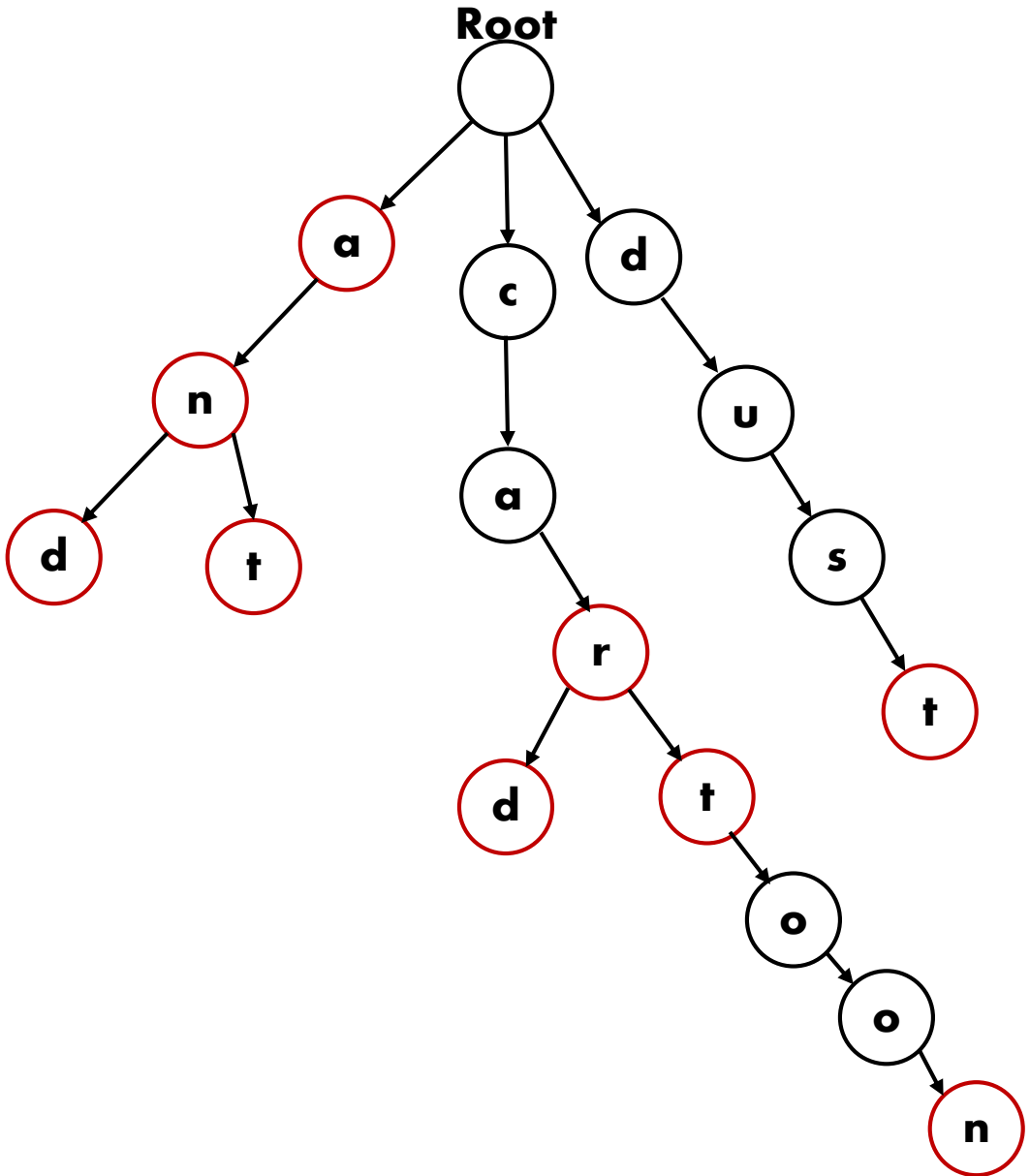
Delete



TRIE function : Delete

Delete

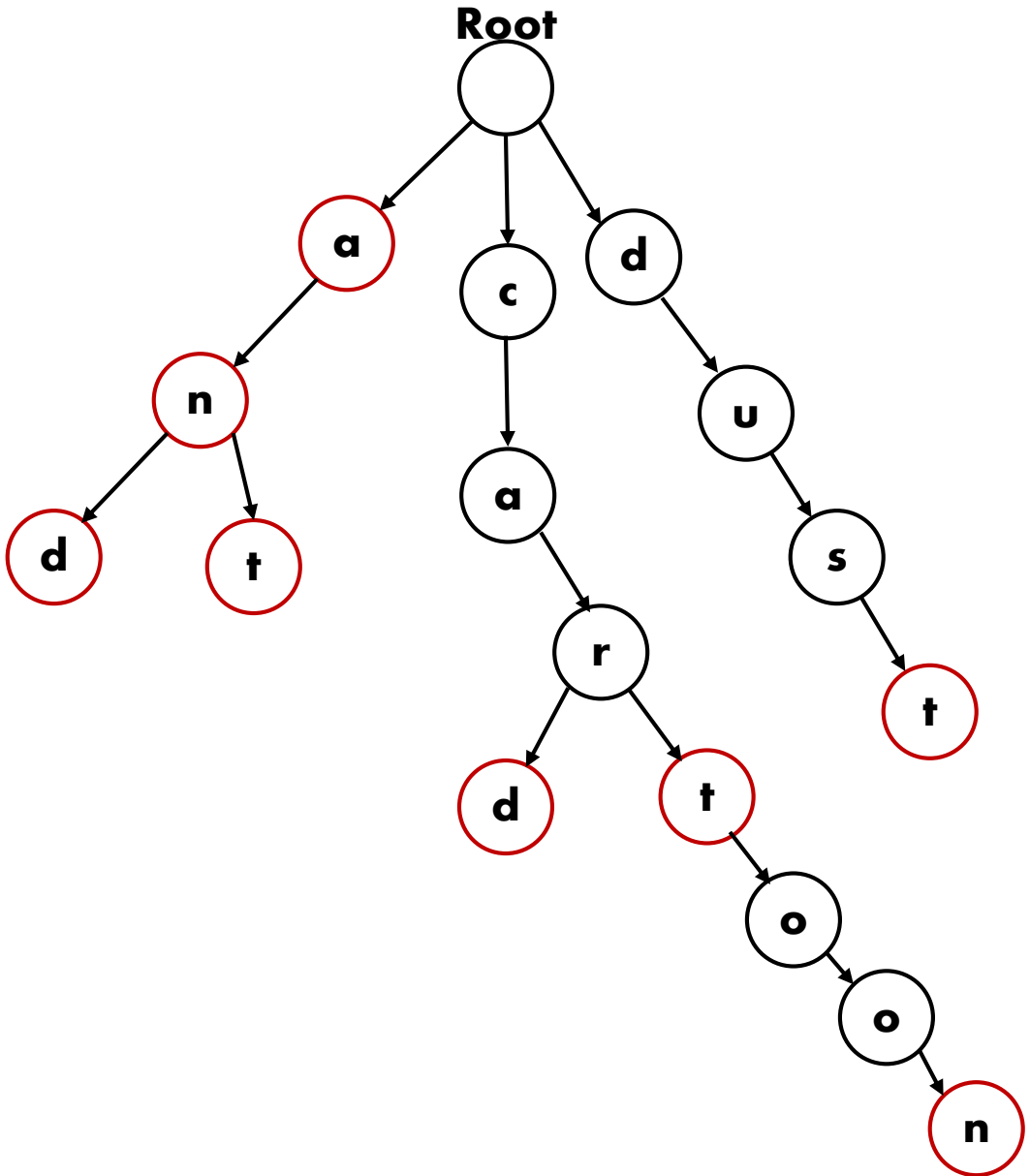
cat



TRIE function : Delete

Delete

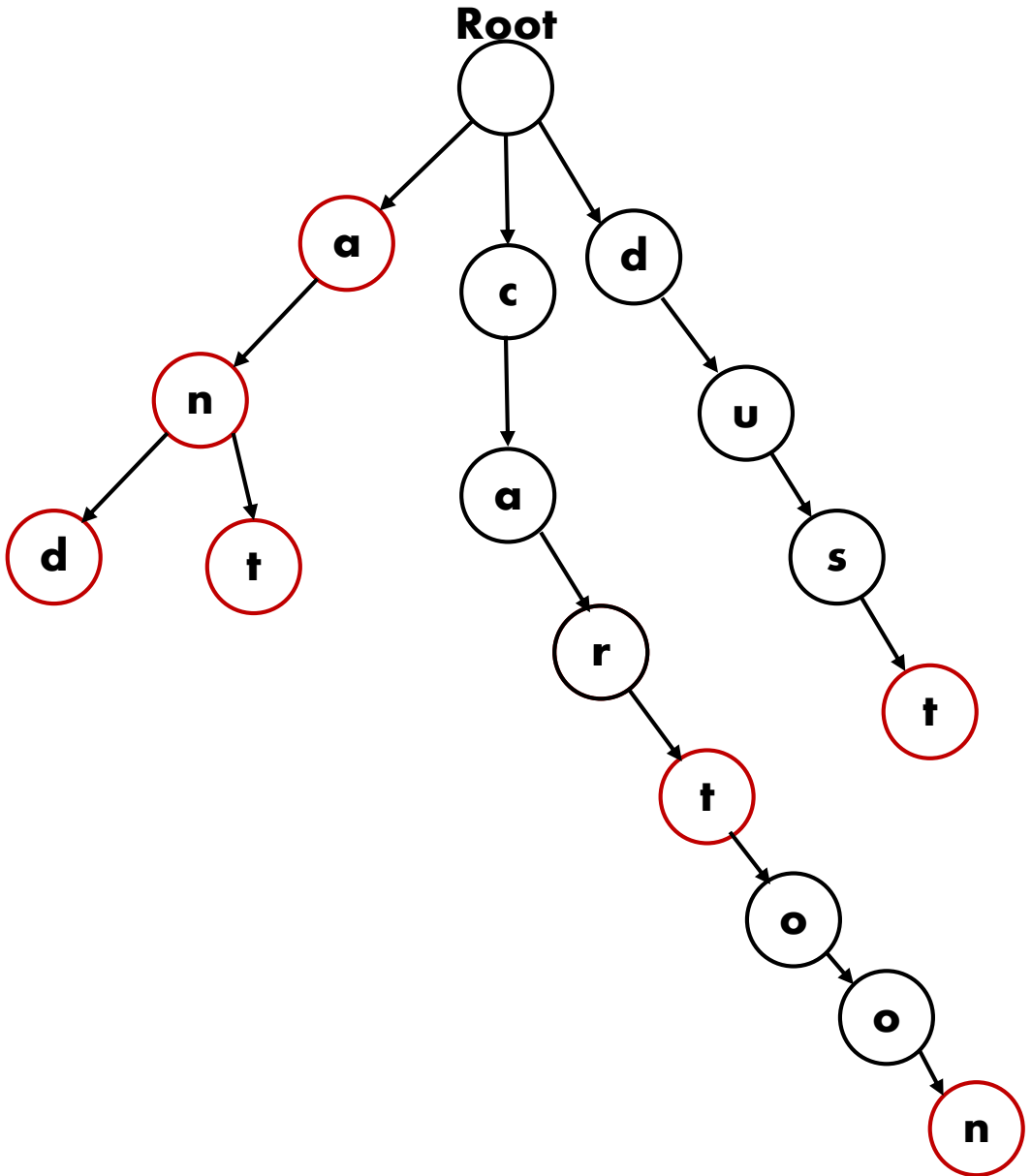
car



TRIE function : Delete

Delete

card



TRIE Node (edited)

- Member variables
  - TRIENode** childs : childs with number of alphabet
    - Dynamic allocation (length = 26)
    - Use alphabet ascii code
  - bool isWord : the variable to check whether the word exists
  - char data : for convenience saving the alphabet of current node

- Member function
  - void print() : print the data of the node

```
class TRIENode
{
public:
    TRIENode();
    ~TRIENode();
    TRIENode(char data, bool isWordNew);
    void print();
    bool isWord;
    TRIENode** childs;
    char data;
};
```

TRIE Implementation (edited)

```
class TRIE
{
public:
    TRIE();
    ~TRIE();
    string find(string word);
    void insertion(string word);
    void deletion(string word);
    void print(string word);
    void printAll();
private:
    TRIENode* root;
    void print_slave(string word, TRIENode* node);
};
```

- void insertion(string word) : inserting node with word
  - Case 1 : If has child, just make ‘isWord’ true
  - Case 2 : If has no child, insert the node
- void deletion(string word) : deleting node with word
  - Case 1 : If has child, just make ‘isWord’ false
  - Case 2 : If has no child, delete the node
  - 예외 처리 “NO WORD” 출력 (test_main 참고)

TRIE Implementation (edited)

```
class TRIE
{
public:
    TRIE();
    ~TRIE();
    string find(string word);
    void insertion(string word);
    void deletion(string word);
    void print(string word);
    void printAll();
private:
    TRIENode* root;
    void print_slave(string word, TRIENode* node);
};
```

- void print(string word) Autocomplete
  - Print the whole node from the word
  - Recursion
    - void print_slave(string word, TRIENode* node)
  - Loop

- void printAll(string word)
  - Print the whole words in TRIE
  - With help of function print

TRIE Implementation (edited)

```
class TRIE
{
public:
    TRIE();
    ~TRIE();
    string find(string word);
    void insertion(string word);
    void deletion(string word);
    void print(string word);
    void printAll();
private:
    TRIENode* root;
    void print_slave(string word, TRIENode* node);
};
```

- string find (string word)
  - If word exists, return word
  - Else, return “”(nothing)

Assignment(edited)

- Due :11/8 23:59

- 주의점
  - Childs 관련 : 알파벳 ascii 이용
  - Print : alphabetical order
  - Find : return word or “”
  - Deletion 예외 처리 “NO WORD” 출력 (test_main 참고)
  - Memory leak 확인