NLP hw#1 : Cky parser 제작

120150241 김상근

1. 개발 환경

C++ 언어와 표준 라이브러리 (Standard Template Library, STL) 을 이용하여 제작하였다. 코드는 unix 기반 컴퓨터에서 작성하였으나, 윈도우 환경의 Visual studio 2010 에서 잘 동작하는 것을 확인하였다. Unix 계열 컴퓨터에서는 make 명령어로 컴파일 할 수 있고, Visual studio 2010 에서는 새로운 프로젝트에 a.cpp 와 grammer.txt, input.txt 를 추가하면 컴파일 할 수 있다.

2. 구조체 설명

```
struct Grammer {
    multimap<string,vector<string> > g;
    multimap<vector<string>,string> bg;
    Grammer() {}
    void insert(svs curGrammer) {
        g.insert(curGrammer);
        bg.insert(pair<vector<string>,
string>(curGrammer.second,curGrammer.first));
    }
    void printG() {
        for ( map<string,vector<string> >::iterator
it=g.begin();it!=g.end();it++ ) {
            fprintf(stdout,"%s -> ",it->first.c_str());
            for ( int i = 0 ; i < (int)it->second.size() ; i++ ) {
                fprintf(stdout,"%s ",it->second[i].c_str());
            }
            fprintf(stdout,"\n");
        }
    }
    void printBg() {
        for ( multimap<vector<string>,string>::iterator
it=bg.begin();it!=bg.end();it++ ) {
            for ( int i = 0 ; i < (int)it->first.size() ; i++ )
                fprintf(stdout,"%s ",it->first[i].c_str());
            fprintf(stdout,"-> ");
            fprintf(stdout,"%s\n",it->second.c_str());
        }
    }
};
```

Grammer 구조체는 multimap<string,vector<string>> 컨테이너 두 개를 갖고 있는데, 이는 문법이 왼쪽일 때 오른쪽 리스트를 구할 수도 있어야하고, 오른쪽 리스트일 때 왼쪽 품사도 구할 수 있어야 하기 때문이다. 또한 key 에 해당하는 품사가 여러개 존재할 수도 있기 때문에 map 이 아닌 다중 키를 제공하는 multimap 을 사용하였다.

```
struct Cky {
    vector<vector<set<pair<string,string> > > > table;
    Cky(){}
    Cky(int n) {
        for ( int i = 0 ; i < n ; i++ ) {
            vector<set<pair<string,string> > > cur;
            for ( int j = 0 ; j < n ; j++ ) {
                set<pair<string,string> > curS;
                cur.push_back(curS);
            }
            table.push_back(cur);
        }
    }
    void print() {
        for ( int i = 0 ; i < (int)table.size() ; i++ ) {
            for ( int j = 0 ; j < (int)table[i].size() ; j++ ) {
                fprintf(stdout,"[");
                for ( set<pair<string,string> >::iterator
it=table[i][j].begin();it!=table[i][j].end();it++ )

fprintf(stdout,"%s%s",it==table[i][j].begin()?"":",",(*it).first.c_st
r());
                fprintf(stdout,"]");
            }
            fprintf(stdout,"\n");
        }
    }
};
```

Cky 구조체는 2 차원 dynamic programming table 을 채우기 위한 구조체로써, vector<vector<set<pair<string,string>>>> 컨테이너를 갖고 있다. 두 Vector 는 2 차원 테이블을 의미하고, 각각의 셀에 품사와 그 품사가 만들어 지는 문법 정보를 갖고 있는 집합을 의미한다.

3. 프로그램 플로우 설명

먼저 grammer.txt 에서 문법을 읽어온 후, 파싱하여 Grammer 구조체를 구성한다. 그 후 input.txt 에서 파싱할 문자열을 입력받는데, input.txt 에 여러개의 테스트 문자열이 있다고 가정하고 개행을 기준으로 구분하여 입력받는다. 그리고 파싱할 테스트 문자열마다 파싱하여 결과를 output.txt 에 저장한다.