Theory of Automata Assignment 2

Ali Faisal – 17K-3791

Aiman Siddiqui – 17K-3810

Shayan Shahid – 17K-3851

5/8/19

Scanner Code

```
#include <iostream>
#include <fstream>
#include <stdlib.h>
#include "Queue.h"

using namespace std;

/*
        -------------------------------- CFG 'GRAMMAR RULES' ---------------------------------

        S --> NP VP

        NP --> Pronoun | Proper-Noun | Determiner NOMINAL

        NOMINAL --> Noun NOMINAL | Noun

        VP --> Verb | Verb NP | Verb NP PP | Verb PP

        PP --> Preposition NP

*/

// Non-Terminal Funcion Prototypes
int S(string sen, Queue<string> &q, Queue<string> &out);
int NP(string sen, Queue<string> &q, Queue<string> &out);
int VP(string sen, Queue<string> &q, Queue<string> &out);
int PP(string sen, Queue<string> &q, Queue<string> &out);
int NOMINAL(string sen, Queue<string> &q, Queue<string> &out);
void Display(string sen, Queue<string> &q);

// utility functions
void pushWordsToQueue(string sen, Queue<string> &q);
bool findNominal(string a);
bool findPronoun(string a);
bool findProp_Nouns(string a);
bool findDet(string a);
bool findPreposition(string a);

// global variables
string _s = "S", _np = "NP", _vp = "VP", _nom = "NOM", _pp = "PP"; // Non-Terminals
string _det = "Det", _pron = "ProN", _prop = "ProP", _verb = "Verb", _prep = "PreP", _noun = "Noun"; //
Terminals
bool _prepbool=false;
string result;
```

```
// main
int main()
{
	Queue<string> q;
	Queue<string> out;

	string sen;

	cout<<"Input Sentence: ";
	getline(cin, sen);

	cout<<"\n\n";

	if(!sen.length())
	{
		cout<<"\nInvalid Sentence.\n";
		return 0;
	}

	pushWordsToQueue(sen, q);

	if(!S(sen, q, out))
	{
		cout<<"\nInvalid Sentence.\n";
	}
	else
	{
		cout<<"\nValid Sentence.\n\n";
		pushWordsToQueue(sen, q);
		Display(sen,q);
	}

	return 0;
}


int S(string sen, Queue<string> &q, Queue<string> &out)
{
//	S --> NP VP

	if(NP(sen, q, out))
	{
		return VP(sen, q, out);
	}
	else
	{
		return 0;
		}

	}

	int NP(string sen, Queue<string> &q, Queue<string> &out)
	{
	//	NP --> Pronoun | Proper-Noun | Determiner NOMINAL

		if(q.emptyQ())
		{
			return 0;
		}

		ifstream inFile1, inFile2, inFile3;

		inFile1.open("Pronouns.txt", ios::in);
		inFile2.open("Proper-Nouns.txt", ios::in);
		inFile3.open("Determiners.txt", ios::in);

		string wordI = q.peekQ() , wordF;

		while(inFile1 >> wordF)
		{
			if(wordF == wordI)
			{

				q.deQueue();
				return 1;
			}
		}

		while(inFile2 >> wordF)
		{
			if(wordF == wordI)
			{

				q.deQueue();
				return 1;
			}
		}

		while(inFile3 >> wordF)
		{
			if(wordF == wordI)
			{

				q.deQueue();
				return NOMINAL(sen, q, out);
```

```
                }
        }
}

int VP(string sen, Queue<string> &q, Queue<string> &out)
{
//      VP --> Verb | Verb NP | Verb NP PP | Verb PP

        if(q.emptyQ())
        {
                return 0;
        }

        ifstream inFile1;

        inFile1.open("Verbs.txt", ios::in);

        string wordI = q.peekQ() , wordF;
        bool checkV = 0;

        while(inFile1 >> wordF)
        {
                if(wordF == wordI)
                {
                        q.deQueue();
                        checkV = 1;
                        break;
                }
        }

        if(!checkV)
        {
                return 0;
        }

        // AT THIS POINT THERE IS A VERB

        if(NP(sen, q, out))
        {
                if (q.emptyQ())
                {

                        return 1;
                }
                else
                {
```

```
                return  PP(sen, q, out);
            }

            return 1;
        }


        if(PP(sen, q, out))
        {


            return 1;
        }


        return 1;
}

int PP(string sen, Queue<string> &q, Queue<string> &out)
{
//      PP --> Preposition NP

        if(q.emptyQ())
        {
            return 0;
        }

        ifstream inFile1;

        inFile1.open("Prepositions.txt", ios::in);

        string wordI = q.peekQ() , wordF;
        bool checkP = 0;

        while(inFile1 >> wordF)
        {
            if(wordF == wordI)
            {
                _prepbool=true;
                q.deQueue();
                checkP = 1;
                break;
            }
        }

        if(!checkP)
        {
            return 0;
```

```
		}
		else
		{
			return NP(sen, q, out);
		}
}

int NOMINAL(string sen, Queue<string> &q, Queue<string> &out)
{
//		NOMINAL --> Noun NOMINAL | Noun

		if(q.emptyQ())
		{
			return 0;
		}

		ifstream inFile1;

		inFile1.open("Nouns.txt", ios::in);

		string wordI = q.peekQ() , wordF;
		bool checkN = 0;

		while(inFile1 >> wordF)
		{
			if(wordF == wordI)
			{
				_nom = wordI;

				q.deQueue();
				checkN = 1;
				break;
			}
		}

		if(!checkN)
		{
			return 0;
		}
		else
		{
			NOMINAL(sen, q, out);
			return 1;
		}
}

// utlity functions
bool findNominal(string a)
```

```
{
        ifstream inFile1;

        inFile1.open("Nouns.txt", ios::in);

        string wordF;


        while(inFile1 >> wordF)
        {
                if(wordF == a)
                {
                        inFile1.close();
                        return true;
                }
        }
        inFile1.close();
        return false;
}

bool findPronoun(string a)
{
        ifstream inFile1;

        inFile1.open("Pronouns.txt", ios::in);

        string wordF;


        while(inFile1 >> wordF)
        {
                if(wordF == a)
                {
                        inFile1.close();
                        return true;
                }
        }
        inFile1.close();
        return false;
}
bool findPreposition(string a)
{
        ifstream inFile1;

        inFile1.open("Prepositions.txt", ios::in);

        string wordF;
```

```
		while(inFile1 >> wordF)
		{
			if(wordF == a)
			{
				inFile1.close();
				return true;
			}
		}
		inFile1.close();
		return false;
}


bool findProp_Nouns(string a)
{
		ifstream inFile1;

		inFile1.open("Proper-Nouns.txt", ios::in);

		string wordF;


		while(inFile1 >> wordF)
		{
			if(wordF == a)
			{
				inFile1.close();
				return true;
			}
		}
		inFile1.close();
		return false;
}
bool findDet(string a)
{
		ifstream inFile1;

		inFile1.open("Determiners.txt", ios::in);

		string wordF;


		while(inFile1 >> wordF)
		{
			if(wordF == a)
			{
				inFile1.close();
```

```
                return true;
            }
        }
        inFile1.close();
        return false;

}
void pushWordsToQueue(string sen, Queue<string> &q)
{
        int i = 0;
        string words;

        while(i != sen.length())
        {
            if(sen[i] == ' ' || sen[i] == '.' || sen[i] == '!' || sen[i] == '?' || sen[i]=='\n')
            {
                q.enQueue(words);
                words.clear();
            }
            else
            {
                words = words + sen[i];
            }


            i++;
        }
        q.enQueue(words);
}

void Display(string sen, Queue<string> &q)
{
        bool check;
        cout<<"S\n";
        cout<<"NP VP\n";
        ifstream inFile1;
        inFile1.open("Pronouns.txt", ios::in);
        string wordI=q.peekQ(),wordF;
        while(inFile1 >> wordF)
        {
            if(wordF == wordI)
            {
                cout<<"Pronouns VP\n";
                _pron=wordF;
                cout<<wordF<<" VP\n";
                result+=_pron;
                q.deQueue();
                check=true;
                break;
            }
            else check =false;
        }
        inFile1.close();
        if(!check)
        {
            inFile1.open("Proper-Nouns.txt", ios::in);
            wordI=q.peekQ();
            while(inFile1 >> wordF)
            {
                if(wordF == wordI)
                {
                    cout<<"Proper-Nouns VP\n";
                    _prop=wordF;
                    result+=_prop;
                    cout<<wordF<<" VP\n";
                    check=true;
                    q.deQueue();
                    break;
                }
                else
                check =false;
            }
            inFile1.close();
        }
        if(!check)
        {
            inFile1.open("Determiners.txt", ios::in);
            wordI=q.peekQ();
            while(inFile1 >> wordF)
            {
                if(wordF == wordI)
                {
                    cout<<"Det Nom VP\n";
                    _det=wordF;
                    cout<<wordF<<" Nom VP\n";
                    result=wordF;
                    check=true;
                    q.deQueue();

                    if(q.emptyQ())
                    {
                        break;
                        }
                }
            }
                    inFile1.close();
```

```
                                        if(check)
                                        while(findNominal(q.peekQ()))
                                        {
                                                cout<<result<<" Noun Nom"<<" VP\n";
                                                result+= " "+ q.peekQ();
                                                cout<<result<<" Nom VP\n";
                                                q.deQueue();
                                                if(q.emptyQ())
                                                                break;
                                        }
                                        cout<<result<<" VP\n";

                        //cout<<endl<<result;

                //      q.displayQ();

                }
                                // now for VP
                        inFile1.open("Verbs.txt", ios::in);

                         wordI = q.peekQ();
                        while(inFile1 >> wordF)
                        {
                                if(wordF == wordI)
                                {
                                        _verb=wordF;
                                        q.deQueue();
                                        break;
                                }
                        }
                        inFile1.close();
                //      q.displayQ();
                        wordI = q.peekQ();
                        //if(!q.emptyQ())
                //      cout<<".............";
        //      cout<<result;
                        if(findPronoun(wordI) || findProp_Nouns(wordI) || findDet(wordI))
                        {
                                cout<<result<<" Verbs NP";
                                if(_prepbool)
                                cout<<" PP\n";
                                else
                                cout<<"\n";
                                result+=" "+_verb;
                                cout<<result <<" NP";
                                if(_prepbool)
                                cout<<" PP\n";
                                else cout<<"\n";
```

```
inFile1.open("Pronouns.txt", ios::in);
wordI=q.peekQ();
while(inFile1 >> wordF)
{
        if(wordF == wordI)
        {
                //cout<<"Pronouns VP\n";
                _pron=wordF;
                cout<< result<< " Pronoun ";
                result+= " "+_pron;
                if(_prepbool)
                cout<<" PP\n";
                else cout<<"\n";
                q.deQueue();
                cout<< result;
                if(_prepbool)
                cout<<" PP\n";
                else cout<<"\n";
                check=true;
                break;
        }
        else check =false;
}
inFile1.close();
if(!check)
{
        inFile1.open("Proper-Nouns.txt", ios::in);
        wordI=q.peekQ();
        while(inFile1 >> wordF)
        {
                if(wordF == wordI)
                {
                        //cout<<"Proper-Nouns VP\n";
                        _prop=wordF;
                        cout<< result<< " Prop-Noun ";
                        result+= " "+_prop;
                        if(_prepbool)
                        cout<<" PP\n";
                        else cout<<"\n";
                        q.deQueue();
                        cout<< result;
                        if(_prepbool)
                        cout<<" PP\n";
                        else cout<<"\n";
                        check=true;
                        break;
                }
                else
```

```
                    check =false;
            }
            inFile1.close();
    }
    if(!check)
    {
            inFile1.open("Determiners.txt", ios::in);
            wordI=q.peekQ();
            while(inFile1 >> wordF)
            {
                    if(wordF == wordI)
                    {

                            _det=wordF;
                            cout<< result<< " det NOM ";
                            result+= " "+_det;
                            if(_prepbool)
                            cout<<" PP\n";
                            else cout<<"\n";
                            cout<< result;
                            if(_prepbool)
                            cout<<" NOM PP\n";
                            else cout<<"  NOM \n";
                            check=true;
                            q.deQueue();

                            if(q.emptyQ())
                            {
                                     break;
                                    }
                    }
            }
                            inFile1.close();
                            if(check)
                            while(findNominal(q.peekQ()))
                            {
                                    if(!q.emptyQ())
                                    cout<<result<<" Noun Nom";;
                                    if(_prepbool)
                                    cout<<" PP\n";
                                    else cout<<"\n";
                                    result+= " "+ q.peekQ();
                                    cout<< result;
                                    if(_prepbool)
                                    cout<<" PP\n";
                                    else cout<<"\n";
                                    q.deQueue();
                                    if(q.emptyQ())
```

```
                                                break;

                                    }


                        }

                        if(_prepbool)
                        {
                                    wordI=q.peekQ();
                                    cout<< result<< " Preposition NP\n";
                                    result+=" "+ wordI;
                                    q.deQueue();
                                    cout<< result<<" NP\n";
                                    //q.displayQ();
                        inFile1.open("Pronouns.txt", ios::in);
                        wordI=q.peekQ();
                        check=false;
                        while(inFile1 >> wordF)
                        {
                                    if(wordF == wordI)
                                    {
                                                //cout<<"Pronouns VP\n";
                                                _pron=wordF;
                                                cout<< result<< " Pronoun ";
                                                result+= " "+_pron;
                                                 cout<<"\n";
                                                q.deQueue();
                                                cout<< result;
                                                 cout<<"\n";
                                                check=true;
                                                break;
                                    }
                                    else check =false;
                        }
                        inFile1.close();
                        if(!check)
                        {
                                    inFile1.open("Proper-Nouns.txt", ios::in);
                                    wordI=q.peekQ();
                                    while(inFile1 >> wordF)
                                    {
                                                if(wordF == wordI)
                                                {
                                                            //cout<<"Proper-Nouns VP\n";
                                                            _prop=wordF;
                                                            cout<< result<< " Prop-Noun ";
                                                             cout<<"\n";
```

```
                    result+= " "+_prop;
                    q.deQueue();
                    cout<< result;
                     cout<<"\n";
                    check=true;
                    break;
               }
               else
               check =false;
          }
          inFile1.close();
     }
     if(!check)
     {
          inFile1.open("Determiners.txt", ios::in);
          wordI=q.peekQ();
          while(inFile1 >> wordF)
          {
               if(wordF == wordI)
               {

                    _det=wordF;
                    cout<< result<< " det NOM ";
                    result+= " "+_det;
                     cout<<"\n";
                    cout<< result;
                    cout<<"\n";
                    check=true;
                    q.deQueue();

                    if(q.emptyQ())
                    {
                          break;
                         }
               }
          }
                    inFile1.close();
                    if(check)
                    while(findNominal(q.peekQ()))
                    {
                         if(!q.emptyQ())
                         cout<<result<<" Noun Nom";;
                         cout<<"\n";
                         result+= " "+ q.peekQ();
                         cout<< result;
                         cout<<"\n";
                         q.deQueue();
                         if(q.emptyQ())
```

```
                                        break;
                                }
                        }
                }//
        }
        else
        {
                cout<<result<<" Verbs PP\n";

                result+=" "+_verb;
                cout<<result <<" PP\n";
                wordI=q.peekQ();
                cout<< result<< " Preposition NP\n";
                result+=" "+ wordI;
                q.deQueue();
                cout<< result<<" NP\n";
                        //q.displayQ();
                inFile1.open("Pronouns.txt", ios::in);
                wordI=q.peekQ();
                check=false;
                while(inFile1 >> wordF)
                {
                        if(wordF == wordI)
                        {
                                //cout<<"Pronouns VP\n";
                                _pron=wordF;
                                cout<< result<< " Pronoun ";
                                result+= " "+_pron;
                                 cout<<"\n";
                                q.deQueue();
                                cout<< result;
                                 cout<<"\n";
                                check=true;
                                break;
                        }
                        else check =false;
                }
                inFile1.close();
                if(!check)
                {
                        inFile1.open("Proper-Nouns.txt", ios::in);
                        wordI=q.peekQ();
                        while(inFile1 >> wordF)
                        {
                                if(wordF == wordI)
                                {
                                        //cout<<"Proper-Nouns VP\n";
                                        _prop=wordF;
                                        cout<< result<< " Prop-Noun ";
                                         cout<<"\n";
                                        result+= " "+_prop;
                                        q.deQueue();
                                        cout<< result;
                                         cout<<"\n";
                                        check=true;
                                        break;
                                }
                                else
                                check =false;
                        }
                        inFile1.close();
                }
                if(!check)
                {
                        inFile1.open("Determiners.txt", ios::in);
                        wordI=q.peekQ();
                        while(inFile1 >> wordF)
                        {
                                if(wordF == wordI)
                                {

                                        _det=wordF;
                                        cout<< result<< " det NOM ";
                                        result+= " "+_det;
                                         cout<<"\n";
                                        cout<< result;
                                        cout<<"\n";
                                        check=true;
                                        q.deQueue();

                                        if(q.emptyQ())
                                        {
                                                 break;
                                                }
                                }
                        }
                                        inFile1.close();
                                        if(check)
                                        while(findNominal(q.peekQ()))
                                        {
                                                if(!q.emptyQ())
                                                cout<<result<<" Noun Nom";;
                                                cout<<"\n";
                                                result+= " "+ q.peekQ();
                                                cout<< result;
                                                cout<<"\n";
```

```
                                    q.deQueue();
                                    if(q.emptyQ())
                                    break;
                                }
                        }
                }
}
```

Queue

```
#include <iostream>
#include <stdlib.h>
#include <math.h>

#define SIZE_Q 100

using namespace std;

template <typename T>
class Queue{

        private:

                T *array;
                int capacity, left, right, sizeQ;

        public:

                Queue()
                {
                        capacity = SIZE_Q;
                        array = new T[SIZE_Q];

                        left = -1;
                        right = -1;
                        sizeQ = 0;
                }

                Queue(int capacity)
                {
                        this->capacity = abs(capacity);
                        array = new T[this->capacity];

                        left = -1;
                        right = -1;
                        sizeQ = 0;
```

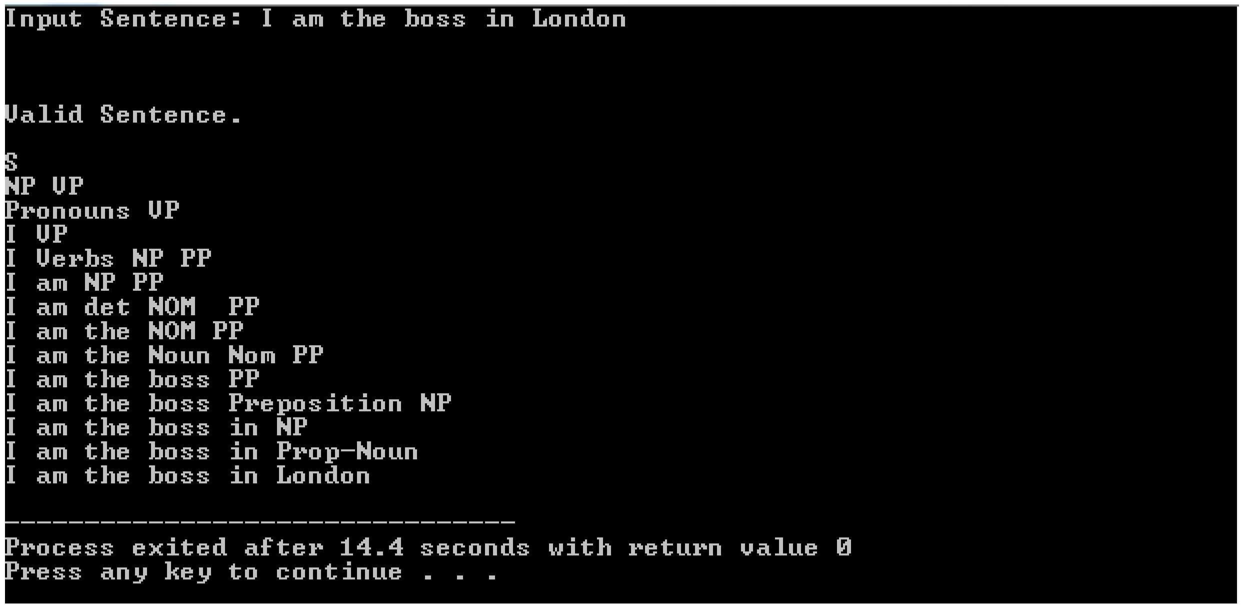
```
}

void enQueue(T val)
{
        if((left == 0 && right == capacity-1) || (right == left - 1))
        {
                cout<<"\nQueue Is Full.\n";
        }
        else if(left == -1 && right == -1)
        {
                left = right = 0;
                array[right] = val;
                sizeQ++;
        }
        else if(left != 0 && right == capacity-1)
        {
                right = 0;
                array[right] = val;
                sizeQ++;
        }
        else
        {
                right++;
                array[right] = val;
                sizeQ++;
        }
}

T deQueue()
{
        T val;

        if(left == -1 && right == -1)
        {
                cout<<"\nQueue Is Empty.\n";
        }
        else if(left == capacity - 1)
        {
                val = array[left];
                left = 0;
                sizeQ--;
        }
        else if(left == right)
        {
                val = array[left];
                left = right = -1;
                sizeQ--;
        }
```

```
		else
		{
			val = array[left];
			left++;
			sizeQ--;
		}

		return val;
	}

	T peekQ()
	{
		if(!emptyQ())
		{
			return array[left];
		}
	}

	void clearQ()
	{
		left = right = -1;
	}

	int getSize()
	{
		return sizeQ;
	}

	int getCapacity()
	{
		return capacity;
	}

	bool emptyQ()
	{
		if(left == -1 && right == -1)
		{
			return 1;
		}
		return 0;
	}

	bool fullQ()
	{
		if((left == 0 && right == capacity-1) || (right == left - 1))
		{
			return 1;
		}
```

```
                return 0;
        }

        void displayQ()
        {
                if(left < right)
                {
                        for(int i=left; i<=right; i++)
                        {
                                cout<<array[i]<<"\n";
                        }
                }
                else if(right < left)
                {
                        for(int i=0; i<=right; i++)
                        {
                                cout<<array[i]<<" \n";
                        }

                        for(int i=left; i<capacity; i++)
                        {
                                cout<<array[i]<<" \n";
                        }
                }
                else
                {
                        if(!emptyQ())
                        {
                                cout<<array[left]<<"\n";
                        }
                        else
                        {
                                cout<<"\nEmpty Queue\n";
                        }
                }
        }

        ~Queue()
        {
                if(array != 0)
                {
                        delete[] array;
                        array = 0;
                }
        }
};
```

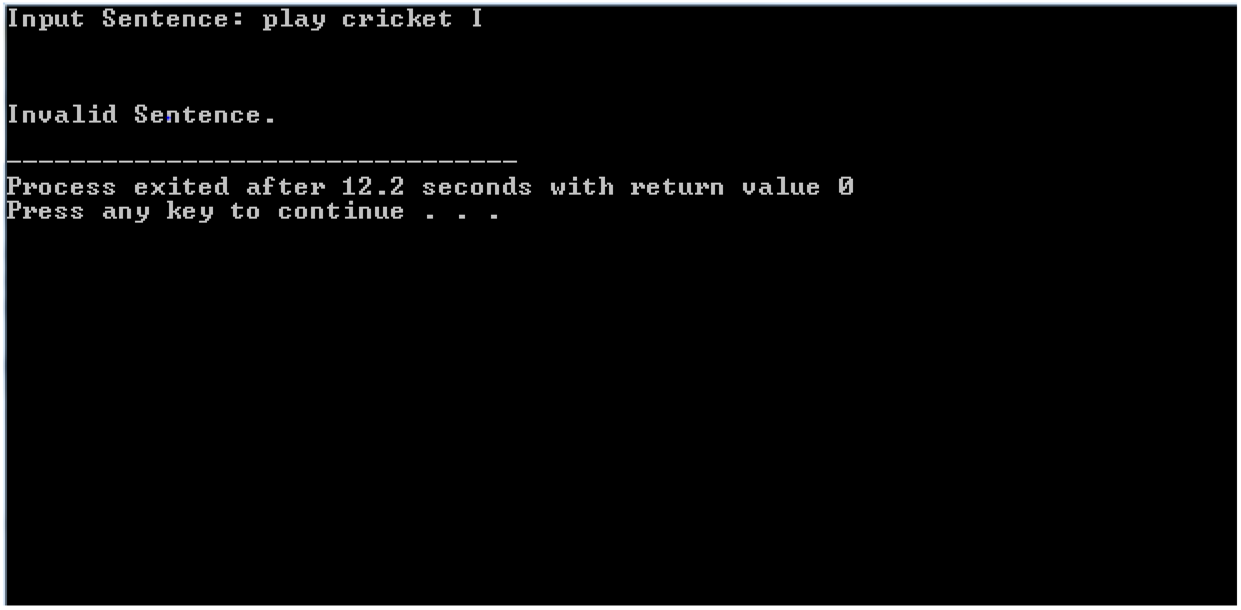
Output

Valid



```
Input Sentence: I am the boss in London



Valid Sentence.

S
NP VP
Pronouns VP
I VP
I Verbs NP PP
I am NP PP
I am det NOM  PP
I am the NOM PP
I am the Noun Nom PP
I am the boss PP
I am the boss Preposition NP
I am the boss in NP
I am the boss in Prop-Noun
I am the boss in London

--------------------------------
Process exited after 14.4 seconds with return value 0
Press any key to continue . . .
```

Invalid



```
Input Sentence: play cricket I



Invalid Sentence.

--------------------------------
Process exited after 12.2 seconds with return value 0
Press any key to continue . . .
```