Aim: Prepare a Program for finding First & Follow Sets.

Software Requirements: GNU C Compiler(GNU Compiler Collection), <stdlib.h> <stdio.h> <ctype.h> <string.h> <iostream> <fstream> <strings.h>

Hardware Requirements: Processor, Memory, Standard Input, Standard Output.

Knowledge Required:

To understatnd this program the reader must have understanding of the Algorithms for computing First and Follow set as well as hand on cpp.

Description:

Program should accept the Grammer as an input and should find the first and follow sets of the given grammer.

Algorithm:

step 1. Read a grammer from the input file. int read_grammar(char*);

Reads a grammar from a specified file and stores it into g (array of structure gammar)
this function ignores the white spaces
it assumes the grammar written in file is CFG (Context Free Grammer)
it assumes all the possible alternatives for a same nonterminal are
written in same line and they are separated by '|'

step 2. ask user choice 1. FIRST set 2. FOLLOW set

get choice.

step 3. if choice = 1 (FIRST SET)

Ask for,

1. Compute FIRST of all NTs
2. Compute FIRST for a specific SENTENTIAL form

if 1. void All_NTs_first_set();
This function computes the first set for all nonterminal in given grammar g
For that first it copies each non-terminal into str and then computes FIRST(str)
void display_first_set(char*);
This function displays the set passed as an argument
else
Ask for,
String.
char* compute_first_set(char*);

This function computes FISRT set of any string of terminal and non-terminal
It applys the following rules:

CASE 1:	= FIRST(XYZ) = {X}	if X is a terminal
CASE 2:	= FIRST(X)	if '@' is not in FIRST(X)
CASE 3:	= FIRST(X) - {@} + FIRST(YZ)	if '@' is in FIRST(X)

Here X,Y,Z represents any grammar symbol (i.e terminal or non-terminal)

Ask for continue. If yes go to *step 2.*

step 4. if choice = 2 (FOLLOW SET)

char* compute_follow_set(char);
This function computes FOLLOW(NT) for given non-terminal NT by using the following rules:
If there is a production of the form A -> <alpha>B<beta> then

FOLLOW(B) = FIRST(<beta>)	if '@' is not there in FIRST(<beta>)
= FIRST(<beta>)-{@} + FOLLOW(A)	if '@' is in FIRST(<beta>)

Ask for continue. If yes go to *step 2.*

step 5. Exit

Input/Output:

```
jazz@linuxmint ~/Desktop/ACT/First Follow $ g++ FIRST_FO.CPP -o ff
FIRST_FO.CPP: In function ‘char* compute_follow_set(char)’:
FIRST_FO.CPP:103: warning: address of local variable ‘follow_set’ returned
jazz@linuxmint ~/Desktop/ACT/First Follow $ ./ff gram5.txt

        1.FIRST
        2.FOLLOW

        Enter your choice
1


********** GRAMMAR **********

                A -> B | C
                B -> n | i
                C -> (D)
                D -> DA | A

********** GRAMMAR **********
```

1. Compute FIRST of all NTs
2. Compute FIRST for a specific SENTENTIAL form

Enter your chice:1

********** FIRST **********

First of A : (,i,n
First of B : i,n
First of C : (
First of D : (,i,n

********** FIRST **********

Would you like to continue PRESS 'Y' or 'y'y

1.FIRST
2.FOLLOW

Enter your choice
2

********** GRAMMAR **********

A -> B | C
B -> n | i
C -> (D)
D -> DA | A

********** GRAMMAR **********

********** FOLLOW **********

FOLLOW of A: $,(,),i,n
FOLLOW of B: $,(,),i,n
FOLLOW of C:
FOLLOW of D: (,),i,n

********** FOLLOW **********

Would you like to continue PRESS 'Y' or 'y'n

```
jazz@linuxmint ~/Desktop/ACT/First Follow $ ./ff gram5.txt

        1.FIRST
        2.FOLLOW

        Enter your choice
1


********** GRAMMAR **********

            A -> B | C
            B -> n | i
            C -> (D)
            D -> DA | A


********** GRAMMAR **********


        1. Compute FIRST of all NTs
        2. Compute FIRST for a specific SENTENTIAL form

        Enter your chice:2

        Enter String: C -> (D)

********** FIRST **********

            First of  C : (
********** FIRST **********

Would you like to continue PRESS 'Y' or 'y'n
jazz@linuxmint ~/Desktop/ACT/First Follow $
```