COMPILER ENGINEERING

YACC PROGRAM IMPLEMENTATION

TITLE: Medicine Recommendation System using YACC and Lex

- Mahima S [2022115018]

1. Introduction

This project implements a basic Medicine Recommendation System that takes user-inputted symptoms and suggests appropriate medicines based on a predefined set of symptoms. The system uses **YACC** (Yet Another Compiler Compiler) and **Lex** (a lexical analyzer generator) to parse and process input in a structured manner. The goal of this system is to provide an interactive way for users to get medicine suggestions based on common health symptoms.

The system operates in a terminal-based environment, where users can input symptoms one by one and receive corresponding medicine recommendations. The system also supports the "exit" command, which gracefully terminates the application.

2. Objective

- To implement a command-line-based application using **YACC** and **Lex** that maps inputted symptoms to medicine suggestions.
- To demonstrate how **Lex** and **YACC** can be used together to build a simple and interactive application in the healthcare domain.
- To provide an extensible framework for adding more symptoms and corresponding medicines.

3. System Design

The system consists of two main components:

1. **Lex Program (Lexical Analyzer)**:
    - The Lex program is responsible for identifying and tokenizing the symptoms and the exit command entered by the user. It converts the user input into tokens that can be processed by the YACC program.
    - It defines rules for various symptoms (e.g., "fever", "headache") and the "exit" keyword.
2. **YACC Program (Parser)**:
    - The YACC program receives tokens from Lex and applies grammar rules to determine the flow of the program.

    - Based on the identified symptom, it calls the appropriate function to suggest a corresponding medicine.
    - The program also handles the "exit" command to terminate the execution.

4. Implementation Details

Lex Program:

- The Lex program contains regular expressions to match symptoms and the exit keyword.
- For each recognized symptom, it returns a token (SYMPTOM), while the exit command is associated with the EXIT token.
- The input string is processed character by character, and Lex generates tokens that are passed to YACC for further processing.

YACC Program:

- The YACC program defines rules for symptom matching and links each symptom to its respective medicine.
- The program suggests medicine based on predefined mappings.
- The exit command is handled explicitly by terminating the program using exit(0).

5. Grammar and Lexical Rules

- **Lex Rules**: These are used to match symptoms and the "exit" command.
    - For instance, the Lex rule for fever would be:

```
fever { return SYMPTOM; }
```

- **YACC Rules**: These are used to structure the input and handle the logic.
    - For example:

```
symptoms:
  SYMPTOM { suggest_medicine(yytext); }
```

6. Functionality

- **Input and Output**:
    - The user enters symptoms like fever, headache, or cold. Based on the input, the system outputs the corresponding medicine suggestion.
    - The user can exit the program by typing exit.
- **Handling Multiple Symptoms**:
    - The system allows multiple symptoms to be entered in sequence. Each symptom is processed, and a recommendation is printed.

CODE :

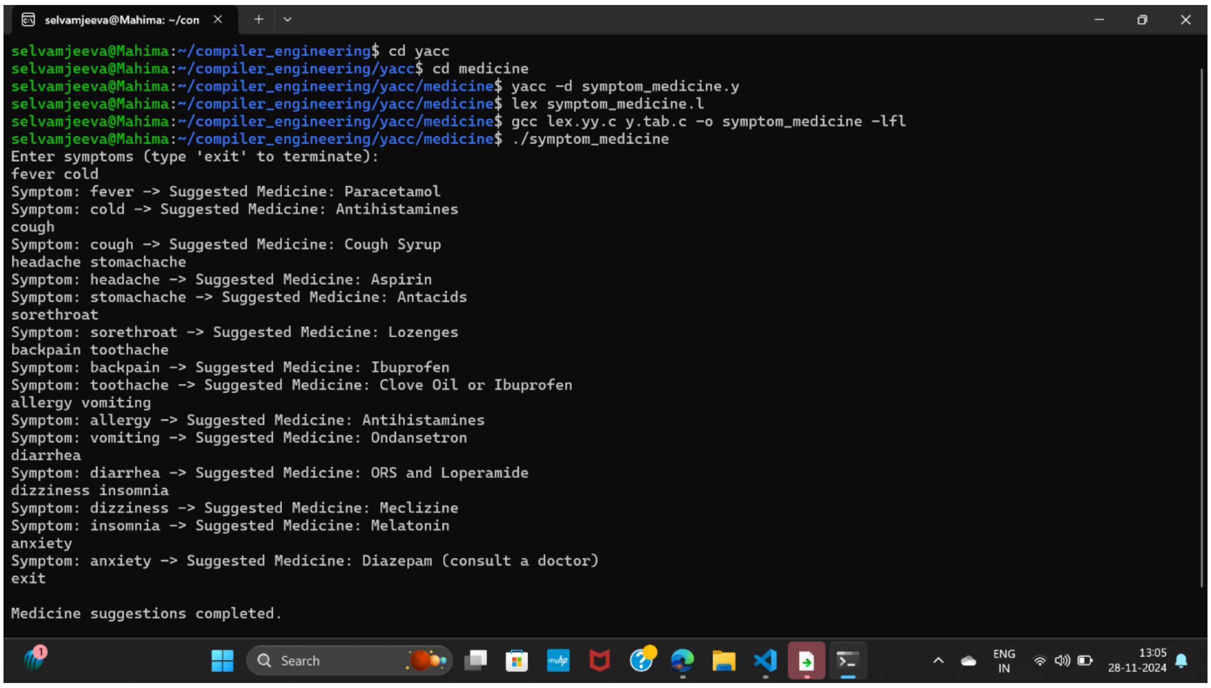
Yacc code

```
%{
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

extern char *yytext;

void yyerror(const char *s);
int yylex();

void suggest_medicine(char *symptom);
%}

%token SYMPTOM EXIT

%%

symptom_input:
  symptoms
  {
    printf("\nMedicine suggestions completed.\n");
  }
  ;

symptoms:
  SYMPTOM
  {
    suggest_medicine(yytext);
  }
  |
  symptoms SYMPTOM
  {
    suggest_medicine(yytext);
  }
  |
  EXIT
  {
    printf("Exiting program. Goodbye!\n");
    exit(0);
  }
  ;

%%

void suggest_medicine(char *symptom) {
  if (strcmp(symptom, "fever") == 0) {
    printf("Symptom: %s -> Suggested Medicine: Paracetamol\n", symptom);
  } else if (strcmp(symptom, "headache") == 0) {
    printf("Symptom: %s -> Suggested Medicine: Aspirin\n", symptom);
  } else if (strcmp(symptom, "cold") == 0) {
    printf("Symptom: %s -> Suggested Medicine: Antihistamines\n", symptom);
  } else if (strcmp(symptom, "cough") == 0) {
    printf("Symptom: %s -> Suggested Medicine: Cough Syrup\n", symptom);
  } else if (strcmp(symptom, "stomachache") == 0) {
    printf("Symptom: %s -> Suggested Medicine: Antacids\n", symptom);
  } else if (strcmp(symptom, "sorethroat") == 0) {
    printf("Symptom: %s -> Suggested Medicine: Lozenges\n", symptom);
  } else if (strcmp(symptom, "backpain") == 0) {
    printf("Symptom: %s -> Suggested Medicine: Ibuprofen\n", symptom);
  } else if (strcmp(symptom, "toothache") == 0) {
    printf("Symptom: %s -> Suggested Medicine: Clove Oil or Ibuprofen\n", symptom);
  } else if (strcmp(symptom, "allergy") == 0) {
    printf("Symptom: %s -> Suggested Medicine: Antihistamines\n", symptom);
  } else if (strcmp(symptom, "vomiting") == 0) {
    printf("Symptom: %s -> Suggested Medicine: Ondansetron\n", symptom);
  } else if (strcmp(symptom, "diarrhea") == 0) {
    printf("Symptom: %s -> Suggested Medicine: ORS and Loperamide\n", symptom);
  } else if (strcmp(symptom, "dizziness") == 0) {
    printf("Symptom: %s -> Suggested Medicine: Meclizine\n", symptom);
  } else if (strcmp(symptom, "insomnia") == 0) {
    printf("Symptom: %s -> Suggested Medicine: Melatonin\n", symptom);
  } else if (strcmp(symptom, "anxiety") == 0) {
    printf("Symptom: %s -> Suggested Medicine: Diazepam (consult a doctor)\n", symptom);
  } else {
    printf("Symptom: %s -> Suggested Medicine: Not Found\n", symptom);
  }
}

void yyerror(const char *s) {
  fprintf(stderr, "Error: %s\n", s);
}

int main() {
  printf("Enter symptoms (type 'exit' to terminate):\n");
  yyparse();
  return 0;
}
```

lex code

```
%{
#include "y.tab.h"
%}
%%
exit             { return EXIT; }
fever             { return SYMPTOM; }
headache           { return SYMPTOM; }
cold              { return SYMPTOM; }
cough              { return SYMPTOM; }
stomachache          { return SYMPTOM; }
sorethroat          { return SYMPTOM; }
backpain           { return SYMPTOM; }
toothache           { return SYMPTOM; }
allergy            { return SYMPTOM; }
vomiting           { return SYMPTOM; }
diarrhea           { return SYMPTOM; }
dizziness          { return SYMPTOM; }
insomnia           { return SYMPTOM; }
anxiety            { return SYMPTOM; }
[ \t\n]+           { /* Ignore whitespace */ }
.                 { return yytext[0]; }
%%
int yywrap() {
   return 1;
}
```

OUTPUT:



```
selvamjeeva@Mahima:~/compiler_engineering$ cd yacc
selvamjeeva@Mahima:~/compiler_engineering/yacc$ cd medicine
selvamjeeva@Mahima:~/compiler_engineering/yacc/medicine$ yacc -d symptom_medicine.y
selvamjeeva@Mahima:~/compiler_engineering/yacc/medicine$ lex symptom_medicine.l
selvamjeeva@Mahima:~/compiler_engineering/yacc/medicine$ gcc lex.yy.c y.tab.c -o symptom_medicine -lfl
selvamjeeva@Mahima:~/compiler_engineering/yacc/medicine$ ./symptom_medicine
Enter symptoms (type 'exit' to terminate):
fever cold
Symptom: fever -> Suggested Medicine: Paracetamol
Symptom: cold -> Suggested Medicine: Antihistamines
cough
Symptom: cough -> Suggested Medicine: Cough Syrup
headache stomachache
Symptom: headache -> Suggested Medicine: Aspirin
Symptom: stomachache -> Suggested Medicine: Antacids
sorethroat
Symptom: sorethroat -> Suggested Medicine: Lozenges
backpain toothache
Symptom: backpain -> Suggested Medicine: Ibuprofen
Symptom: toothache -> Suggested Medicine: Clove Oil or Ibuprofen
allergy vomiting
Symptom: allergy -> Suggested Medicine: Antihistamines
Symptom: vomiting -> Suggested Medicine: Ondansetron
diarrhea
Symptom: diarrhea -> Suggested Medicine: ORS and Loperamide
dizziness insomnia
Symptom: dizziness -> Suggested Medicine: Meclizine
Symptom: insomnia -> Suggested Medicine: Melatonin
anxiety
Symptom: anxiety -> Suggested Medicine: Diazepam (consult a doctor)
exit

Medicine suggestions completed.
```

7. Code Walkthrough

- **Lex Program**:
  - The Lex program first matches the symptom keywords like fever, headache, etc., and returns them as tokens.
  - For the exit keyword, Lex returns the EXIT token, which is handled in the YACC grammar.
- **YACC Program**:
  - The YACC program processes the tokens. Each symptom is checked against predefined patterns, and corresponding medicines are suggested.
  - The program uses strcmp to compare the input symptom with a list of predefined symptoms and outputs the appropriate medicine.

8. Error Handling

- The system does not handle invalid symptoms directly, but any unrecognized input is ignored or flagged as "Not Found" under the suggest_medicine function.
- An error handling function, yyerror(), is included in the YACC program to report issues related to parsing errors.

9. Extensibility

- The system can easily be extended by adding more symptoms to the Lex and YACC programs.
- Additional functionality such as storing user queries or integrating with an external database for dynamic suggestions can also be incorporated.

10. Conclusion

This project successfully demonstrates the use of Lex and YACC to build a simple interactive medicine recommendation system. By mapping symptoms to specific medicines, users can receive useful health advice through an intuitive command-line interface. The system can be expanded by adding more symptoms, improving error handling, and even incorporating real-time data from healthcare APIs.

The use of Lex and YACC in this context illustrates how compiler construction tools can be applied beyond traditional use cases like language processing to build practical, user-interactive applications in various domains like healthcare.