Project Report

Lexical Analysis for C Compiler

Submitted by
Agnitha Mohanram-15CO201
Vasudha Boddukuri-15CO211

Under Guidance of
Sushmita
Umapriya

National Institute of Technology-Karnataka

Contents

Acknowledgement

We would like to express our gratitude to Ms.Santhi Thilagam, Ms. Sushmita and Ms.Umapriya for their constant support, guidance and help. A special thanks to NITK for providing us with the resources and facilities required to complete our project successfully. This learning experience was extremely helpful and valuable. Last but not the least, we’re very thankful to our parents for their unwavering support and encouragement.

Introduction

Overview

The report elaborates on the lexical analysis phase of building a C-compiler. It describes the procedure used to built the lexical analyzer and contains the output obtained from several test cases. It also enlists the various tools and concepts used in the implementation.

Motivation

A compiler is a software that aims to transform code submitted to it which is written in one programming language(the source language) into another programming language(the target language).
Compilers are extensively used in the programming world and it's essential for a computer science engineer to understand the working of a compiler. By doing so, he/she gains thorough knowledge of the system software and can manipulate it with ease if need arises.
The project aims to build a primitive compiler from scratch through the various phases of compiler design.
This report is on the lexical analysis phase where the input program is converted into sequence of tokens which are stored in a symbol table.

Objective

The objectives of the project are as follows:

- to study the grammar of the source language and identify the lexical components of the language specifications
- to study the input format of Flex and clearly understand how tokens are described
- to identify the tokens to be returned by the scanner
- to store the identified tokens in a symbol table
- to generate the scanner
- to understand parallelisation of programs to achieve optimised performance

Concepts and Tools

Compiler

Compilers are system softwares that convert code written in one language (usually high level language) into an equivalent machine understandable language to generate an executable.

Lexical Analysis

Lexical Analysis is the process of converting the input program into a sequence of tokens. The lexical analyzer scans the source program from left to right in a single pass. It removes unwanted information from the source program.

Hashing

Hashing is the process of converting given string into a key value by using some mathematical functions. These key values are used to optimize time complexity for searching, insertion and deletion from memory.

Hash Table

A hash table is a data structure which is used to map keys generated by hashing to the values they were generated from.

Flex

Lex is a computer program that generates lexical analyzers ("scanners" or "lexers").Lex reads an input stream specifying the lexical analyzer and outputs source code implementing the lexer in the C programming language.

Git

Git is a version control system(VCS) which helps keep track of changes made in projects. It is used to maintain records of changes made to any set of files.

Compiler Design

Compiler

A compiler is computer software that transforms computer code written in one programming language(the source language) into another programming language (the target language) without changing the meaning of the program.
The compiler is expected to make the target code efficient and optimized I terms of time and space complexities.
Compiler design deals with the translation and optimization processes. It covers basic translation mechanism, error detection and errors recovery.

Compiler design has two phases:
FRONT END

a) Lexical Analysis
b) Syntax Analysis
c) Semantic Analysis
d) Intermediate Code Generation

BACK END

e) Code Optimization
f) Code Generation

The front end builds an intermediate representation(IR) of the source code. It also manages the symbol table and mapping of each token in the program to its associated information.

The back end on the other hand deals with the CPU architecture specific optimizations and code generation.

Lexical Analysis

Lexical analysis is the first phase of compiler design. It takes the source code as input an translates it into a sequence of tokens. It removes the unnecessary elements of the source code such as white spaces and comments and generates appropriate error messages if any invalid token is encountered.

Syntax Analysis

A syntax analyzer checks the syntactical correctness of the source program. It’s takes the token of streams generated by the lexical phase as input and analyzes
the source code against the production rules to detect errors. The output is in the form of a parse tree.

Semantic Analysis

Semantics of the language add meaning to its tokens and syntax structure. The semantic analyzer analysis the parse tree and determines if it derives any meaning. The output of this phase is a more verified parse tree.

Intermediate Code Generation

Intermediate code is generated from the verified parse tree. It is then converted to machine language through the last two phases. All the phases till the intermediate code generation are platform independent, whereas the next two phases are not. In order to build a compiler for a platform, one need not start from scratch. He/she could take the intermediate code from an already existing compiler and build the last two phases.

Code optimization

The code optimizer transforms the code into an optimized version which is designed to consume fewer resources and work faster. There are two types of optimizations, machine dependent and machine independent.

Target Code Generator

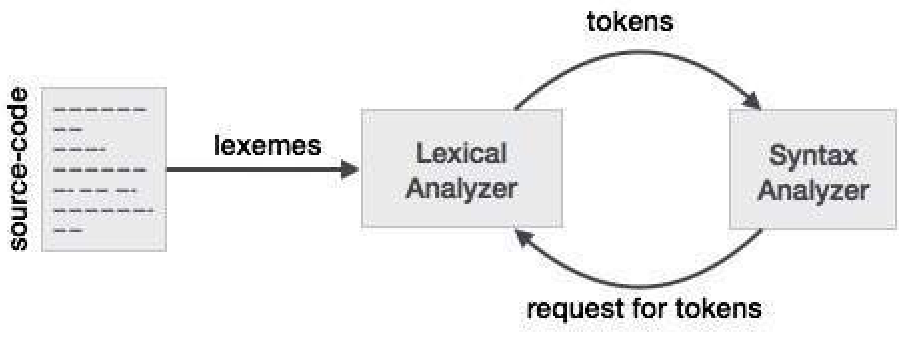
This can be looked at as the final stage of compilation. Its function is to write a code that the machine can understand. The output is assembler dependent.

Lexical Analysis

The first phase of a compiler is the lexical analysis. It takes a source program as the input and generates a sequence of tokens. A token is a basic component of source code. These tokens are inserted into a symbol table which maps the tokens to their types. The symbol often holds other information like memory location which may prove to be useful in the later phases.

Common token names are:

- identifiers: names the programmer chooses
- keywords: preexisting names in the programming language
- separators/punctuation: punctuation characters and paired-delimiters
- operators: symbols that operate on arguments and produce results
- literals: numeric, logical, textual, reference literals
- comments: line, block



Source Code

Source code:

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

      #include <string.h>

      #include <stdlib.h>
          //int searchInHash(int key, char *name) ;
          int top=-1;
          char a[100], err_id_str[20][10];
          int co_paranthesis=0,co_curly=0 , co_square=0 , co_comment=0 , co_quotes=0 , mul_line_comment=0 ;
          int cc_paranthesis=0, cc_curly=0 , cc_square=0 , cc_comment=0 , err_comment=0 ;
          int mismatch_paranthesis = 0 , mismatch_curly = 0 , mismatch_square = 0 , err_id=0 , no_of_errors=0;

%}
alpha [a-zA-Z]

digit [0-9]

keyword
"printf"|"scanf"|"int"|"float"|"if"|"break"|"case"|"char"|"const"|"main"|"default"|"do"|"double"|"else"|"enum"|"floa
t"|"for"|"goto"|"if"|"int"|"long"|"register"|"return"|"short"|"signed"|"sizeof"|"static"|"struct"|"switch"|"typedef"|"un
ion"|"unsigned"|"void"|"volatile"|"while"

operator "+"|"-"|"*"|"/"|"%"|"++"|"--"|"="|"<"|">"|"&&"|"||"|"|"|"&"|"!"

spl_symbols ";"

quotes ["]
%%
\n

#include[ ]?<{alpha}*.h> {
          printf("\n\n\t%-30s is a Pre-processor directive\n",yytext);

}
,
" "
\t

"void main()"|"int main()" {
 printf("\n\t%-30s is the main function\n",yytext);
```

```
}

{alpha}({alpha}|{digit}|[_])*\(({alpha}|{digit}|[_]|[ ])*\) {

 printf("\n\t%-30s is a user defined function\n",yytext);
}

"//".* {
            printf("\n\t%-30s is single line comment\n",yytext);
}

"/*"([^*]|\*+[^*/])*+"*/" {
     printf("\n\t%-30s is multi line comment\n",yytext);
}

{keyword} {

            printf("\n\t%-30s is a Keyword\n",yytext);
            int key=hash_func(yytext);
            if(searchInHash(key,yytext)==0)
            insertToHash(key,yytext,"keyword");

}

{spl_symbols} {
            printf("\n\t%-30s is a Special Symbol\n",yytext);
            int key=hash_func(yytext);
            if(searchInHash(key,yytext)==0)
            insertToHash(key,yytext,"special symbol");

}

{digit}+{alpha}+|{alpha}*{digit}*@{alpha}*{digit}*|%{alpha}{alpha}+ {
            strcpy(err_id_str[err_id],yytext);
            err_id++;
}

"(" {
            printf("\n\t%-30s is a Special Symbol\n",yytext);
            co_paranthesis++;
            int key=hash_func(yytext);
            if(searchInHash(key,yytext)==0)
            insertToHash(key,yytext,"special symbol");
            top++;
            a[top]='(';
}

"{" {
            printf("\n\t%-30s is a Special Symbol\n",yytext);
            co_curly++;
            int key=hash_func(yytext);
            if(searchInHash(key,yytext)==0)
            insertToHash(key,yytext,"special symbol");
            top++;
            a[top]='{';
}

"[" {
            printf("\n\t%-30s is a Special Symbol\n",yytext);
            co_square++;
            int key=hash_func(yytext);
            if(searchInHash(key,yytext)==0)
            insertToHash(key,yytext,"special symbol");
            top++;
            a[top]='[';
}

")" {
            printf("\n\t%-30s is a Special Symbol\n",yytext);
            cc_paranthesis++;
            if(a[top]!='(')
              {
         mismatch_paranthesis=1;
            }
            else
              {top--;}
            int key=hash_func(yytext);
            if(searchInHash(key,yytext)==0)
            insertToHash(key,yytext,"special symbol");
}


"}" {
            printf("\n\t%-30s is a Special Symbol\n",yytext);
            cc_curly++;
            if(a[top]!='{')
              {
         mismatch_curly=1;
            }
            else
              {top--;}
            int key=hash_func(yytext);
            if(searchInHash(key,yytext)==0)
            insertToHash(key,yytext,"special symbol");
}

"]" {
            printf("\n\t%-30s is a Special Symbol\n",yytext);
            cc_square++;
            if(a[top]!='[')
              {
         mismatch_square=1;
            }
            else
              {top--;}
            int key=hash_func(yytext);
            if(searchInHash(key,yytext)==0)
            insertToHash(key,yytext,"special symbol");
}

{quotes} {
            printf("\n\t%-30s is a Special Symbol\n",yytext);
            co_quotes++;
            int key=hash_func(yytext);
            if(searchInHash(key,yytext)==0)
            insertToHash(key,yytext,"special symbol");

}

"/*" {  co_comment++;
            if(mul_line_comment==0)
            {mul_line_comment++;}
            else
            {err_comment=1;}

}

"*/" {  cc_comment++;
            if(mul_line_comment==1)
            {mul_line_comment--;}
            else
            {err_comment=1;}
}

{operator} {
            printf("\n\t%-30s is an Operator\n",yytext);
            int key=hash_func(yytext);
            if(searchInHash(key,yytext)==0)
            insertToHash(key,yytext,"operator");
            }

{alpha}({alpha}|{digit})* {
            printf("\n\t%-30s is an Identifier\n",yytext);
            int key=hash_func(yytext);
            if(searchInHash(key,yytext)==0)
            insertToHash(key,yytext,"identifier");

}
{digit} {
            printf("\n\t%-30s is a Constant\n",yytext);
            int key=hash_func(yytext);
            if(searchInHash(key,yytext)==0)
            insertToHash(key,yytext,"constant");

}

\"(\\.|[^"\\])*\" {
            printf("\n\t%-30s is a String\n",yytext);
            int key=hash_func(yytext);
            if(searchInHash(key,yytext)==0)
            insertToHash(key,yytext,"string");

}
%%


/*-------------------------------------------------------------*/


         struct hash *hashTable = NULL;

    int eleCount = 0;

     struct node {
        char name[20];
            char type[20];
            int key;
        struct node *next;
     };



    struct hash {

        struct node *head;

        int count;

    };

         struct node * createNode(int key, char *name, char *type) {

        struct node *newnode;

        newnode = (struct node *) malloc(sizeof(struct node));

            newnode->key=key;
        strcpy(newnode->name, name);
            strcpy(newnode->type, type);

        newnode->next = NULL;

        return newnode;

    }



    void insertToHash(int key, char *name, char *type) {

        int hashIndex = key % eleCount;
        struct node *newnode = createNode(key, name, type);



  if (!hashTable[hashIndex].head) {

            hashTable[hashIndex].head = newnode;

            hashTable[hashIndex].count = 1;

            return;

        }
   newnode->next = (hashTable[hashIndex].head);

   hashTable[hashIndex].head = newnode;

        hashTable[hashIndex].count++;

        return;
}

      int searchInHash(int key, char *name) {

        int hashIndex = key % eleCount, flag = 0;


        struct node *myNode;

        myNode = hashTable[hashIndex].head;

        if (!myNode) {


             return flag;

        }

        while (myNode != NULL) {

            if (strcmp(myNode->name,name)==0) {


                flag = 1;

                break;

            }

            myNode = myNode->next;
```

```
    }

    return flag;
}


void display() {
    struct node *myNode;
    int i;
         printf("\n\n\n");
        printf("\t|-----------------------------------------------------------------------|\n");
         printf("\t|                               |                          |\n");
        printf("\t|       Name                    |     Type                 |\n");
        printf("\t|-----------------------------------------------------------------------|\n");
        printf("\t|                               |                          |\n");

    for (i = 0; i < eleCount; i++) {
        if (hashTable[i].count == 0)
            continue;
        myNode = hashTable[i].head;
        if (!myNode)
            continue;


        while (myNode != NULL) {
            printf("\t|\t%-30s", myNode->name);
            printf("\t|\t%-15s\n", myNode->type);
        printf("\t|-----------------------------------------------------------------------|\n");


            myNode = myNode->next;
        }
        }

        return;

    }


            int hash_func(char val[20])
            {
            int sum=0;
            for(int i=0; i<strlen(val);i++)
            {
                        sum+=val[i];
            }
            sum%=50;

            return sum;
            }


void main(int argc[], char **argv[])
{

            //yyin=fopen("test.c","r");
            int n;
            eleCount=50;
             hashTable = (struct hash *) calloc(50, sizeof(struct hash));

            yylex();

            printf("\n\n\n\n\t\t\t\t\t   ERRORS");

printf("\n\t_______________________________________________________________________________\n");

            if(co_paranthesis!=cc_paranthesis)
            {no_of_errors++;
            printf("\n\t%d)---------------------UNMATCHED ( !!!-------------\n",no_of_errors);}
            if(co_curly!=cc_curly)
            {no_of_errors++;
            printf("\n\t%d)---------------------UNMATCHED { !!!-------------\n",no_of_errors);}
            if(co_square!=cc_square)
            {no_of_errors++;
            printf("\n\t%d)---------------------UNMATCHED [ !!!-------------\n",no_of_errors);}
            if(co_quotes%2!=0)
            {no_of_errors++;
            printf("\n\t%d)---------------------UNMATCHED QUOTES !!!-------------\n",no_of_errors);}
            if(co_comment!=cc_comment)
            {no_of_errors++;
            printf("\n\t%d)---------------------UNMATCHED                                    MULTI_LINE
COMMENT!!!-------------\n",no_of_errors);}
            if(err_comment==1)
            {no_of_errors++;
```

```
	printf("\n\t%d)---------------------NESTED MULTI_LINE COMMENT!!!-------------\n",no_of_errors);}
	if(mismatch_paranthesis==1)
	{no_of_errors++;
	printf("\n\t%d)---------------------MISMATCHED PARANTHESES!!!-------------\n",no_of_errors);}
	if(mismatch_curly==1)
	{no_of_errors++;
	printf("\n\t%d)---------------------MISMATCHED CURLY BRACES!!!-------------\n",no_of_errors);}
	if(mismatch_square==1)
	{no_of_errors++;
	printf("\n\t%d)---------------------MISMATCHED SQUARE BRACES!!!-------------\n",no_of_errors);}
	if(err_id>0)
	{
	for (int i =0 ; i< err_id ; i++)
	{
		no_of_errors++;
		printf("\n\t%d)---------------------INVALID                                        IDENTIFIER!!![-
%s]-------------\n",no_of_errors,err_id_str[i]);
	}
	}
	if(no_of_errors==0)
		printf("\n\tNO LEXICAL ERRORS!");

	printf("\n\n\n\n\t\t\t\t\tSYMBOL TABLE");

printf("\n\t_________________________________________________________________________");


	display();

	printf("%d%d",co_comment, cc_comment);

}


int yywrap()
{
	return 1;
}
```

Regular Expressions

alpha [a-zA-Z]

digit [0-9]

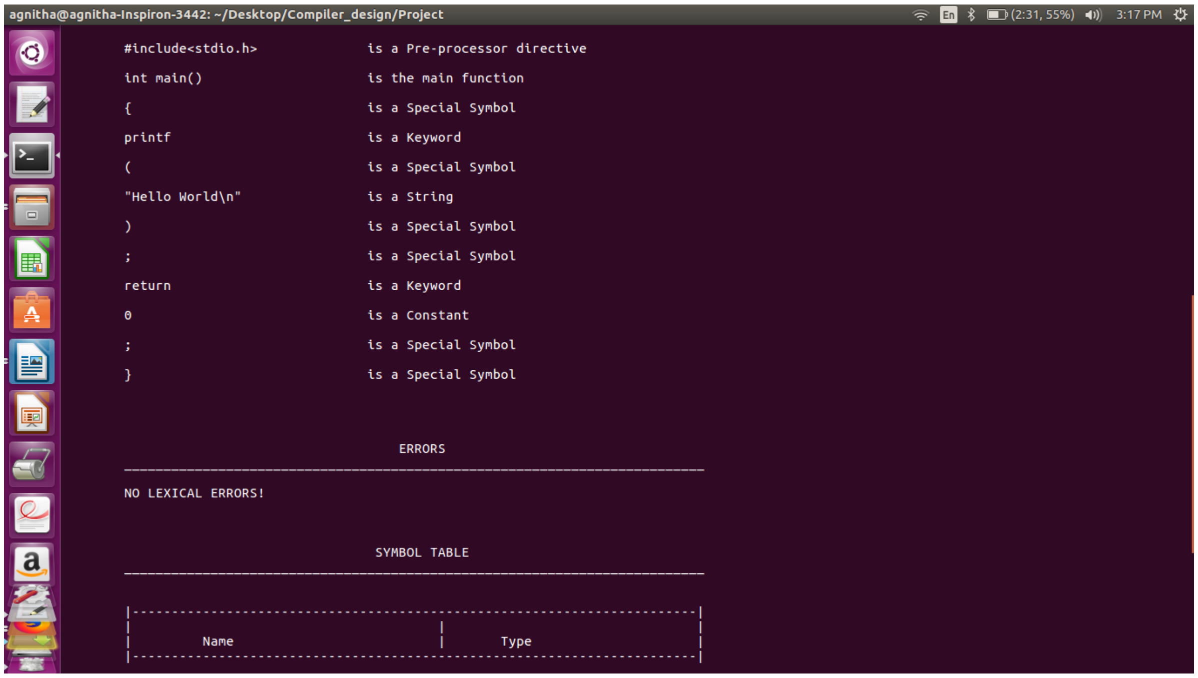
Pre-processor directive	#include[ ]?<{alpha}*.h>
User defined function	{alpha}({alpha}|{digit}|[_])*\\(({alpha}|{digit}|[_]|[ ])*\\)
Single line comment	"//".*
Identifier	{alpha}({alpha}|{digit})*
String	\\"(\\\\.|[^"\\\\])*\\"
Invalid-Identifier %	{digit}+{alpha}+|{alpha}*{digit}*@{alpha}*{digit}*| {alpha}{alpha}+

Output

Test Case-1

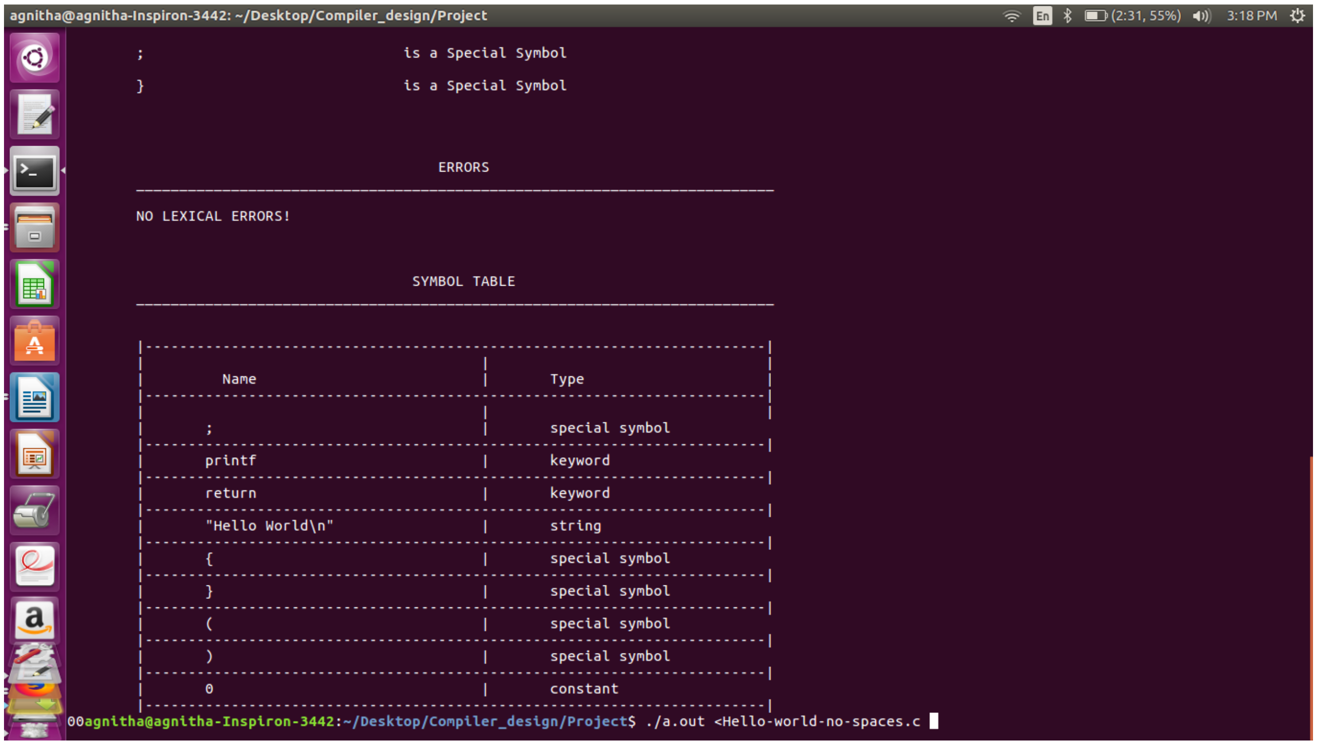
Input:

```
#include<stdio.h>
int main()
{
        printf("Hello World\n");
        return 0;
}
```



agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project

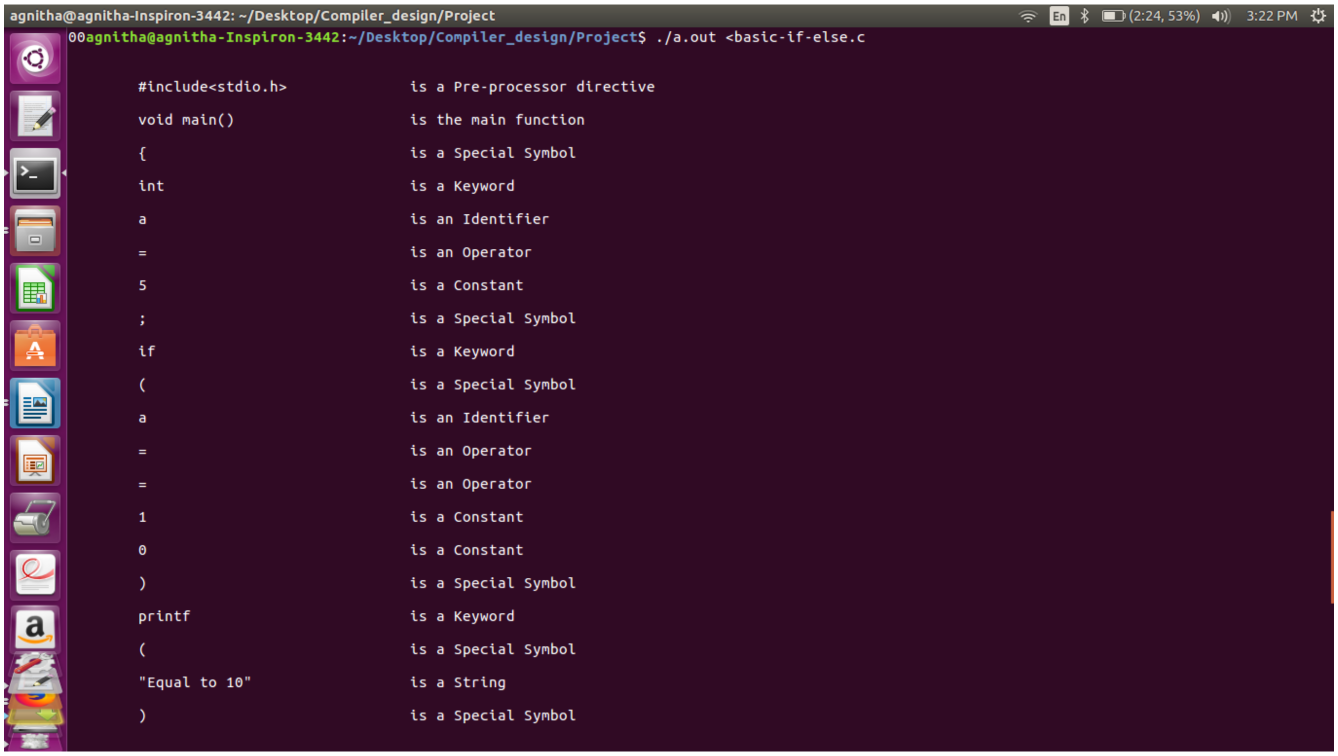
```
#include<stdio.h>               is a Pre-processor directive
int main()                      is the main function
{                               is a Special Symbol
printf                          is a Keyword
(                               is a Special Symbol
"Hello World\n"                 is a String
)                               is a Special Symbol
;                               is a Special Symbol
return                          is a Keyword
0                               is a Constant
;                               is a Special Symbol
}                               is a Special Symbol


                                    ERRORS
---------------------------------------------------------------------------
NO LEXICAL ERRORS!


                                 SYMBOL TABLE
---------------------------------------------------------------------------

|-------------------------------------------------------------------------|
|                                        |                                |
|         Name                           |         Type                   |
|-------------------------------------------------------------------------|
```



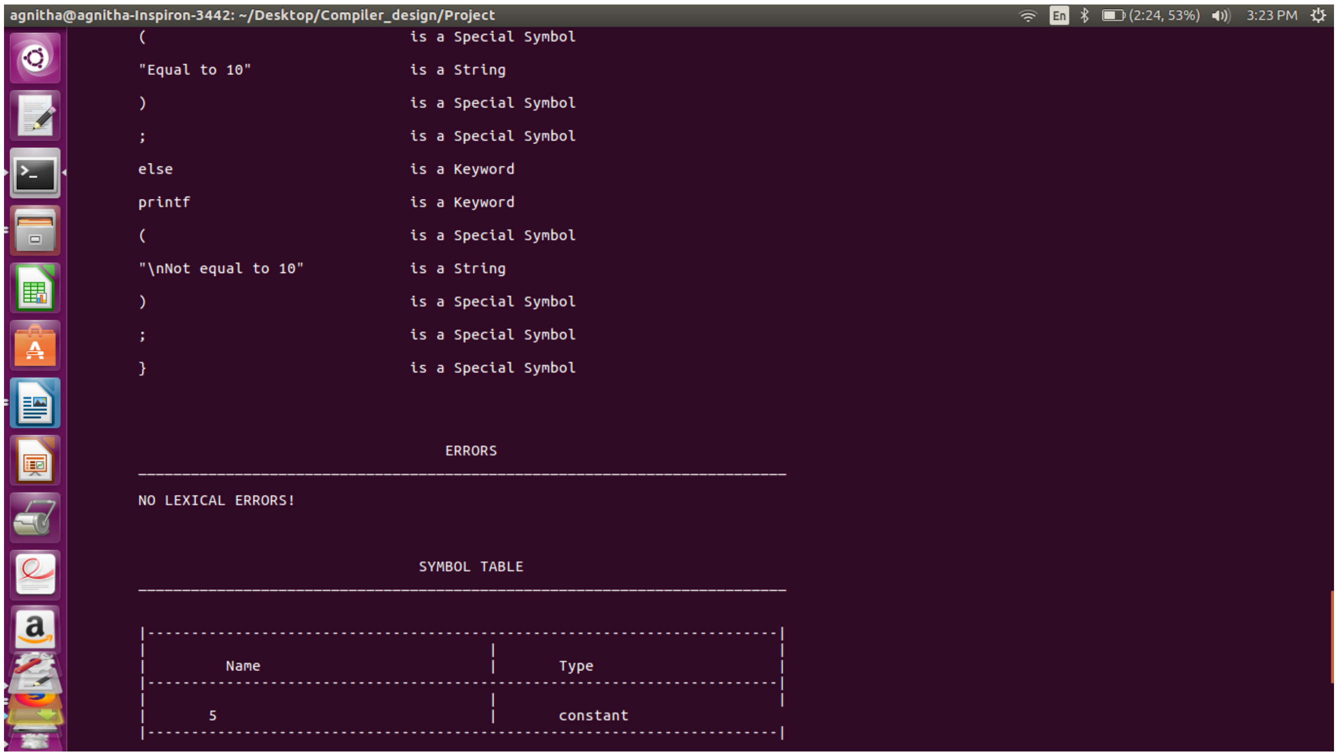
```
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project

        ;                               is a Special Symbol

        }                               is a Special Symbol



                                            ERRORS
        ---------------------------------------------------------------------------

        NO LEXICAL ERRORS!


                                         SYMBOL TABLE
        ---------------------------------------------------------------------------


        |-------------------------------------------------------------------------|
        |                                        |                                |
        |         Name                           |       Type                     |
        |-------------------------------------------------------------------------|
        |                                        |                                |
        |       ;                                |       special symbol           |
        |-------------------------------------------------------------------------|
        |       printf                           |       keyword                  |
        |-------------------------------------------------------------------------|
        |       return                           |       keyword                  |
        |-------------------------------------------------------------------------|
        |       "Hello World\n"                  |       string                   |
        |-------------------------------------------------------------------------|
        |       {                                |       special symbol           |
        |-------------------------------------------------------------------------|
        |       }                                |       special symbol           |
        |-------------------------------------------------------------------------|
        |       (                                |       special symbol           |
        |-------------------------------------------------------------------------|
        |       )                                |       special symbol           |
        |-------------------------------------------------------------------------|
        |       0                                |       constant                 |
        |-------------------------------------------------------------------------|
00agnitha@agnitha-Inspiron-3442:~/Desktop/Compiler_design/Project$ ./a.out <Hello-world-no-spaces.c
```

Test Case-2

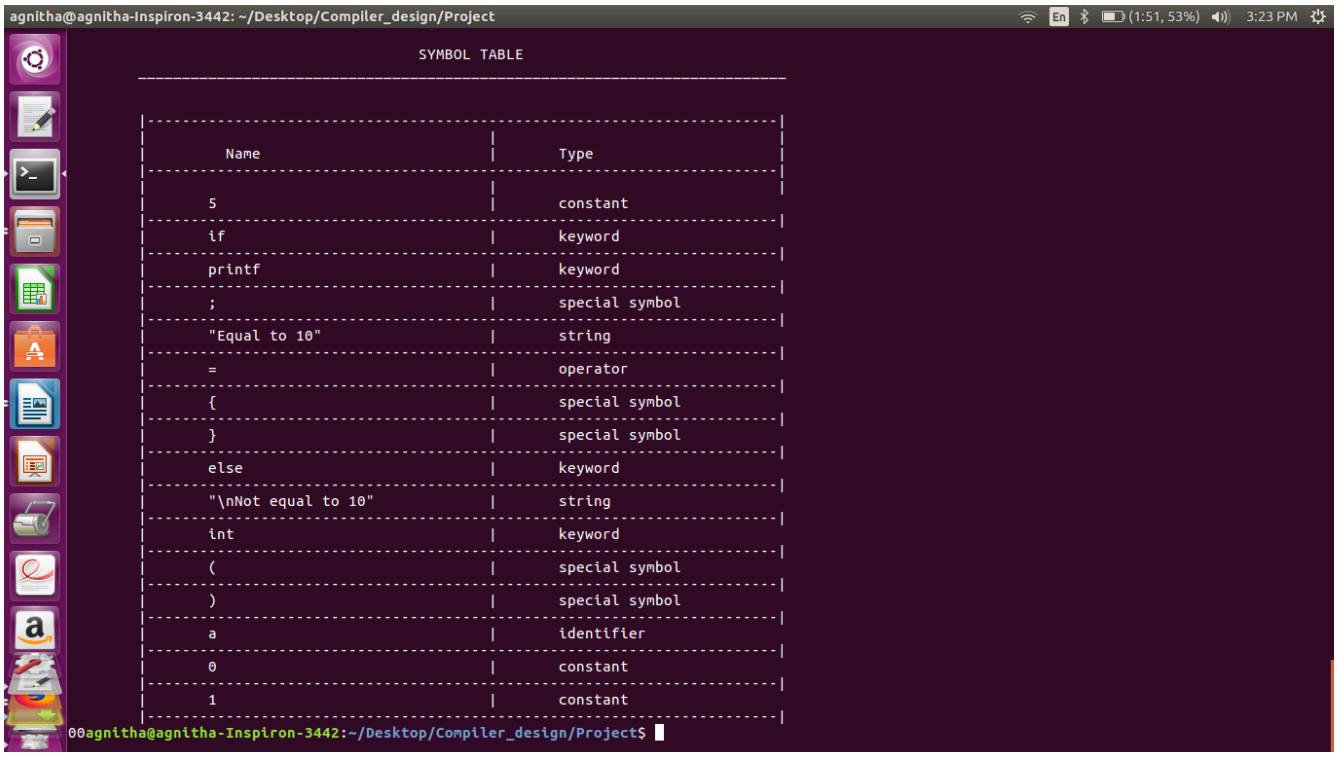
```
#include<stdio.h>

void main()
{
            int a=5;
            if(a==10)
              printf("Equal to 10");

            else
              printf("\nNot equal to 10");

}
```



```
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project
00agnitha@agnitha-Inspiron-3442:~/Desktop/Compiler_design/Project$ ./a.out <basic-if-else.c

        #include<stdio.h>               is a Pre-processor directive
        void main()                     is the main function
        {                               is a Special Symbol
        int                             is a Keyword
        a                               is an Identifier
        =                               is an Operator
        5                               is a Constant
        ;                               is a Special Symbol
        if                              is a Keyword
        (                               is a Special Symbol
        a                               is an Identifier
        =                               is an Operator
        =                               is an Operator
        1                               is a Constant
        0                               is a Constant
        )                               is a Special Symbol
        printf                          is a Keyword
        (                               is a Special Symbol
        "Equal to 10"                   is a String
        )                               is a Special Symbol
```



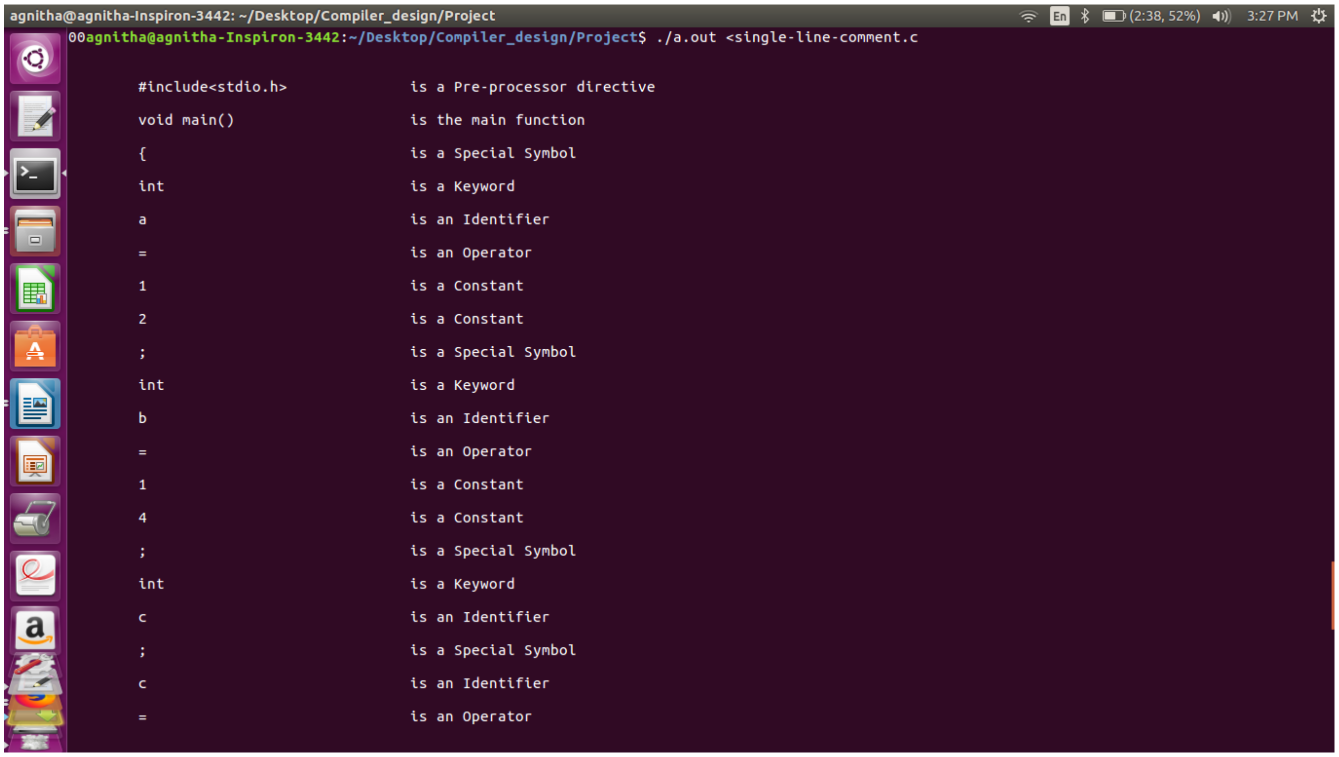
```
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project
        (                              is a Special Symbol
        "Equal to 10"                  is a String
        )                              is a Special Symbol
        ;                              is a Special Symbol
        else                           is a Keyword
        printf                         is a Keyword
        (                              is a Special Symbol
        "\nNot equal to 10"            is a String
        )                              is a Special Symbol
        ;                              is a Special Symbol
        }                              is a Special Symbol


                                           ERRORS
        --------------------------------------------------------------------------
        NO LEXICAL ERRORS!


                                        SYMBOL TABLE
        --------------------------------------------------------------------------

        |-------------------------------------------------------------------------|
        |                                       |                                 |
        |        Name                           |        Type                     |
        |-------------------------------------------------------------------------|
        |                                       |                                 |
        |       5                               |        constant                 |
        |-------------------------------------------------------------------------|
```



```
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project
                                        SYMBOL TABLE
        --------------------------------------------------------------------------

        |-------------------------------------------------------------------------|
        |                                       |                                 |
        |        Name                           |        Type                     |
        |-------------------------------------------------------------------------|
        |                                       |                                 |
        |       5                               |        constant                 |
        |-------------------------------------------------------------------------|
        |       if                              |        keyword                  |
        |-------------------------------------------------------------------------|
        |       printf                          |        keyword                  |
        |-------------------------------------------------------------------------|
        |       ;                               |        special symbol           |
        |-------------------------------------------------------------------------|
        |       "Equal to 10"                   |        string                   |
        |-------------------------------------------------------------------------|
        |       =                               |        operator                 |
        |-------------------------------------------------------------------------|
        |       {                               |        special symbol           |
        |-------------------------------------------------------------------------|
        |       }                               |        special symbol           |
        |-------------------------------------------------------------------------|
        |       else                            |        keyword                  |
        |-------------------------------------------------------------------------|
        |       "\nNot equal to 10"             |        string                   |
        |-------------------------------------------------------------------------|
        |       int                             |        keyword                  |
        |-------------------------------------------------------------------------|
        |       (                               |        special symbol           |
        |-------------------------------------------------------------------------|
        |       )                               |        special symbol           |
        |-------------------------------------------------------------------------|
        |       a                               |        identifier               |
        |-------------------------------------------------------------------------|
        |       0                               |        constant                 |
        |-------------------------------------------------------------------------|
        |       1                               |        constant                 |
        |-------------------------------------------------------------------------|
00agnitha@agnitha-Inspiron-3442:~/Desktop/Compiler_design/Project$
```

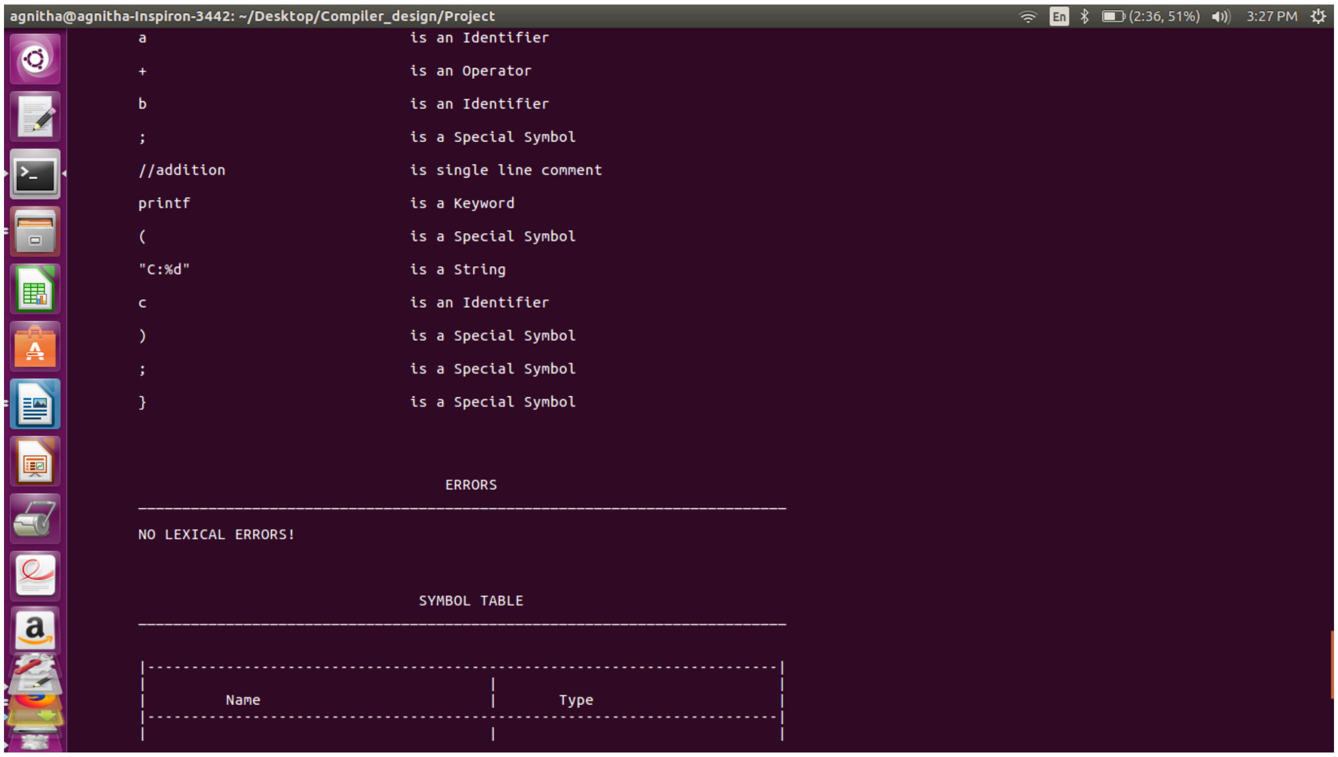
Test Case-3

```
#include<stdio.h>

void main()
{

		int a=12;
		int b=14;
		int c;
		c=a+b;  //addition
		printf("C:%d",c);
}
```

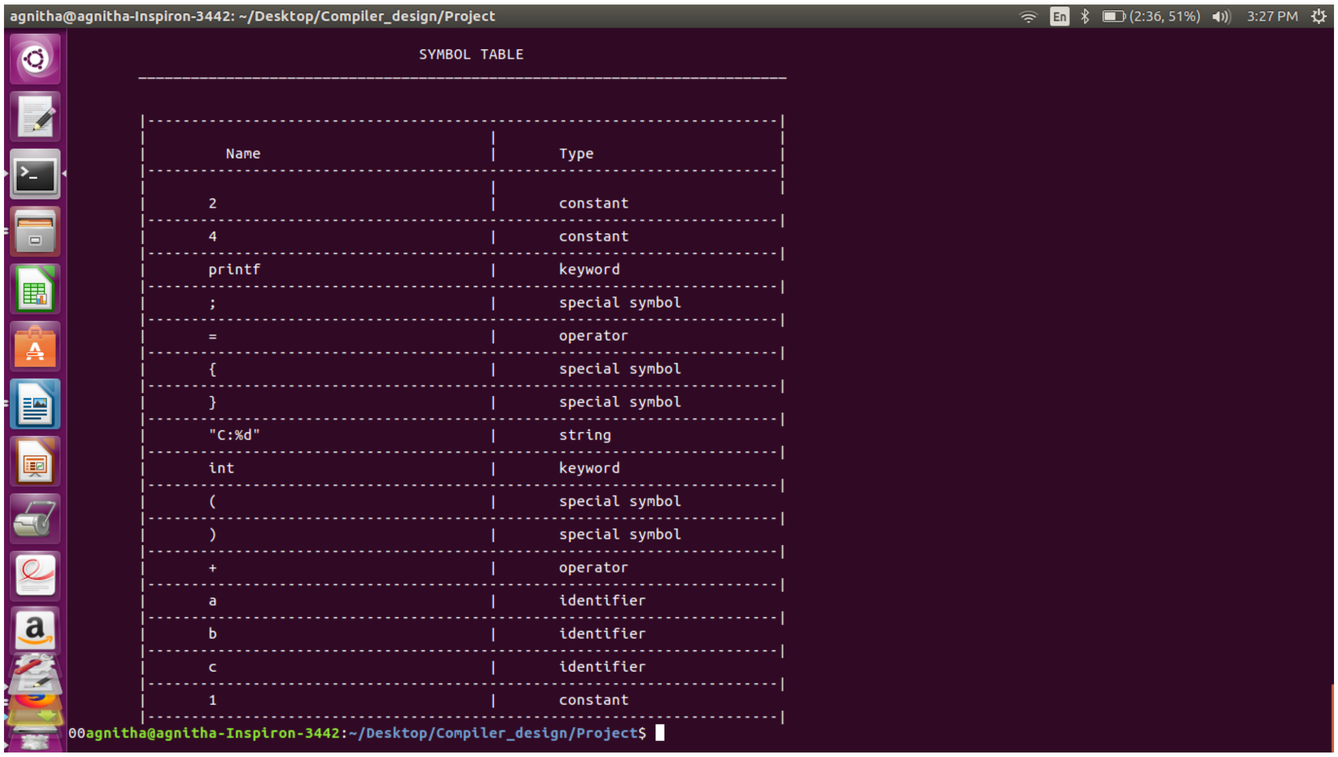


agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project
En
(2:38, 52%)
3:27 PM

```
00agnitha@agnitha-Inspiron-3442:~/Desktop/Compiler_design/Project$ ./a.out <single-line-comment.c

        #include<stdio.h>               is a Pre-processor directive
        void main()                     is the main function
        {                               is a Special Symbol
        int                             is a Keyword
        a                               is an Identifier
        =                               is an Operator
        1                               is a Constant
        2                               is a Constant
        ;                               is a Special Symbol
        int                             is a Keyword
        b                               is an Identifier
        =                               is an Operator
        1                               is a Constant
        4                               is a Constant
        ;                               is a Special Symbol
        int                             is a Keyword
        c                               is an Identifier
        ;                               is a Special Symbol
        c                               is an Identifier
        =                               is an Operator
```



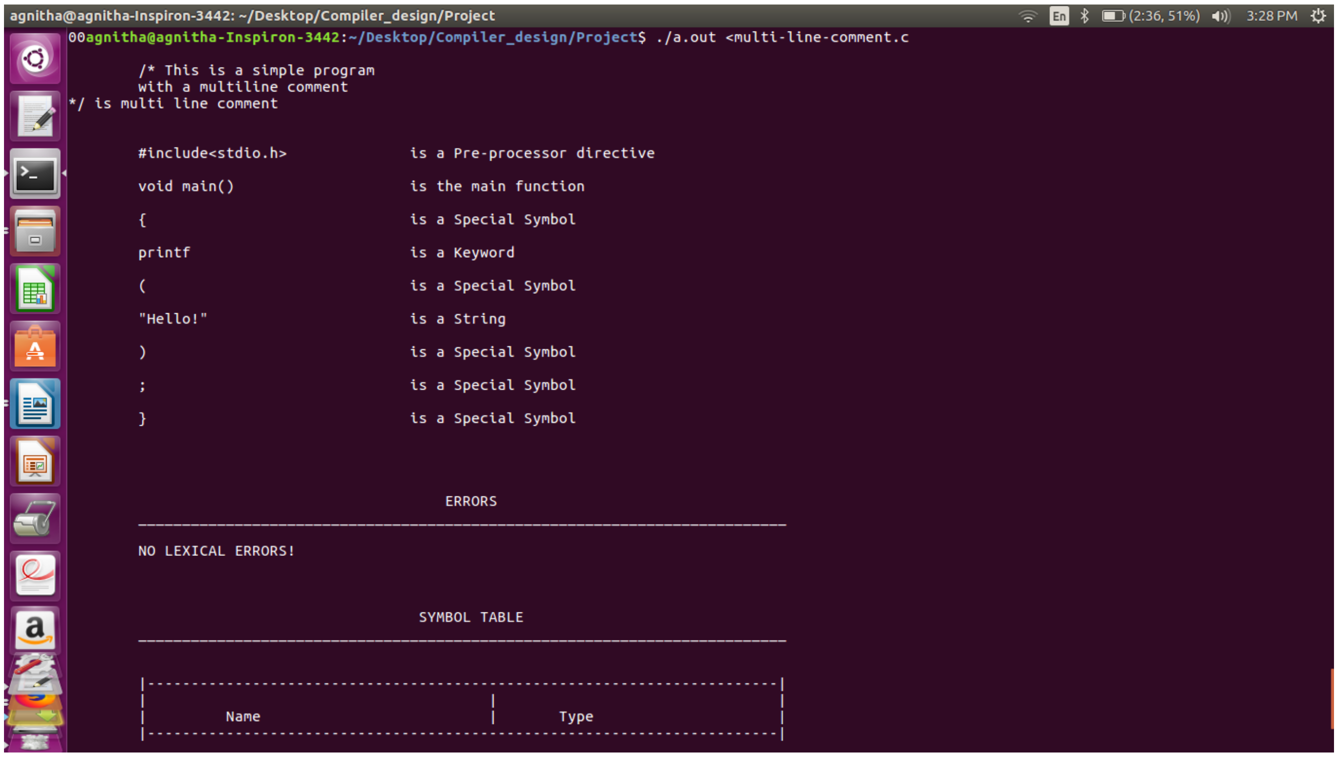
```
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project                (2:36, 51%)   3:27 PM
a                               is an Identifier
+                               is an Operator
b                               is an Identifier
;                               is a Special Symbol
//addition                      is single line comment
printf                          is a Keyword
(                               is a Special Symbol
"C:%d"                          is a String
c                               is an Identifier
)                               is a Special Symbol
;                               is a Special Symbol
}                               is a Special Symbol


                                    ERRORS
----------------------------------------------------------------------------
NO LEXICAL ERRORS!


                                 SYMBOL TABLE
----------------------------------------------------------------------------

|-------------------------------------------------------------------------|
|                                       |                                 |
|         Name                          |       Type                      |
|-------------------------------------------------------------------------|
|                                       |                                 |
```



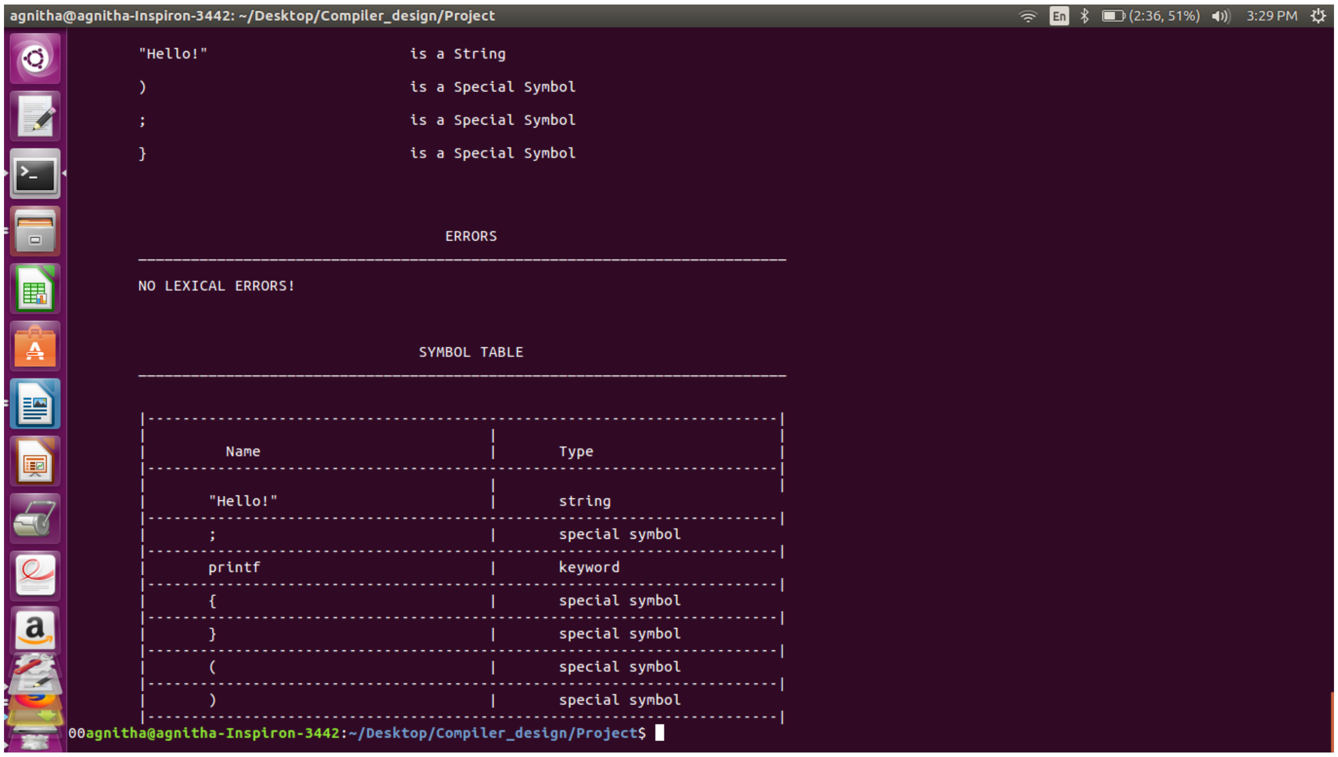
```
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project                (2:36, 51%)   3:27 PM
                                 SYMBOL TABLE
----------------------------------------------------------------------------

|-------------------------------------------------------------------------|
|                                       |                                 |
|         Name                          |       Type                      |
|-------------------------------------------------------------------------|
|                                       |                                 |
|       2                               |       constant
|-------------------------------------------------------------------------|
|       4                               |       constant
|-------------------------------------------------------------------------|
|       printf                          |       keyword
|-------------------------------------------------------------------------|
|       ;                               |       special symbol
|-------------------------------------------------------------------------|
|       =                               |       operator
|-------------------------------------------------------------------------|
|       {                               |       special symbol
|-------------------------------------------------------------------------|
|       }                               |       special symbol
|-------------------------------------------------------------------------|
|       "C:%d"                          |       string
|-------------------------------------------------------------------------|
|       int                             |       keyword
|-------------------------------------------------------------------------|
|       (                               |       special symbol
|-------------------------------------------------------------------------|
|       )                               |       special symbol
|-------------------------------------------------------------------------|
|       +                               |       operator
|-------------------------------------------------------------------------|
|       a                               |       identifier
|-------------------------------------------------------------------------|
|       b                               |       identifier
|-------------------------------------------------------------------------|
|       c                               |       identifier
|-------------------------------------------------------------------------|
|       1                               |       constant
|-------------------------------------------------------------------------|
00agnitha@agnitha-Inspiron-3442:~/Desktop/Compiler_design/Project$
```

Test Case-4

```
/* This is a simple program
                with a multiline comment
*/


#include<stdio.h>

void main()
{
                printf("Hello!");
}
```



```
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project
00agnitha@agnitha-Inspiron-3442:~/Desktop/Compiler_design/Project$ ./a.out <multi-line-comment.c

        /* This is a simple program
        with a multiline comment
*/ is multi line comment


        #include<stdio.h>               is a Pre-processor directive

        void main()                     is the main function

        {                               is a Special Symbol

        printf                          is a Keyword

        (                               is a Special Symbol

        "Hello!"                        is a String

        )                               is a Special Symbol

        ;                               is a Special Symbol

        }                               is a Special Symbol



                                            ERRORS
        ___________________________________________________________________________

        NO LEXICAL ERRORS!


                                         SYMBOL TABLE
        ___________________________________________________________________________


        |------------------------------------------------------------------------|
        |                                        |                               |
        |         Name                           |         Type                  |
        |------------------------------------------------------------------------|
```

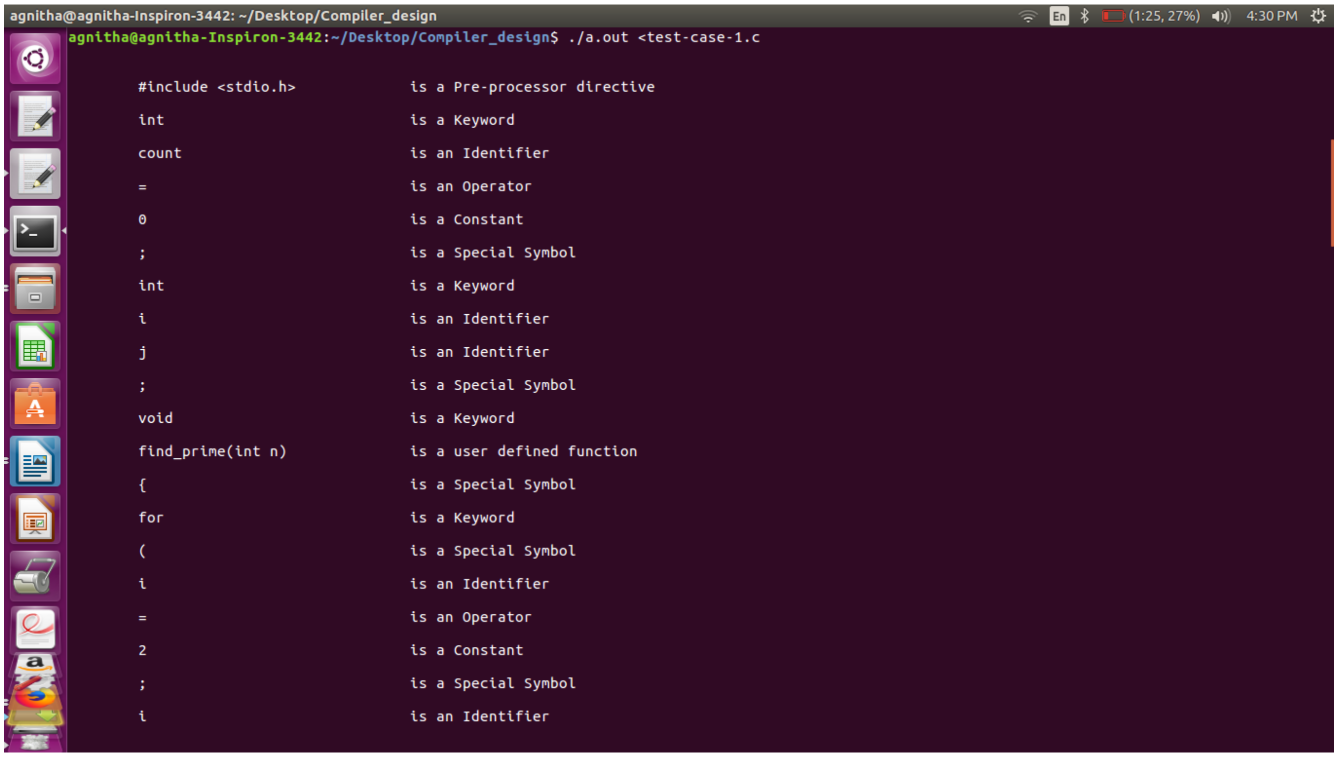


agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project

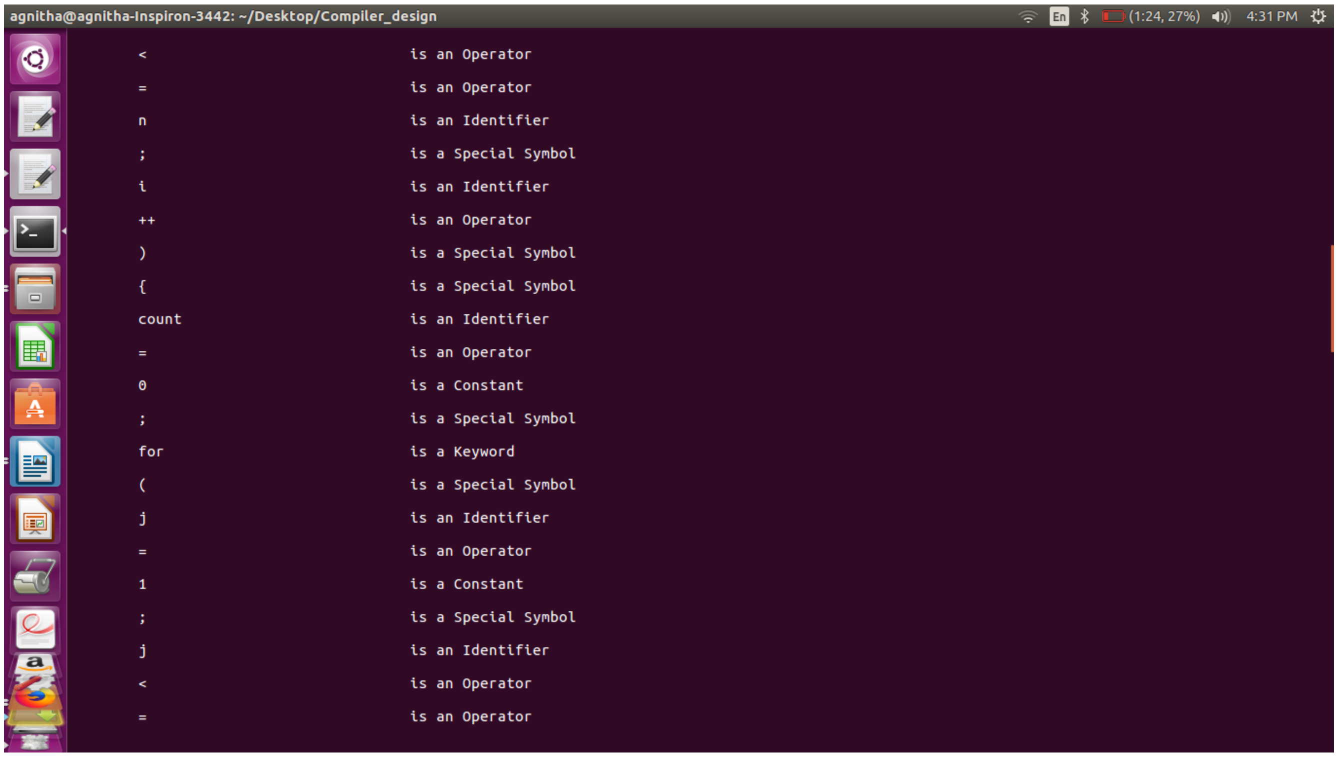
```
        "Hello!"                        is a String

        )                               is a Special Symbol

        ;                               is a Special Symbol

        }                               is a Special Symbol



                                    ERRORS
        ----------------------------------------------------------------------------

        NO LEXICAL ERRORS!


                                 SYMBOL TABLE
        ----------------------------------------------------------------------------


        |--------------------------------------------------------------------------|
        |                                       |                                  |
        |         Name                          |       Type                       |
        |--------------------------------------------------------------------------|
        |                                       |                                  |
        |       "Hello!"                        |       string                     |
        |--------------------------------------------------------------------------|
        |       ;                               |       special symbol             |
        |--------------------------------------------------------------------------|
        |       printf                          |       keyword                    |
        |--------------------------------------------------------------------------|
        |       {                               |       special symbol             |
        |--------------------------------------------------------------------------|
        |       }                               |       special symbol             |
        |--------------------------------------------------------------------------|
        |       (                               |       special symbol             |
        |--------------------------------------------------------------------------|
        |       )                               |       special symbol             |
        |--------------------------------------------------------------------------|
00agnitha@agnitha-Inspiron-3442:~/Desktop/Compiler_design/Project$ 
```

Test Case-5

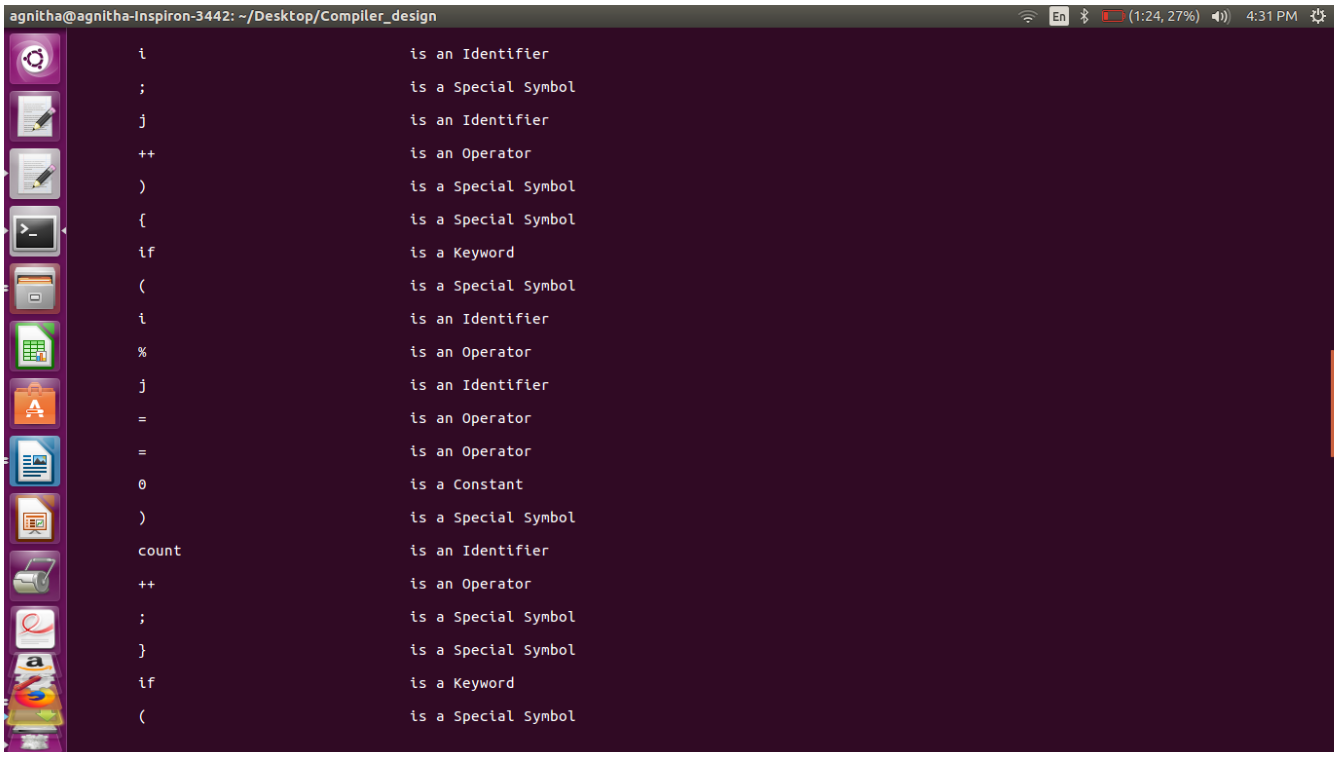
```
#include <stdio.h>

int count=0;
int i, j;

void find_prime(int n)
{
              for(i=2 ; i<=n ; i++)
             {
              count = 0;
              for(j=1 ; j<=i ; j++)
              {
                     if(i%j==0)
                            count++;
              }

              if(count == 2)
                     printf("\n %d is prime" , i);
              else
                     printf("\n %d is composite", i);

             }

}

void main()
{
             find_prime(10);
}
```



```
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design
agnitha@agnitha-Inspiron-3442:~/Desktop/Compiler_design$ ./a.out <test-case-1.c

        #include <stdio.h>              is a Pre-processor directive
        int                             is a Keyword
        count                           is an Identifier
        =                               is an Operator
        0                               is a Constant
        ;                               is a Special Symbol
        int                             is a Keyword
        i                               is an Identifier
        j                               is an Identifier
        ;                               is a Special Symbol
        void                            is a Keyword
        find_prime(int n)               is a user defined function
        {                               is a Special Symbol
        for                             is a Keyword
        (                               is a Special Symbol
        i                               is an Identifier
        =                               is an Operator
        2                               is a Constant
        ;                               is a Special Symbol
        i                               is an Identifier
```

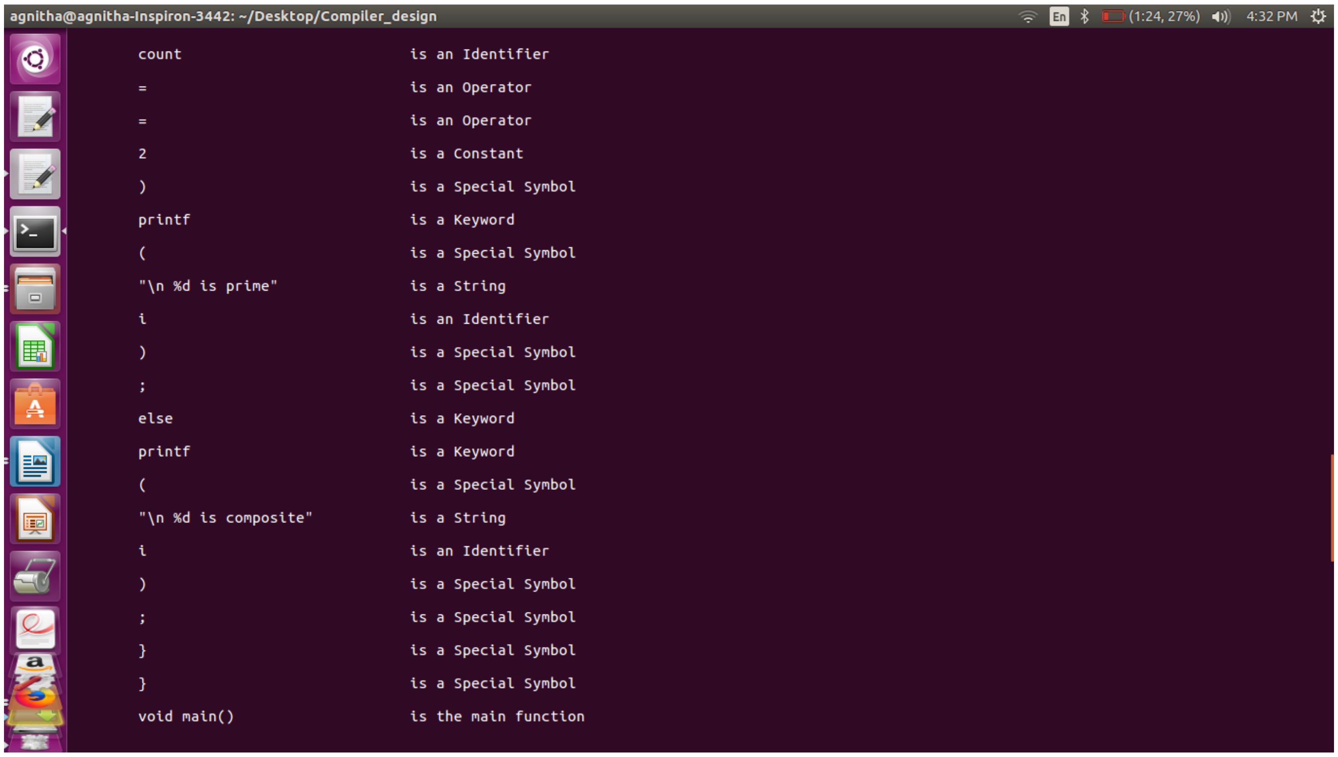


```
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design
        <                               is an Operator
        =                               is an Operator
        n                               is an Identifier
        ;                               is a Special Symbol
        i                               is an Identifier
        ++                              is an Operator
        )                               is a Special Symbol
        {                               is a Special Symbol
        count                           is an Identifier
        =                               is an Operator
        0                               is a Constant
        ;                               is a Special Symbol
        for                             is a Keyword
        (                               is a Special Symbol
        j                               is an Identifier
        =                               is an Operator
        1                               is a Constant
        ;                               is a Special Symbol
        j                               is an Identifier
        <                               is an Operator
        =                               is an Operator
```



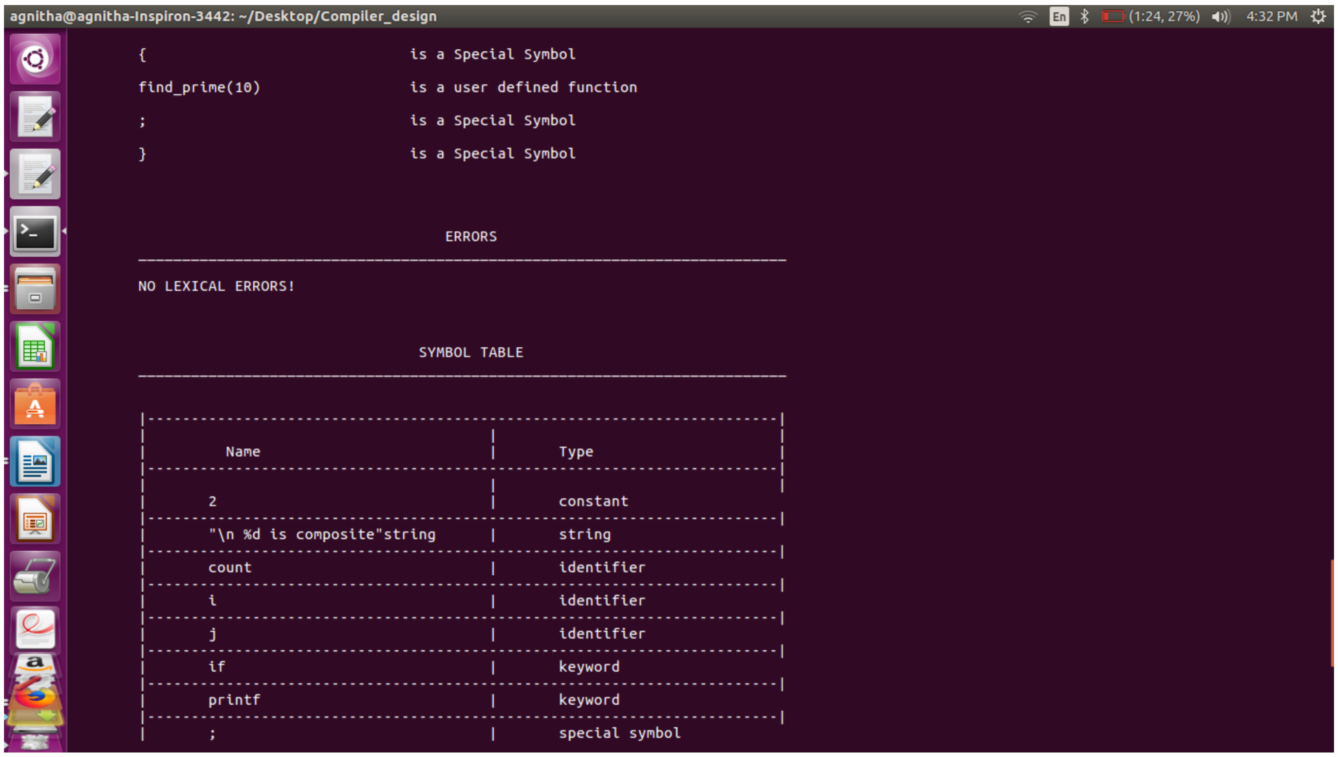
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design

```
i                              is an Identifier
;                              is a Special Symbol
j                              is an Identifier
++                             is an Operator
)                              is a Special Symbol
{                              is a Special Symbol
if                             is a Keyword
(                              is a Special Symbol
i                              is an Identifier
%                              is an Operator
j                              is an Identifier
=                              is an Operator
=                              is an Operator
0                              is a Constant
)                              is a Special Symbol
count                          is an Identifier
++                             is an Operator
;                              is a Special Symbol
}                              is a Special Symbol
if                             is a Keyword
(                              is a Special Symbol
```



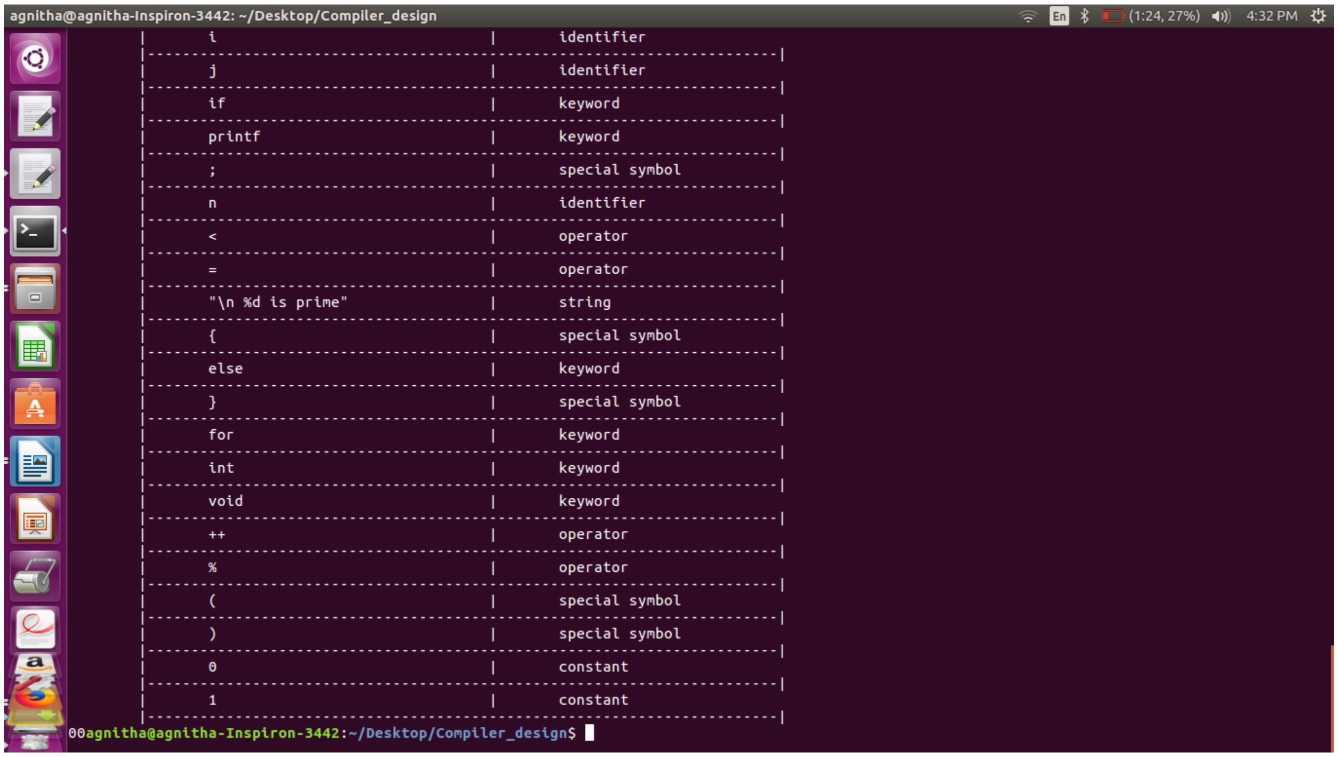
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design

```
count                          is an Identifier
=                              is an Operator
=                              is an Operator
2                              is a Constant
)                              is a Special Symbol
printf                         is a Keyword
(                              is a Special Symbol
"\n %d is prime"               is a String
i                              is an Identifier
)                              is a Special Symbol
;                              is a Special Symbol
else                           is a Keyword
printf                         is a Keyword
(                              is a Special Symbol
"\n %d is composite"           is a String
i                              is an Identifier
)                              is a Special Symbol
;                              is a Special Symbol
}                              is a Special Symbol
}                              is a Special Symbol
void main()                    is the main function
```



agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design

```
        {                               is a Special Symbol

        find_prime(10)                  is a user defined function

        ;                               is a Special Symbol

        }                               is a Special Symbol



                                            ERRORS
        ---------------------------------------------------------------------------

        NO LEXICAL ERRORS!


                                        SYMBOL TABLE
        ---------------------------------------------------------------------------


        |--------------------------------------------------------------------------|
        |                                       |                                  |
        |         Name                          |       Type                       |
        |---------------------------------------------------------------------------|
        |                                       |                                  |
        |       2                               |       constant                   |
        ---------------------------------------------------------------------------|
        |       "\n %d is composite"string      |       string                     |
        ---------------------------------------------------------------------------|
        |       count                           |       identifier                 |
        ---------------------------------------------------------------------------|
        |       i                               |       identifier                 |
        ---------------------------------------------------------------------------|
        |       j                               |       identifier                 |
        ---------------------------------------------------------------------------|
        |       if                              |       keyword                    |
        ---------------------------------------------------------------------------|
        |       printf                          |       keyword                    |
        ---------------------------------------------------------------------------|
        |       ;                               |       special symbol             |
```

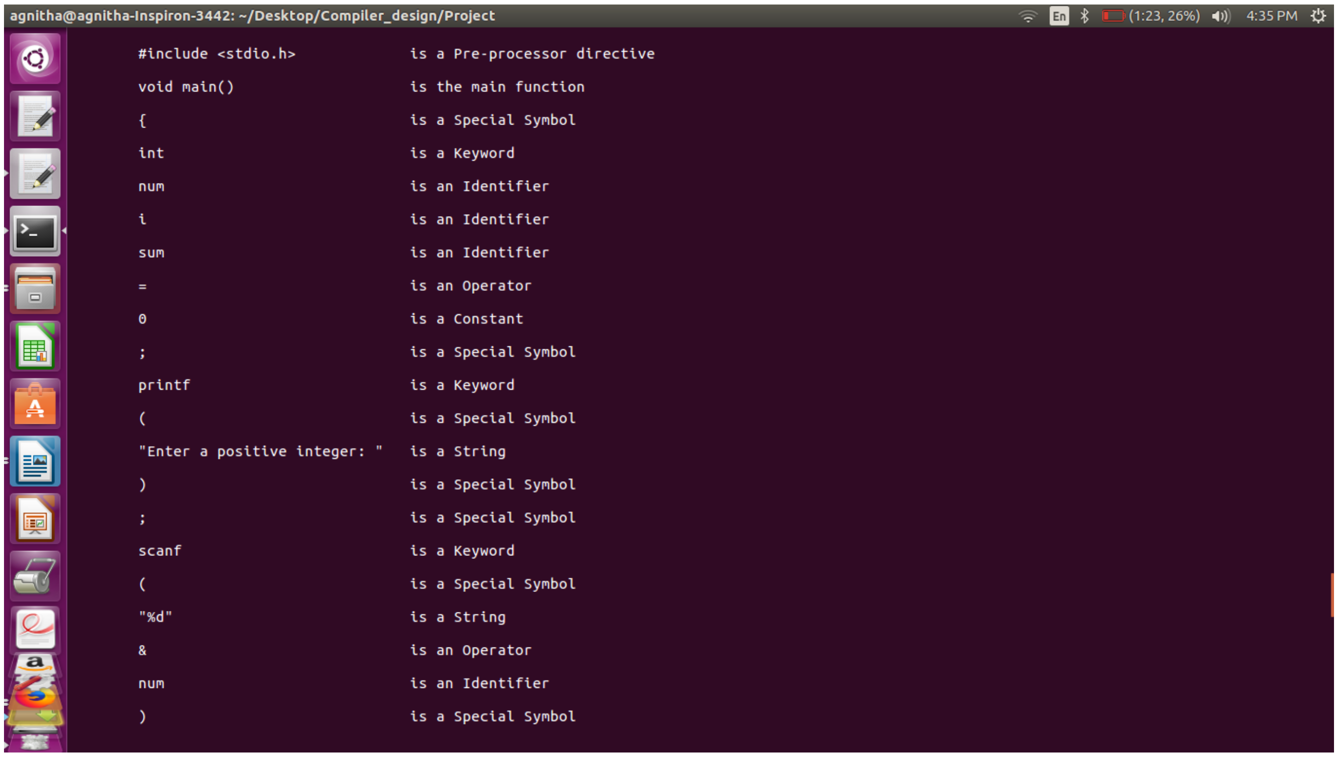


agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design

```
        |       i                               |       identifier                 |
        ---------------------------------------------------------------------------|
        |       j                               |       identifier                 |
        ---------------------------------------------------------------------------|
        |       if                              |       keyword                    |
        ---------------------------------------------------------------------------|
        |       printf                          |       keyword                    |
        ---------------------------------------------------------------------------|
        |       ;                               |       special symbol             |
        ---------------------------------------------------------------------------|
        |       n                               |       identifier                 |
        ---------------------------------------------------------------------------|
        |       <                               |       operator                   |
        ---------------------------------------------------------------------------|
        |       =                               |       operator                   |
        ---------------------------------------------------------------------------|
        |       "\n %d is prime"                |       string                     |
        ---------------------------------------------------------------------------|
        |       {                               |       special symbol             |
        ---------------------------------------------------------------------------|
        |       else                            |       keyword                    |
        ---------------------------------------------------------------------------|
        |       }                               |       special symbol             |
        ---------------------------------------------------------------------------|
        |       for                             |       keyword                    |
        ---------------------------------------------------------------------------|
        |       int                             |       keyword                    |
        ---------------------------------------------------------------------------|
        |       void                            |       keyword                    |
        ---------------------------------------------------------------------------|
        |       ++                              |       operator                   |
        ---------------------------------------------------------------------------|
        |       %                               |       operator                   |
        ---------------------------------------------------------------------------|
        |       (                               |       special symbol             |
        ---------------------------------------------------------------------------|
        |       )                               |       special symbol             |
        ---------------------------------------------------------------------------|
        |       0                               |       constant                   |
        ---------------------------------------------------------------------------|
        |       1                               |       constant                   |
        ---------------------------------------------------------------------------|
00agnitha@agnitha-Inspiron-3442:~/Desktop/Compiler_design$
```

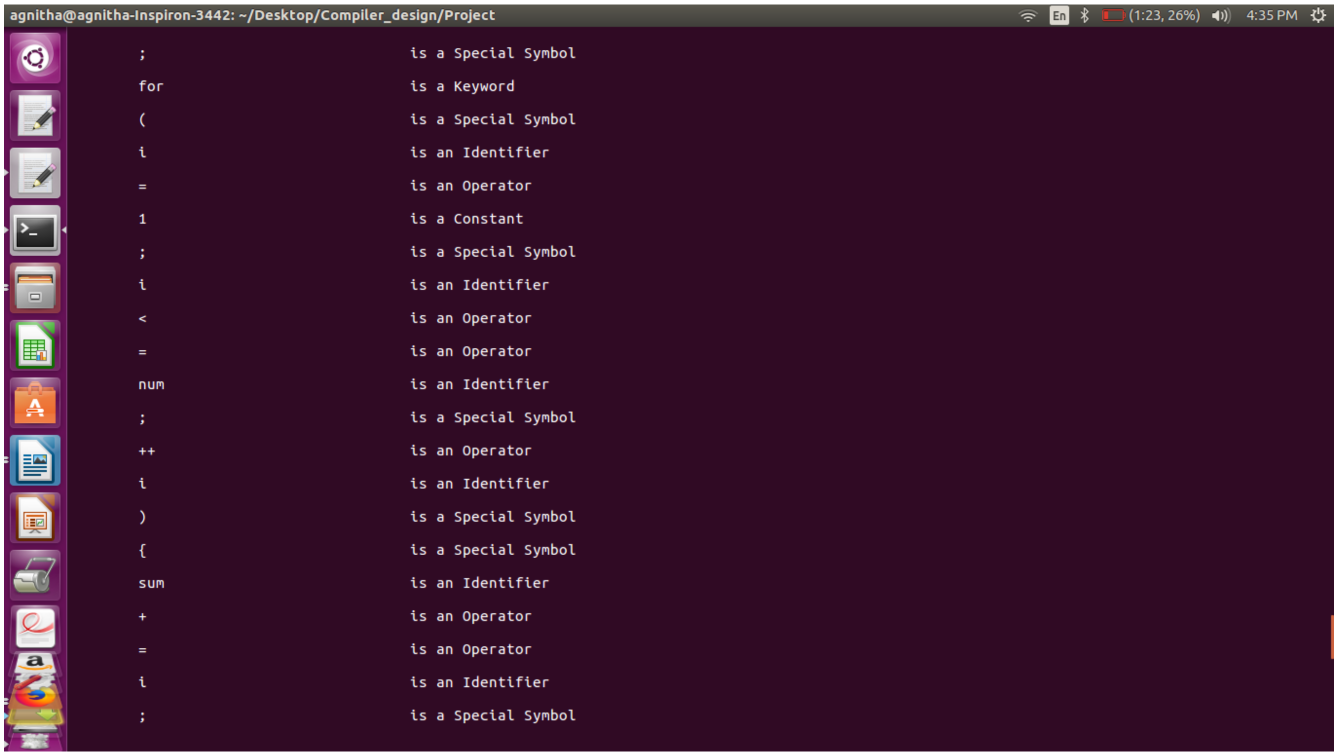
Test Case-6

```
/*program to calculate sum of first
                n positive integers */


#include <stdio.h>


void main()
{
    int num, i, sum = 0;

    printf("Enter a positive integer: ");
    scanf("%d", &num);


    for(i = 1; i <= num; ++i)
    {
        sum += i;
    }

    printf("Sum = %d", sum); // prints calculated sum

    return 0;
}
```



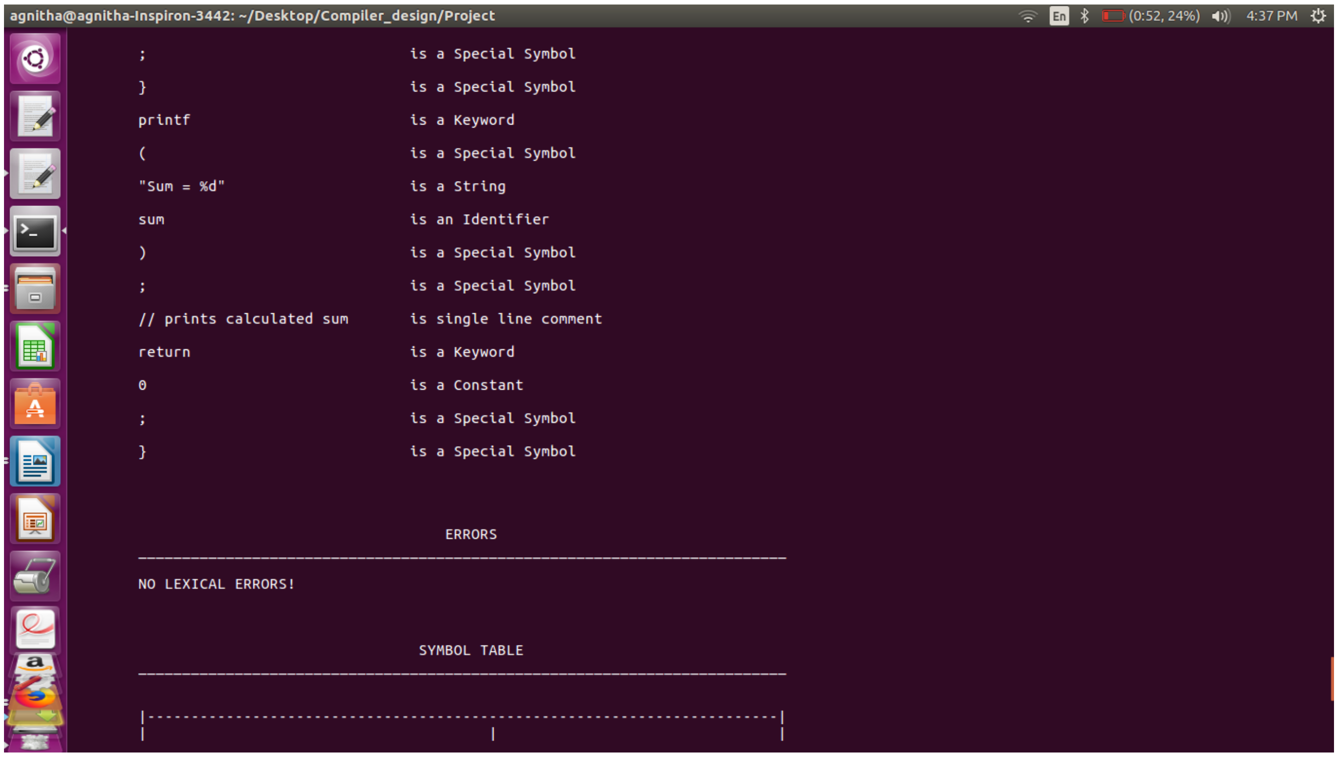
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project

```
#include <stdio.h>              is a Pre-processor directive
void main()                     is the main function
{                               is a Special Symbol
int                             is a Keyword
num                             is an Identifier
i                               is an Identifier
sum                             is an Identifier
=                               is an Operator
0                               is a Constant
;                               is a Special Symbol
printf                          is a Keyword
(                               is a Special Symbol
"Enter a positive integer: "    is a String
)                               is a Special Symbol
;                               is a Special Symbol
scanf                           is a Keyword
(                               is a Special Symbol
"%d"                            is a String
&                               is an Operator
num                             is an Identifier
)                               is a Special Symbol
```

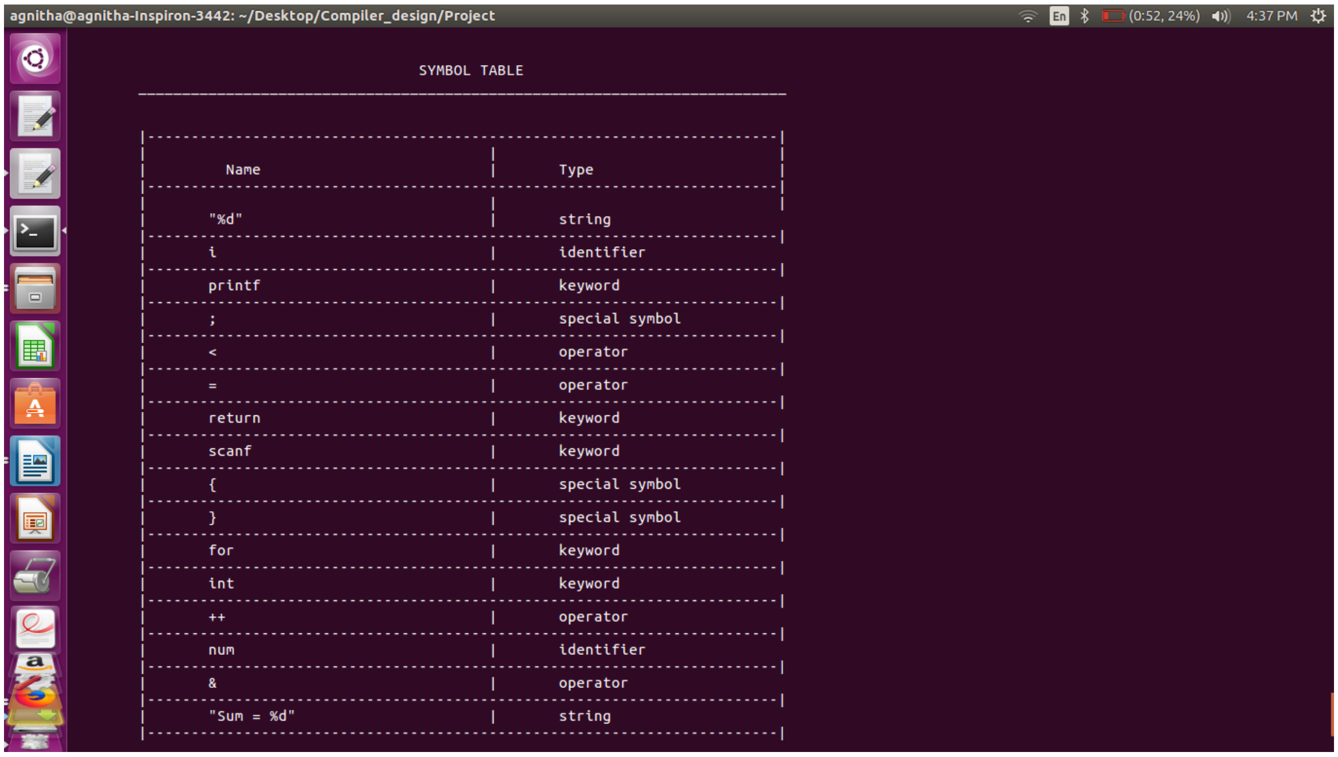


agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project

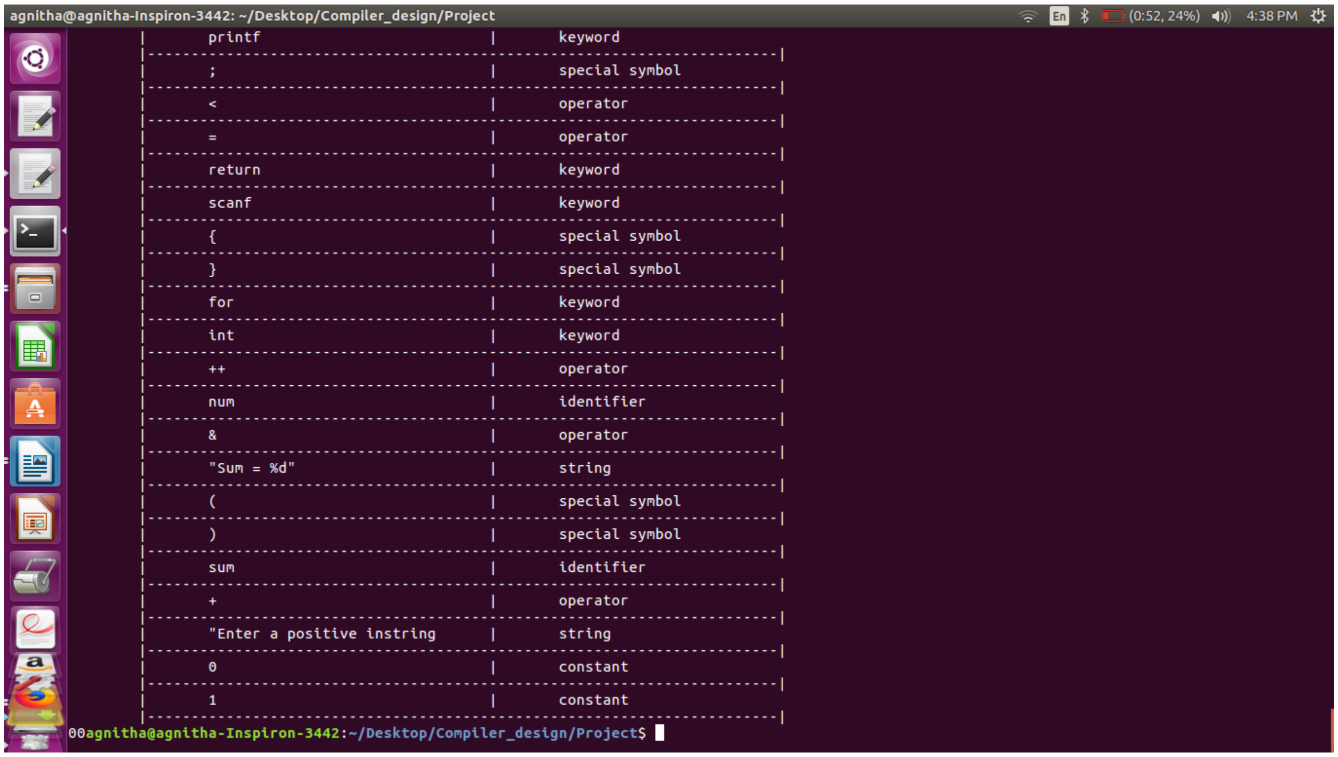
```
;                               is a Special Symbol
for                             is a Keyword
(                               is a Special Symbol
i                               is an Identifier
=                               is an Operator
1                               is a Constant
;                               is a Special Symbol
i                               is an Identifier
<                               is an Operator
=                               is an Operator
num                             is an Identifier
;                               is a Special Symbol
++                              is an Operator
i                               is an Identifier
)                               is a Special Symbol
{                               is a Special Symbol
sum                             is an Identifier
+                               is an Operator
=                               is an Operator
i                               is an Identifier
;                               is a Special Symbol
```



```
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project        (0:52, 24%)   4:37 PM

;                              is a Special Symbol
}                              is a Special Symbol
printf                         is a Keyword
(                              is a Special Symbol
"Sum = %d"                     is a String
sum                            is an Identifier
)                              is a Special Symbol
;                              is a Special Symbol
// prints calculated sum       is single line comment
return                         is a Keyword
0                              is a Constant
;                              is a Special Symbol
}                              is a Special Symbol


                                   ERRORS
----------------------------------------------------------------------------
NO LEXICAL ERRORS!


                                SYMBOL TABLE
----------------------------------------------------------------------------

|--------------------------------------------------------------------------|
|                                        |                                 |
```



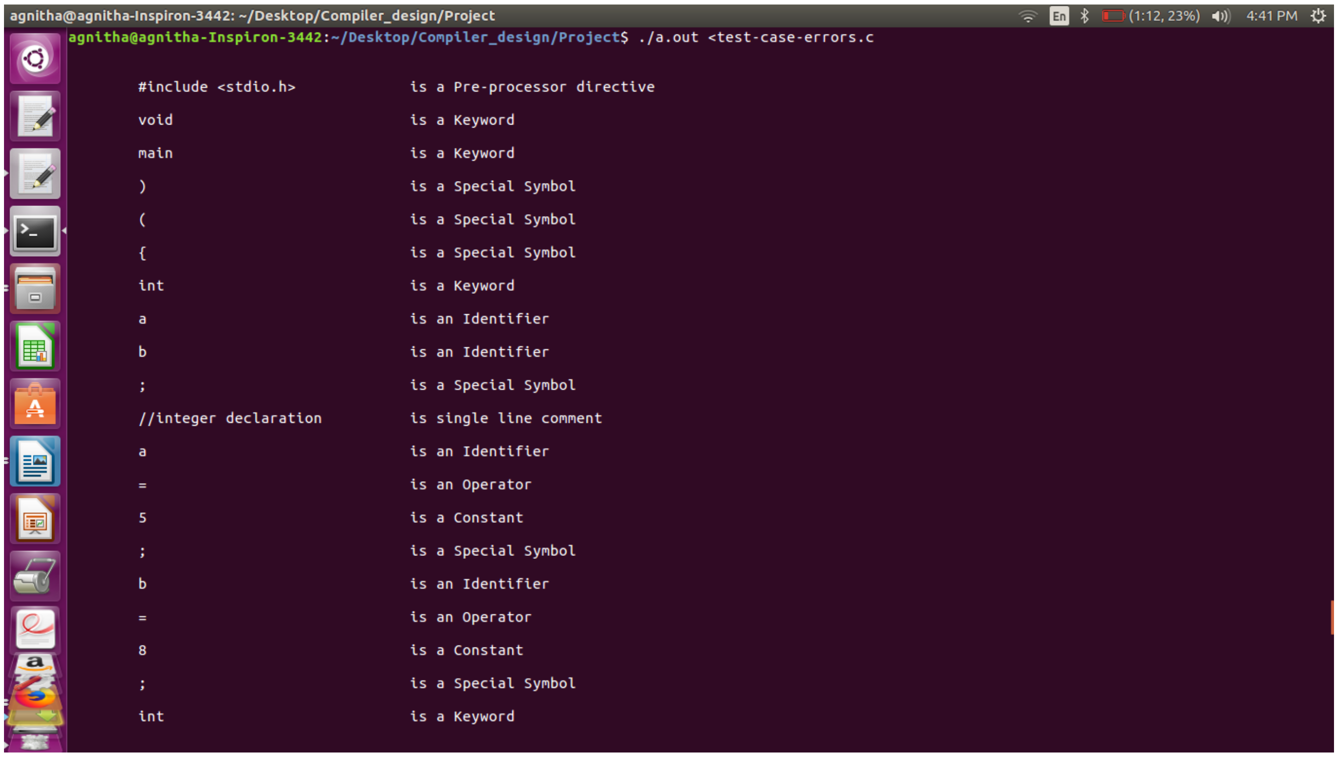
```
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project        (0:52, 24%)   4:37 PM

                                SYMBOL TABLE
----------------------------------------------------------------------------

|--------------------------------------------------------------------------|
|                                        |                                 |
|         Name                           |       Type                      |
|--------------------------------------------------------------------------|
|                                        |                                 |
|       "%d"                             |       string                    |
|--------------------------------------------------------------------------|
|       i                                |       identifier                |
|--------------------------------------------------------------------------|
|       printf                           |       keyword                   |
|--------------------------------------------------------------------------|
|       ;                                |       special symbol            |
|--------------------------------------------------------------------------|
|       <                                |       operator                  |
|--------------------------------------------------------------------------|
|       =                                |       operator                  |
|--------------------------------------------------------------------------|
|       return                           |       keyword                   |
|--------------------------------------------------------------------------|
|       scanf                            |       keyword                   |
|--------------------------------------------------------------------------|
|       {                                |       special symbol            |
|--------------------------------------------------------------------------|
|       }                                |       special symbol            |
|--------------------------------------------------------------------------|
|       for                              |       keyword                   |
|--------------------------------------------------------------------------|
|       int                              |       keyword                   |
|--------------------------------------------------------------------------|
|       ++                               |       operator                  |
|--------------------------------------------------------------------------|
|       num                              |       identifier                |
|--------------------------------------------------------------------------|
|       &                                |       operator                  |
|--------------------------------------------------------------------------|
|       "Sum = %d"                       |       string                    |
|--------------------------------------------------------------------------|
```


agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project
En
(0:52, 24%)
4:38 PM

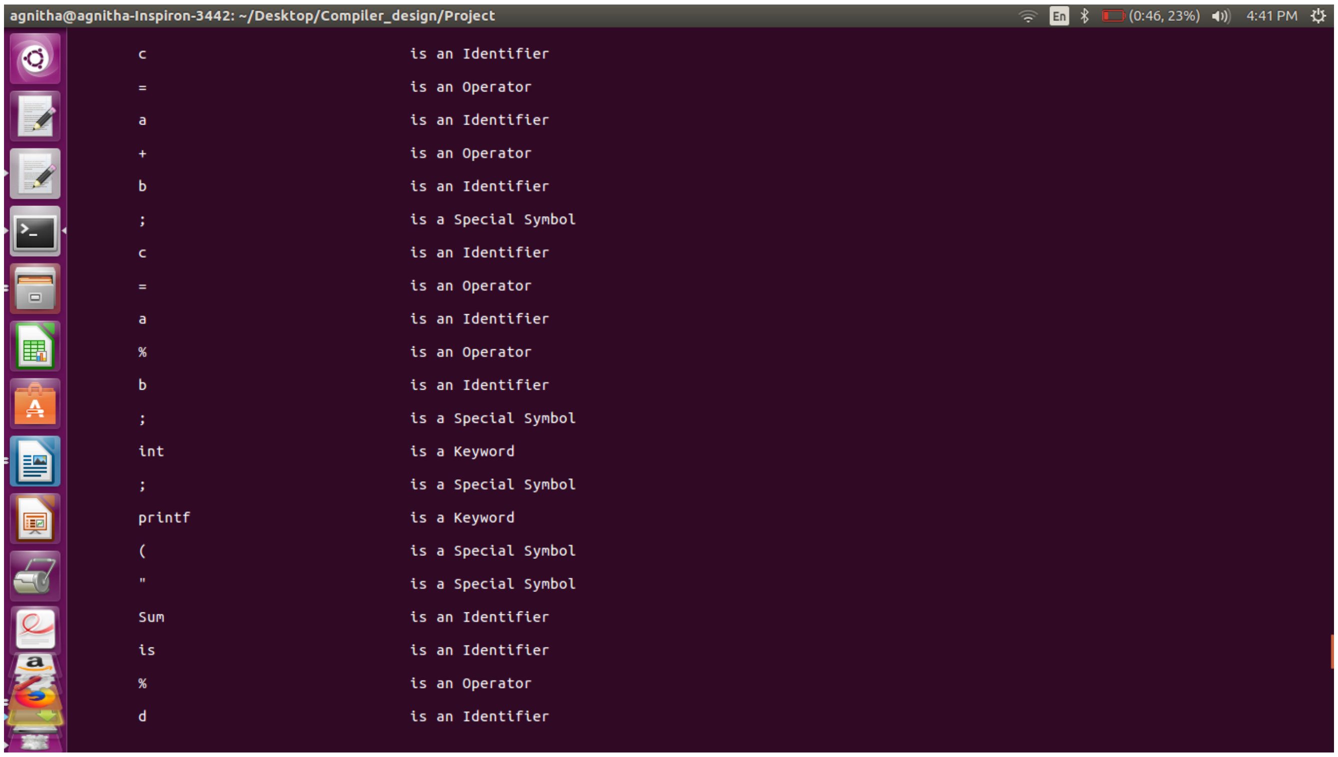
```
        |       printf                          |       keyword
        |-----------------------------------------------------------------------|
        |       ;                               |       special symbol
        |-----------------------------------------------------------------------|
        |       <                               |       operator
        |-----------------------------------------------------------------------|
        |       =                               |       operator
        |-----------------------------------------------------------------------|
        |       return                          |       keyword
        |-----------------------------------------------------------------------|
        |       scanf                           |       keyword
        |-----------------------------------------------------------------------|
        |       {                               |       special symbol
        |-----------------------------------------------------------------------|
        |       }                               |       special symbol
        |-----------------------------------------------------------------------|
        |       for                             |       keyword
        |-----------------------------------------------------------------------|
        |       int                             |       keyword
        |-----------------------------------------------------------------------|
        |       ++                              |       operator
        |-----------------------------------------------------------------------|
        |       num                             |       identifier
        |-----------------------------------------------------------------------|
        |       &                               |       operator
        |-----------------------------------------------------------------------|
        |       "Sum = %d"                      |       string
        |-----------------------------------------------------------------------|
        |       (                               |       special symbol
        |-----------------------------------------------------------------------|
        |       )                               |       special symbol
        |-----------------------------------------------------------------------|
        |       sum                             |       identifier
        |-----------------------------------------------------------------------|
        |       +                               |       operator
        |-----------------------------------------------------------------------|
        |       "Enter a positive instring      |       string
        |-----------------------------------------------------------------------|
        |       0                               |       constant
        |-----------------------------------------------------------------------|
        |       1                               |       constant
        |-----------------------------------------------------------------------|
00agnitha@agnitha-Inspiron-3442:~/Desktop/Compiler_design/Project$
```

Testcase-7

```
#include <stdio.h>

void main)(
{
		int a,b; //integer declaration

		a=5;
		b=8;

		int c=a+b;
		c=a%b;
		int %han;

		printf("Sum is %d , c ;

		/*Program to calculate
		  sum of 2 integers
```

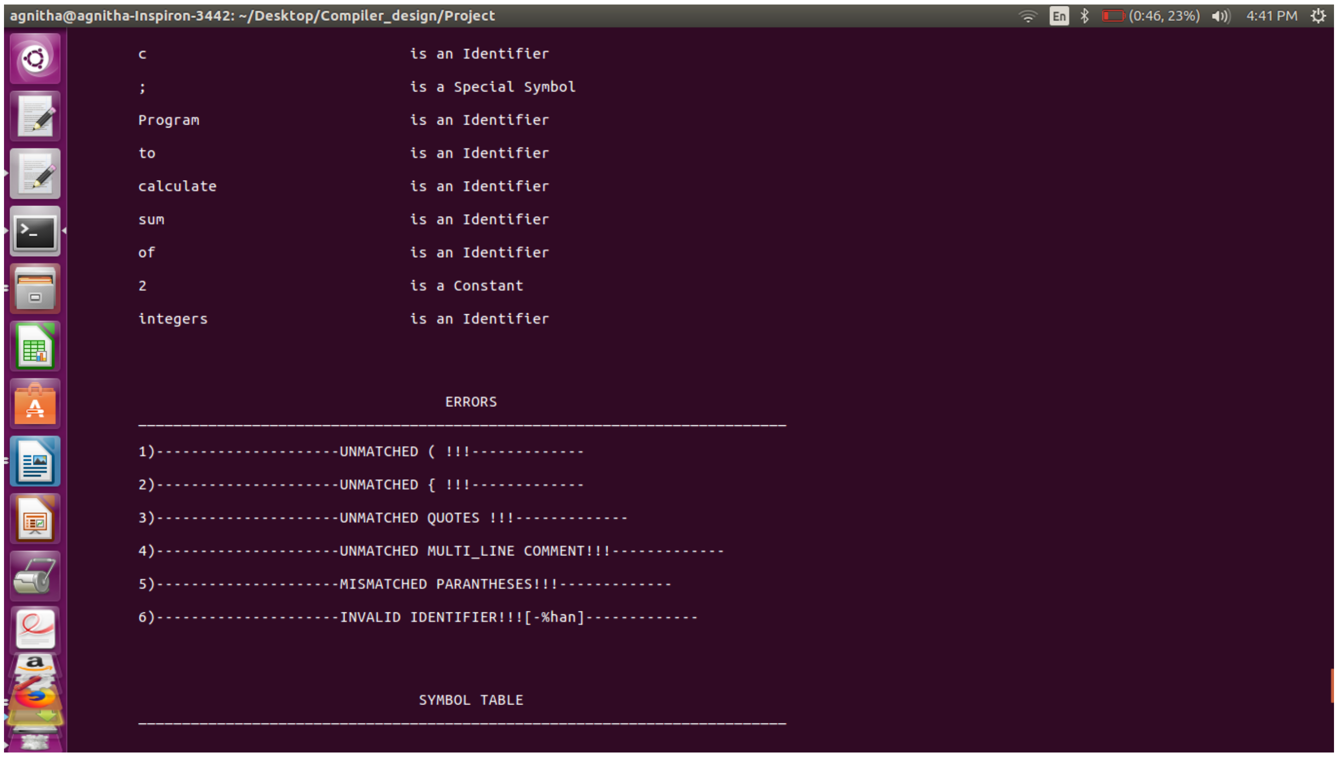


```
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project
agnitha@agnitha-Inspiron-3442:~/Desktop/Compiler_design/Project$ ./a.out <test-case-errors.c

        #include <stdio.h>              is a Pre-processor directive
        void                            is a Keyword
        main                            is a Keyword
        )                               is a Special Symbol
        (                               is a Special Symbol
        {                               is a Special Symbol
        int                             is a Keyword
        a                               is an Identifier
        b                               is an Identifier
        ;                               is a Special Symbol
        //integer declaration           is single line comment
        a                               is an Identifier
        =                               is an Operator
        5                               is a Constant
        ;                               is a Special Symbol
        b                               is an Identifier
        =                               is an Operator
        8                               is a Constant
        ;                               is a Special Symbol
        int                             is a Keyword
```



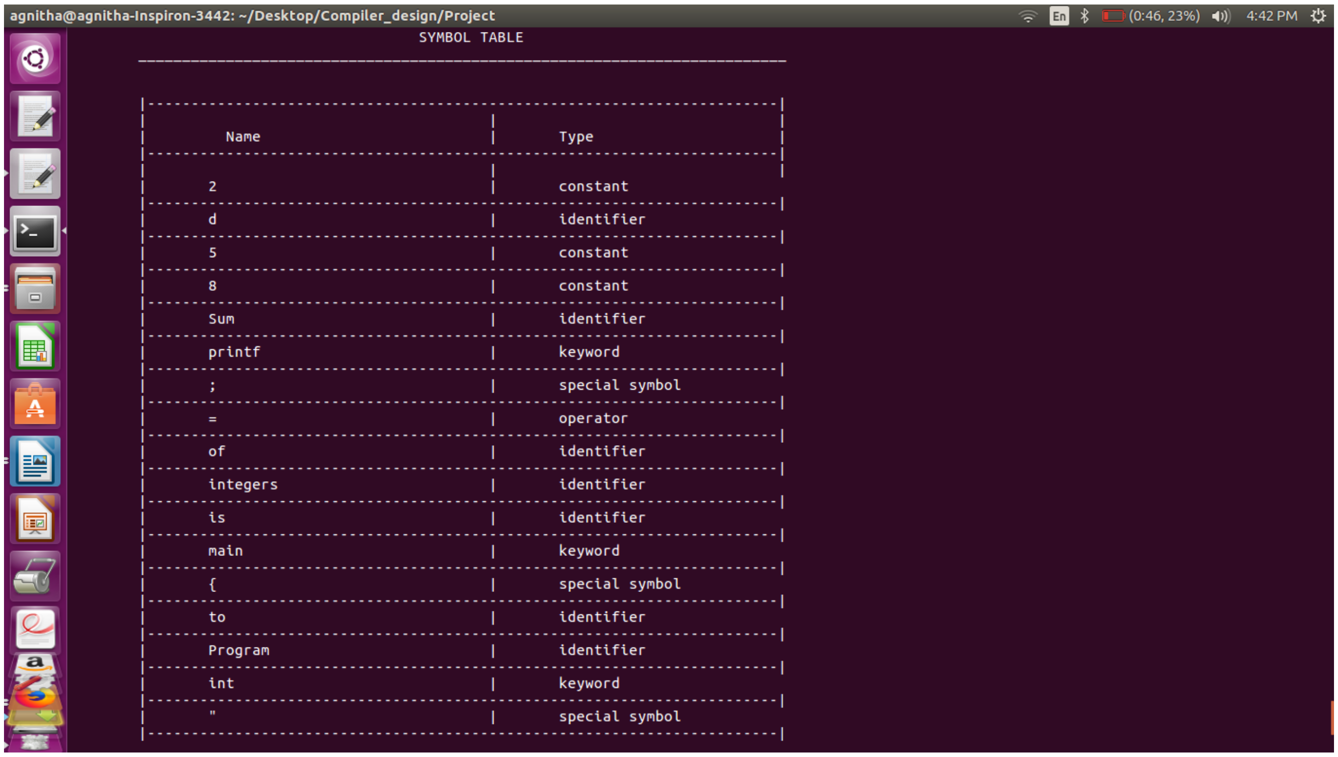
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project

```
c                              is an Identifier
=                              is an Operator
a                              is an Identifier
+                              is an Operator
b                              is an Identifier
;                              is a Special Symbol
c                              is an Identifier
=                              is an Operator
a                              is an Identifier
%                              is an Operator
b                              is an Identifier
;                              is a Special Symbol
int                            is a Keyword
;                              is a Special Symbol
printf                         is a Keyword
(                              is a Special Symbol
"                              is a Special Symbol
Sum                            is an Identifier
is                             is an Identifier
%                              is an Operator
d                              is an Identifier
```

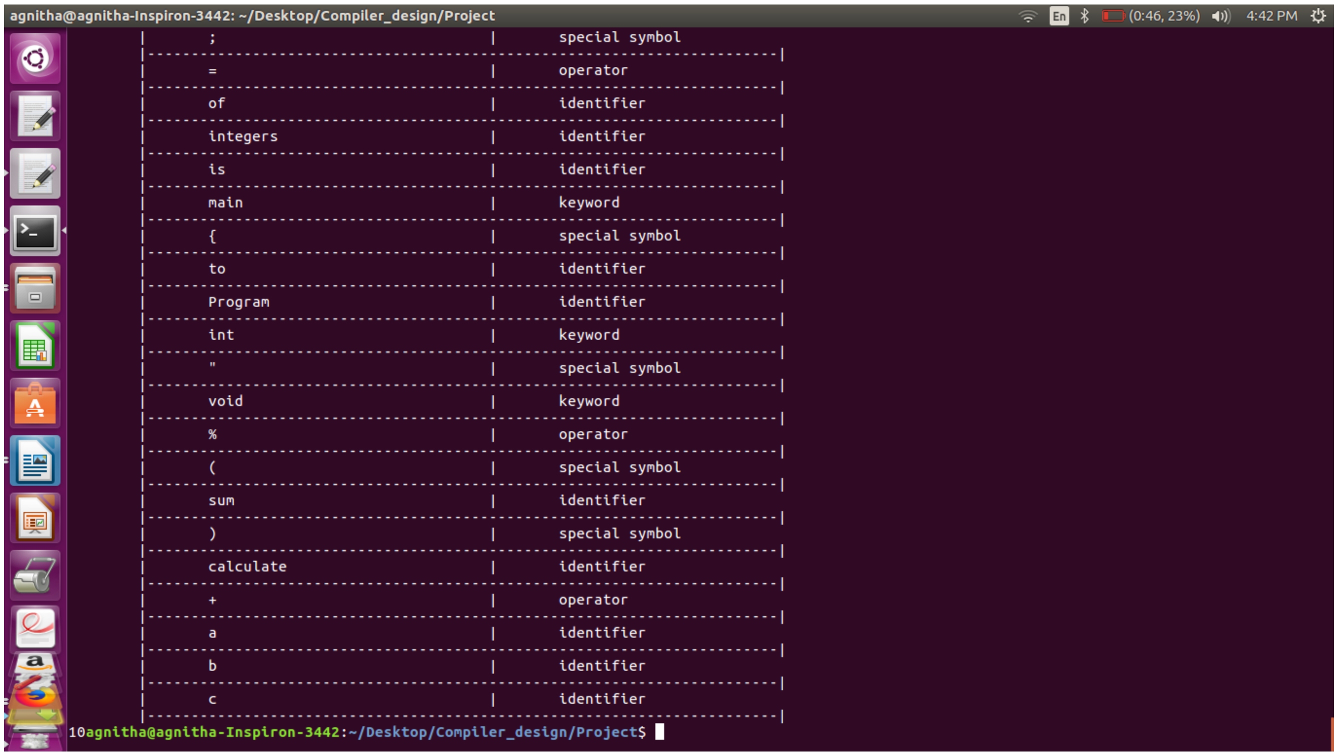


agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project

```
c                              is an Identifier
;                              is a Special Symbol
Program                        is an Identifier
to                             is an Identifier
calculate                      is an Identifier
sum                            is an Identifier
of                             is an Identifier
2                              is a Constant
integers                       is an Identifier


                                  ERRORS
__________________________________________________________________________
1)--------------------UNMATCHED ( !!!------------
2)--------------------UNMATCHED { !!!------------
3)--------------------UNMATCHED QUOTES !!!------------
4)--------------------UNMATCHED MULTI_LINE COMMENT!!!------------
5)--------------------MISMATCHED PARANTHESES!!!------------
6)--------------------INVALID IDENTIFIER!!![-%han]------------


                               SYMBOL TABLE
__________________________________________________________________________
```



```
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project
                                        SYMBOL TABLE
          ------------------------------------------------------------------------

          |-----------------------------------------------------------------------|
          |                                       |                               |
          |        Name                           |       Type                    |
          |-----------------------------------------------------------------------|
          |                                       |                               |
          |      2                                |       constant                |
          |-----------------------------------------------------------------------|
          |      d                                |       identifier              |
          |-----------------------------------------------------------------------|
          |      5                                |       constant                |
          |-----------------------------------------------------------------------|
          |      8                                |       constant                |
          |-----------------------------------------------------------------------|
          |      Sum                              |       identifier              |
          |-----------------------------------------------------------------------|
          |      printf                           |       keyword                 |
          |-----------------------------------------------------------------------|
          |      ;                                |       special symbol          |
          |-----------------------------------------------------------------------|
          |      =                                |       operator                |
          |-----------------------------------------------------------------------|
          |      of                               |       identifier              |
          |-----------------------------------------------------------------------|
          |      integers                         |       identifier              |
          |-----------------------------------------------------------------------|
          |      is                               |       identifier              |
          |-----------------------------------------------------------------------|
          |      main                             |       keyword                 |
          |-----------------------------------------------------------------------|
          |      {                                |       special symbol          |
          |-----------------------------------------------------------------------|
          |      to                               |       identifier              |
          |-----------------------------------------------------------------------|
          |      Program                          |       identifier              |
          |-----------------------------------------------------------------------|
          |      int                              |       keyword                 |
          |-----------------------------------------------------------------------|
          |      "                                |       special symbol          |
          |-----------------------------------------------------------------------|
```



```
agnitha@agnitha-Inspiron-3442: ~/Desktop/Compiler_design/Project
          |      ;                                |       special symbol
          |-----------------------------------------------------------------------|
          |      =                                |       operator                |
          |-----------------------------------------------------------------------|
          |      of                               |       identifier              |
          |-----------------------------------------------------------------------|
          |      integers                         |       identifier              |
          |-----------------------------------------------------------------------|
          |      is                               |       identifier              |
          |-----------------------------------------------------------------------|
          |      main                             |       keyword                 |
          |-----------------------------------------------------------------------|
          |      {                                |       special symbol          |
          |-----------------------------------------------------------------------|
          |      to                               |       identifier              |
          |-----------------------------------------------------------------------|
          |      Program                          |       identifier              |
          |-----------------------------------------------------------------------|
          |      int                              |       keyword                 |
          |-----------------------------------------------------------------------|
          |      "                                |       special symbol          |
          |-----------------------------------------------------------------------|
          |      void                             |       keyword                 |
          |-----------------------------------------------------------------------|
          |      %                                |       operator                |
          |-----------------------------------------------------------------------|
          |      (                                |       special symbol          |
          |-----------------------------------------------------------------------|
          |      sum                              |       identifier              |
          |-----------------------------------------------------------------------|
          |      )                                |       special symbol          |
          |-----------------------------------------------------------------------|
          |      calculate                        |       identifier              |
          |-----------------------------------------------------------------------|
          |      +                                |       operator                |
          |-----------------------------------------------------------------------|
          |      a                                |       identifier              |
          |-----------------------------------------------------------------------|
          |      b                                |       identifier              |
          |-----------------------------------------------------------------------|
          |      c                                |       identifier              |
          |-----------------------------------------------------------------------|
10agnitha@agnitha-Inspiron-3442:~/Desktop/Compiler_design/Project$
```

Description and Status of Test Cases:

TEST CASE	DESCRIPTION	STATUS
1	Simple Hello World	PASS
2	Basic-if-else for	PASS
3	Single-line-comment	PASS
4	Multi-line-comment	PASS
5	Nested-for-loop for finding prime numbers within a range	PASS
6	Simple for loop that does arithmetic operations for calculating the sum-of-first-n-positive numbers	PASS
7	Program with errors like: • Unmatched paranthesis • Unmatched curly braces • Unmatched quotes • Unmatched multi line comment • Mismatched paranthesis • Invalid identifier	PASS

Conclusion and Future Work

The project aimed to build a lexical analyzer which converted source program in high level language to a sequence of tokens.
It took care of lexical errors and used hashing concepts to optimize the storage of tokens.

Future work:
The program uses a very primitive and crude way to accept nested comments. We plan on optimizing by doing more research.

References

- Compilers – Principles , Techniques and tools by Alfred V. Aho , Monica S. Larn , Ravi Sethi , Jeffrey D. Ullman
- Lex and Yacc by John R. Levine , Tony Mason , Doug Brown
- https://en.wikipedia.org/wiki/
- https://www.tutorialspoint.