|  |  |  |  |
| --- | --- | --- | --- |
| **Ex. No. 02** | **TOKEN SEPERATION USING LEX** | | |
| Date of Exercise | \_\_\_\_\_\_\_\_\_\_\_\_\_ | Date of Output Verification | \_\_\_\_\_\_\_\_\_\_\_\_ |

**Question**

Design a lexical analyzer for a given language using LEX.

**Program**

digit [0-9]

letter [a-zA-Z]

backslash [/]

forwardslash [\]

closedcurvedbrace [)]

openedcurvedbrace [(]

star [\*]

comma [,]

space [ ]

doublequotes ["]

%%

#include<{letter}\*.{letter}> printf("\n%s - - - - - - - Header\n", yytext);

{backslash}{backslash}({digit}|{letter}|{space})\* printf("\n%s - - - - - - - Single Line Comment\n", yytext);

{backslash}{star}({letter}|{digit}|{space})\*{star}{backslash} printf("\n%s - - - - - - - Multi Line Comment\n", yytext);

{space}\*{letter}({letter}|{digit})\*{space}\*({comma}|;) printf("\n%s - - - - - - - Declared Variable\n", yytext);

if |

then |

else |

int |

char |

float |

printf printf("\n%s -------- Keywords\n", yytext);

{doublequotes}({digit}|{letter}|{space})\*{doublequotes} printf("\n %s ------- Literal\n", yytext);

{letter}\*{openedcurvedbrace}{closedcurvedbrace}; printf("\n%s - - - - - - - Function Name\n", yytext);

%%

int yywrap(){return 1;}

int main()

{

yyin=fopen("sample.c", "r");

printf("\nExp2 - Token Seperation Using LEX - UR12CS135\n\n");

yylex();

return 0;

}

**Input**

Sample.c

#include<stdio.h>

int main()

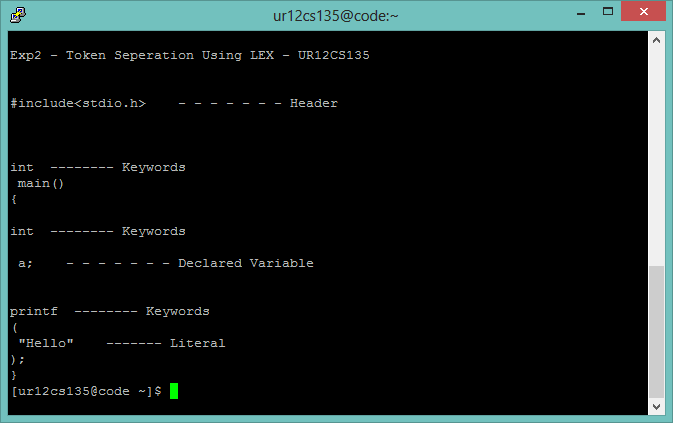
{

int a;

printf("Hello");

}

**Output**



**Result**

Lexical Analyzer for C language is successfully Designed.

[Signature of the Staff In-charge]

Name of the Staff In – charge: Mr. Jeban Chandir Moses

Date: \_\_\_\_\_\_\_\_\_\_\_\_\_