**Roll No:**

**Lab Performance Test [No]**

**Lab Task Q[No]**

**Question: Q1.** Consider given code,

#include<math.h>

int main(){

float a, b, c;

a=1;  b=2;

c=a+b;

return 0;

}

Show output file with extension ".i" generated by C compiler along with Makefile (point penalty for adding extra commands other than necessary).

**Solution (Bold your own written code):**

|  |
| --- |
|  |

**Output (Screen/SnapShot):**

**Question: Q2**.

Consider given statements,

1. WHILE a>b: a++;

2. WHILE a!=b: a--;

3. WHILE a<b: a+=b;

a) Show a flex file which can tokenize given statements.

b) Show a bison file which can parse given statements.

**Solution (Bold your own written code):a**

|  |
| --- |
| %option noyywrap  %{    #include "prog.tab.h"  %}  delim [ \t\n]  ws {delim}+  letter [A-Za-z]  us [\_]  id ({us}|{letter})({us}|{letter}|{digit})\*  %%  {ws} {}  "++" {return (INCR);}  "--" {return (DECR);}  ":" {return (THEN);}  "WHILE" {return (WHILE);}  {id} {return (ID);}  "!=" {return (NE);}  "==" {return (EQ);}  "+=" {return (ADD\_ASSIGN);}  ">" {return (GT);}  "<" {return (LT);}  ";" {return (SEMI);}  %% |

**Solution (Bold your own written code):b**

|  |
| --- |
| %{  /\*  \*/  #include<stdio.h>  void yyerror(char \*s);  int yylex();  %}  %token INCR DECR THEN WHILE ID NE EQ ADD\_ASSIGN GT LT SEMI  %start prog  %%  prog: statements  ;  statements : statements statement  |  ;  statement : while\_statement  ;  while\_statement: WHILE exp THEN exp  ;  exp: ID INCR  | ID DECR  | ID ADD\_ASSIGN ID  ;  %%  int main(){  yyparse();  printf("Parshing Finished\n");  return 0;  }  void yyerror(char \*s){  fprintf(stderr,"Error:%s\n",s);  } |

**Output (Screen/SnapShot):**