Week 11 Assignment

Problem

Implement LALR Parser for the following grammar

E → E+T |T

E'→ T*F | F

F → (E) | d

lalr.y

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

%union{
  char* str;
}

%token <str> NUM

%type <str> E F T

%%

exp: E '\n' { printf("= %s\nsuccessfully parsed\n", $1); }
   |exp E '\n' { printf("= %s\nsuccessfully parsed\n", $2); };

E: E '+' T { char temp[256]; sprintf(temp, "%s+%s", $1, $3); free($1);
free($3); $$ = strdup(temp); }
   | T { $$ = $1; };
```

```
T: T '*' F { char temp[256]; sprintf(temp, "%s*%s", $1, $3); free($1);
free($3); $$ = strdup(temp); }
      | F { $$ = $1; };


F: '(' E ')' { char temp[256]; sprintf(temp, "(%s)", $2); free($2); $$ =
strdup(temp); }
   | NUM { $$ = $1; };



%%


void yyerror(const char *s) {
    fprintf(stderr, "Can't parse the string\n");
}


int main()
{
yyparse();
return 0;
}
```

lalr.l

```
%{
#include "lalr.tab.h"
%}

%%

d { yylval.str = strdup(yytext); return NUM; }
[-+/*()\n] { return *yytext; }
[ \t] ;
. { fprintf(stderr, "Invalid character: %s\n", yytext); }

%%

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

Input & Output:

```
PS D:\SRM AP\SEM 5\Compiler design Lab\Week11> .\parser.exe
d*d+d
= d*d+d
successfully parsed
d*(d+d)
= d*(d+d)
successfully parsed
```