1. Design and Implementation of a Simple Compiler using Flex & Bison

Description: This project demonstrates the implementation of a basic compiler front-end using Flex (Fast Lexical Analyzer) and Bison (GNU Parser Generator). It performs lexical analysis to identify tokens from the input and uses syntax analysis to validate the structure of expressions based on predefined grammar rules. The goal is to understand the compilation process and the roles of lexical and syntax analyzers in compiler design.

Bison code:

```
%{
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
int yylex(void);
void yyerror(const char *s);
%}
%union
{
int num;
char *str;
};

%token PRINT ADD
%token <str> STRING
%token <num> NUMBER

%%
start:
|start input
;
input:
STRING PRINT ';' {printf("output:%s\n",$1);}
| NUMBER ADD NUMBER ';' {printf("sum:%d\n",$1+$3);}
;
%%

void yyerror(const char *s)
{printf("syntax error: %s\n",s);}

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

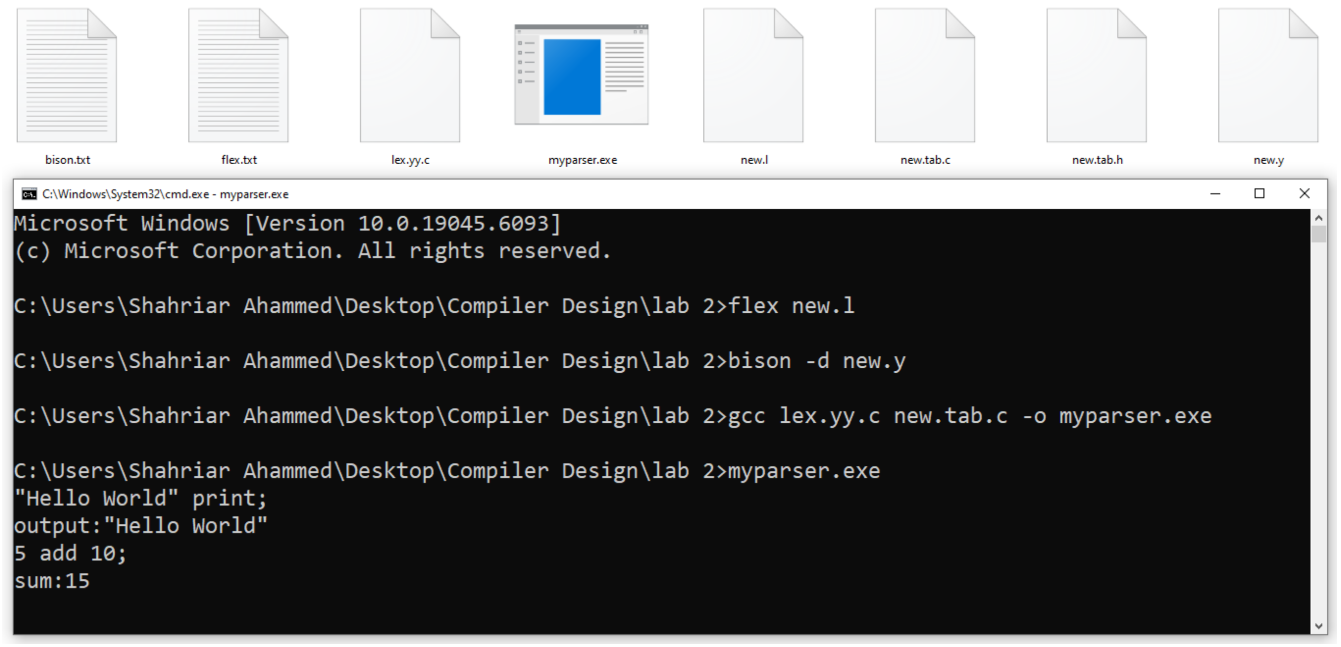
Flex code:

```
%{
#include "new.tab.h"
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
%}

%%
print {return PRINT;}
add {return ADD;}
\"[^\"]+\" {yylval.str=strdup(yytext); return STRING;}
[0-9]+ {yylval.num=atoi(yytext); return NUMBER;}
[ \n\t] ;
. {return yytext[0];}
%%

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

Output :



```
Microsoft Windows [Version 10.0.19045.6093]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Shahriar Ahammed\Desktop\Compiler Design\lab 2>flex new.l

C:\Users\Shahriar Ahammed\Desktop\Compiler Design\lab 2>bison -d new.y

C:\Users\Shahriar Ahammed\Desktop\Compiler Design\lab 2>gcc lex.yy.c new.tab.c -o myparser.exe

C:\Users\Shahriar Ahammed\Desktop\Compiler Design\lab 2>myparser.exe
"Hello World" print;
output:"Hello World"
5 add 10;
sum:15
```