

Name: Spandan Mukherjee
Subject: Compiler Design

Experiment 4

1) Program to recognize the grammar A^nB^n

ALGORITHM :

1. Start the program.
2. Write the code for parser. l in the declaration port.
3. Write the code for the 'y' parser.
4. Also write the code for different arithmetical operations.
6. Execute and verify it.
7. Stop the program.

Code:

Yacc Code:

```
% {  
    #include<stdio.h>  
    #include<stdlib.h>  
% }  
  
%token A B NL  
%%  
stmt: S NL { printf("valid string\n");  
            exit(0); }  
;  
S: A S B |  
;
```

%%

```
int yyerror(char *msg)
{
    printf("invalid string\n");
    exit(0);
}
```

```
main()
{
    printf("enter the string\n");
    yyparse();
}
```

Lex code:

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

```
%%
[aA] {return A;}
[bB] {return B;}
\n {return NL;}
. {return yytext[0];}
%%
```

```
int yywrap() { }
```

OUTPUT:

```
spandan@spandan-VirtualBox: ~  
spandan@spandan-VirtualBox:~$ cc lex.yy.c y.tab.c  
y.tab.c: In function 'yyparse':  
y.tab.c:1017:16: warning: implicit declaration of function 'yylex' [-Wimplicit-f  
unction-declaration]  
1017 |         yychar = yylex ();  
      |                  ^~~~~~  
y.tab.c:1159:7: warning: implicit declaration of function 'yyerror'; did you mea  
n 'yyerrok'? [-Wimplicit-function-declaration]  
1159 |         yyerror (YY_("syntax error"));  
      |         ^~~~~~  
      |         yyerrok  
yacc2.y: At top level:  
yacc2.y:21:1: warning: return type defaults to 'int' [-Wimplicit-int]  
21 | main()  
   | ^~~~~~  
spandan@spandan-VirtualBox:~$ ./a.out  
enter the string  
AAABBB  
valid string  
spandan@spandan-VirtualBox:~$ gedit yacc2.l  
spandan@spandan-VirtualBox:~$ gedit yacc2.y  
^C  
spandan@spandan-VirtualBox:~$ gedit lex.l
```

2) Program to recognize to AB^n

ALGORITHM :

1. Start the program.
2. Write the code for parser. l in the declaration port.
3. Write the code for the 'y' parser.
4. Also write the code for different arithmetical operations.
6. Execute and verify it.
7. Stop the program.

CODE:

LEX CODE:

```
% {  
  
#include "y.tab.h"  
  
% }  
  
%%  
  
[aA] { return A;}  
  
[bB] {return B;}  
  
\n {return N;}  
. {return yytext[0];}  
  
%%  
  
int yywrap()
```

```
{  
  
return 1;  
  
}
```

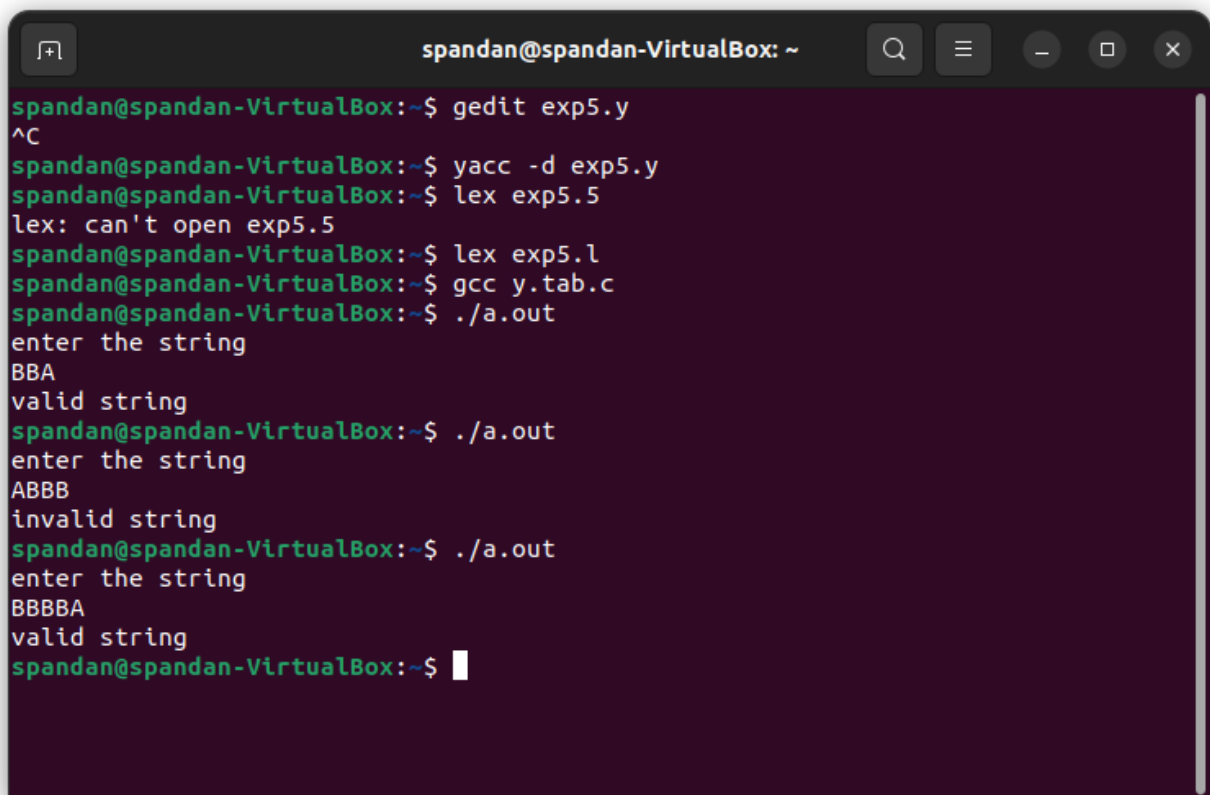
YACC CODE:

```
% {  
/* Definition section */  
#include <stdio.h>  
#include <stdlib.h>  
#include "lex.yy.c"  
int yywrap();  
int yylex();  
int yyerror(char *msg);  
% }  
  
%token A B N  
  
/* Rule Section */  
%%  
S: B S A N { printf("valid string\n"); exit(0); }  
;  
S:B S A|  
  
;  
%%  
int yyerror(char *msg)  
{
```

```
printf("invalid string\n");  
exit(0);  
}
```

```
int main(){  
printf("enter the string\n");  
yyparse();  
}
```

OUTPUT:



```
spandan@spandan-VirtualBox: ~  
spandan@spandan-VirtualBox:~$ gedit exp5.y  
^C  
spandan@spandan-VirtualBox:~$ yacc -d exp5.y  
spandan@spandan-VirtualBox:~$ lex exp5.5  
lex: can't open exp5.5  
spandan@spandan-VirtualBox:~$ lex exp5.l  
spandan@spandan-VirtualBox:~$ gcc y.tab.c  
spandan@spandan-VirtualBox:~$ ./a.out  
enter the string  
BBA  
valid string  
spandan@spandan-VirtualBox:~$ ./a.out  
enter the string  
ABBB  
invalid string  
spandan@spandan-VirtualBox:~$ ./a.out  
enter the string  
BBBBBA  
valid string  
spandan@spandan-VirtualBox:~$
```

3) Program to recognize the Grammar A^nB

ALGORITHM :

1. Start the program.
2. Write the code for parser. l in the declaration port.
3. Write the code for the 'y' parser.
4. Also write the code for different arithmetical operations.
6. Execute and verify it.
7. Stop the program.

CODE:

Yacc code

```
% {  
/* Definition section */  
#include <stdio.h>  
#include <stdlib.h>  
#include "lex.yy.c"  
int yywrap();  
int yylex();  
int yyerror(char *msg);  
% }
```

%token A B N

%%

```
S: A S B N { printf("valid string\n"); exit(0); }  
;
```

S:A S B|

```
;  
%%  
int yyerror(char *msg)  
{  
    printf("invalid string\n");  
    exit(0);  
}  
  
int main(){  
    printf("enter the string\n");  
    yyparse();  
}
```

LEX CODE:

```
% {  
  
#include "y.tab.h"  
  
% }  
  
%%  
  
[aA] { return A;}  
  
[bB] {return B;}  
  
\n {return N;}  
. {return yytext[0];}  
  
%%
```



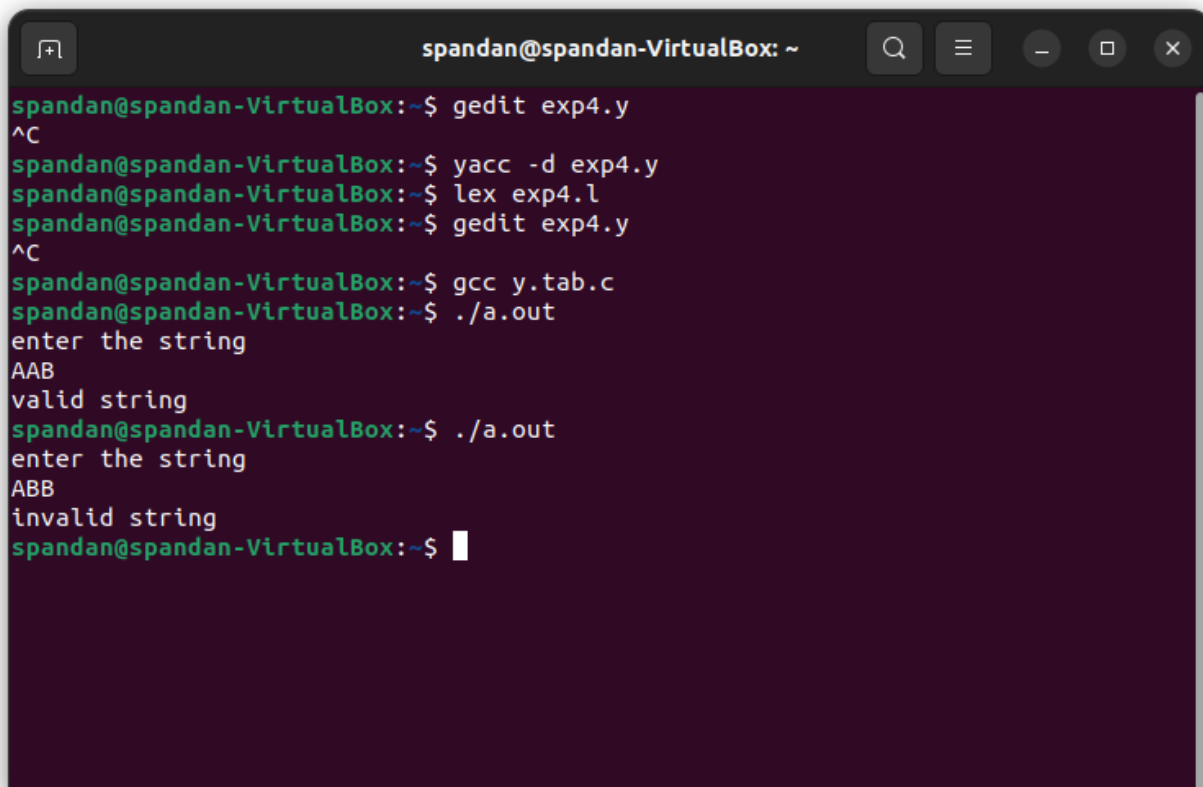
```
int yywrap()

{

return 1;

}
```

OUTPUT:

A terminal window titled 'spandan@spandan-VirtualBox: ~' with standard window controls. The terminal shows the following sequence of commands and output:

```
spandan@spandan-VirtualBox:~$ gedit exp4.y
^C
spandan@spandan-VirtualBox:~$ yacc -d exp4.y
spandan@spandan-VirtualBox:~$ lex exp4.l
spandan@spandan-VirtualBox:~$ gedit exp4.y
^C
spandan@spandan-VirtualBox:~$ gcc y.tab.c
spandan@spandan-VirtualBox:~$ ./a.out
enter the string
AAB
valid string
spandan@spandan-VirtualBox:~$ ./a.out
enter the string
ABB
invalid string
spandan@spandan-VirtualBox:~$
```