BCSE307P – Compiler Design Lab

Winter Semester 2023-24

Assessment 9

Implementation of Shift Reduce Parsing

Name: Sujay Ghosh

Reg. No: 21BLC1607

Slot: L7 + L8

Faculty: Dr. Rathna

Task 1:

Given grammar

E -> E +E

E -> E * E

E -> (E)

E -> id

Input: id+(id*id)

Code:

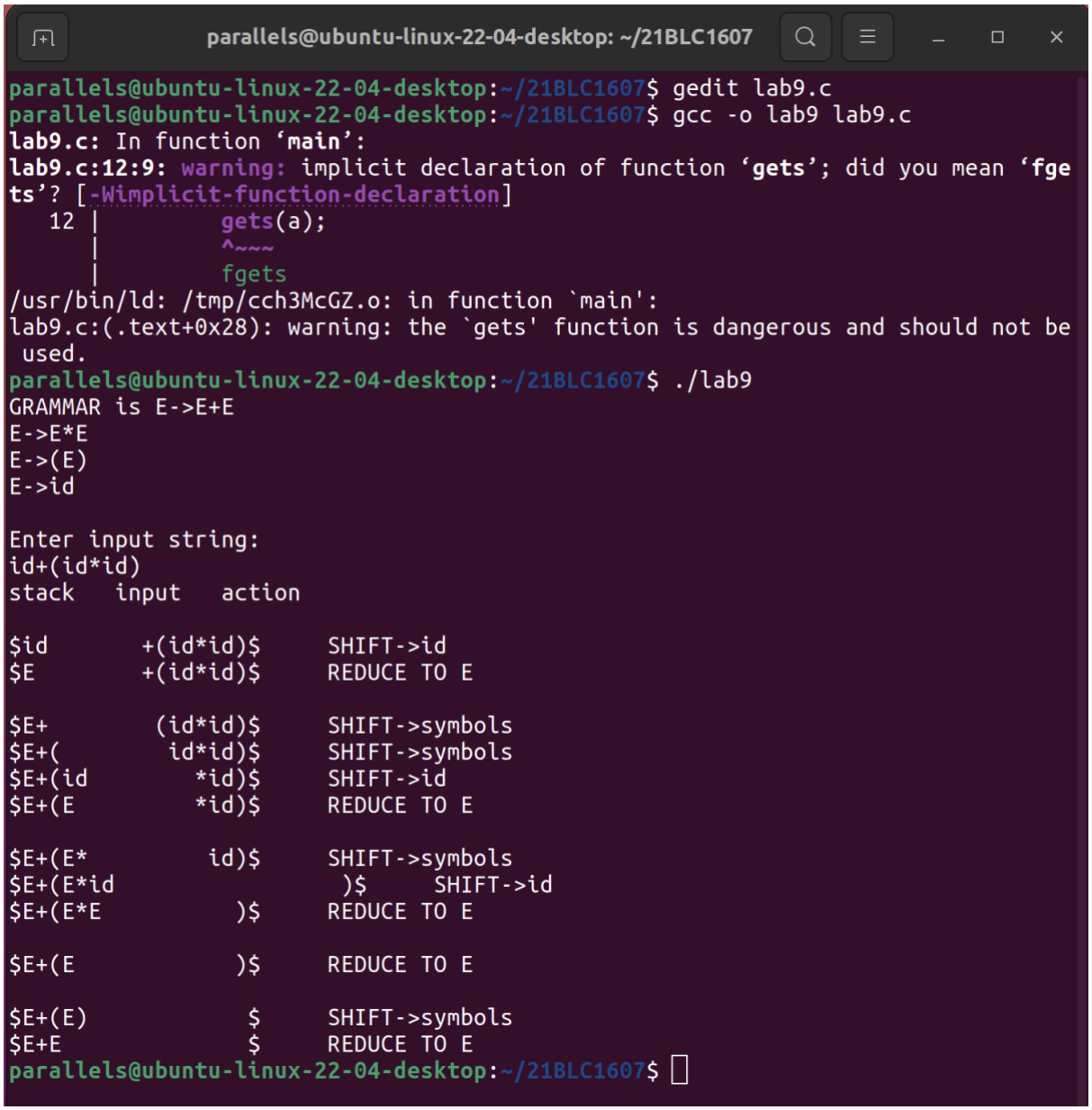
```
#include <stdio.h>
#include <string.h>

int k=0, z=0, i=0, j=0, c=0;
char a[16], ac[20], stk[15], act[10];

void check();

int main() {
      puts("GRAMMAR is E->E+E\nE->E*E\nE->(E)\nE->id\n");
      puts("Enter input string: ");
      gets(a);
      c = strlen(a);
      strcpy(act, "SHIFT->");
      puts("stack\tinput\taction");
      for (k=0, i=0; j<c; k++, i++, j++) {
            if (a[j] == 'i' && a[j+1] == 'd') {
```

```
                stk[i] = a[j];
                stk[i+1] = a[j+1];
                stk[i+2] = '\0';
                a[j] = ' ';
                a[j+1] = ' ';
                printf("\n$%s\t%s$\t%sid", stk, a, act);
                check();
          }
          else {
                stk[i] = a[j];
                stk[i+1] = '\0';
                a[j] = ' ';
                printf("\n$%s\t%s$\t%ssymbols", stk, a, act);
                check();
          }
     }
}
void check() {
     strcpy(ac, "REDUCE TO E\n");
     for (z=0; z<c; z++) {
          if (stk[z] == 'i' && stk[z+1] == 'd') {
                stk[z] = 'E';
                stk[z+1] = '\0';
                printf("\n$%s\t%s$\t%s", stk, a, ac);
                j++;
          }
```

```
	}
	for (z=0; z<c; z++) {
		if (stk[z] == 'E' && stk[z+1] == '+' && stk[z+2] == 'E')
{
			stk[z] = 'E';
			stk[z+1] = '\0';
			stk[z+2] = '\0';
			printf("\n$%s\t%s$\t%s", stk, a, ac);
			i = i-2;
		}
	}
	for (z=0; z<c; z++) {
		if (stk[z] == 'E' && stk[z+1] == '*' && stk[z+2] == 'E')
{
			stk[z] = 'E';
			stk[z+1] = '\0';
			stk[z+2] = '\0';
			printf("\n$%s\t%s$\t%s", stk, a, ac);
			i = i-2;
		}
	}
	for (z=0; z<c; z++) {
		if (stk[z] == '(' && stk[z+1] == 'E' && stk[z+2] == ')')
{
			stk[z] = 'E';
			stk[z+1] = '\0';
			stk[z+2] = '\0';
			printf("\n$%s\t%s$\t%s", stk, a, ac);
```

```
                i = i-2;

            }

        }

}
```

Output:


parallels@ubuntu-linux-22-04-desktop: ~/21BLC1607
parallels@ubuntu-linux-22-04-desktop:~/21BLC1607$ gedit lab9.c
parallels@ubuntu-linux-22-04-desktop:~/21BLC1607$ gcc -o lab9 lab9.c
lab9.c: In function ‘main’:
lab9.c:12:9: warning: implicit declaration of function ‘gets’; did you mean ‘fgets’? [-Wimplicit-function-declaration]
12 | gets(a);
| ^~~~
| fgets
/usr/bin/ld: /tmp/cch3McGZ.o: in function `main':
lab9.c:(.text+0x28): warning: the `gets' function is dangerous and should not be used.
parallels@ubuntu-linux-22-04-desktop:~/21BLC1607$ ./lab9
GRAMMAR is E->E+E
E->E*E
E->(E)
E->id
Enter input string:
id+(id*id)
stack input action
$id +(id*id)$ SHIFT->id
$E +(id*id)$ REDUCE TO E
$E+ (id*id)$ SHIFT->symbols
$E+(id*id)$ SHIFT->symbols
$E+(id *id)$ SHIFT->id
$E+(E *id)$ REDUCE TO E
$E+(E* id)$ SHIFT->symbols
$E+(E*id)$ SHIFT->id
$E+(E*E)$ REDUCE TO E
$E+(E)$ REDUCE TO E
$E+(E) $ SHIFT->symbols
$E+E $ REDUCE TO E
parallels@ubuntu-linux-22-04-desktop:~/21BLC1607$

Task 2:

Given grammar

S -> AA

A -> aA

A -> b

Input: abab

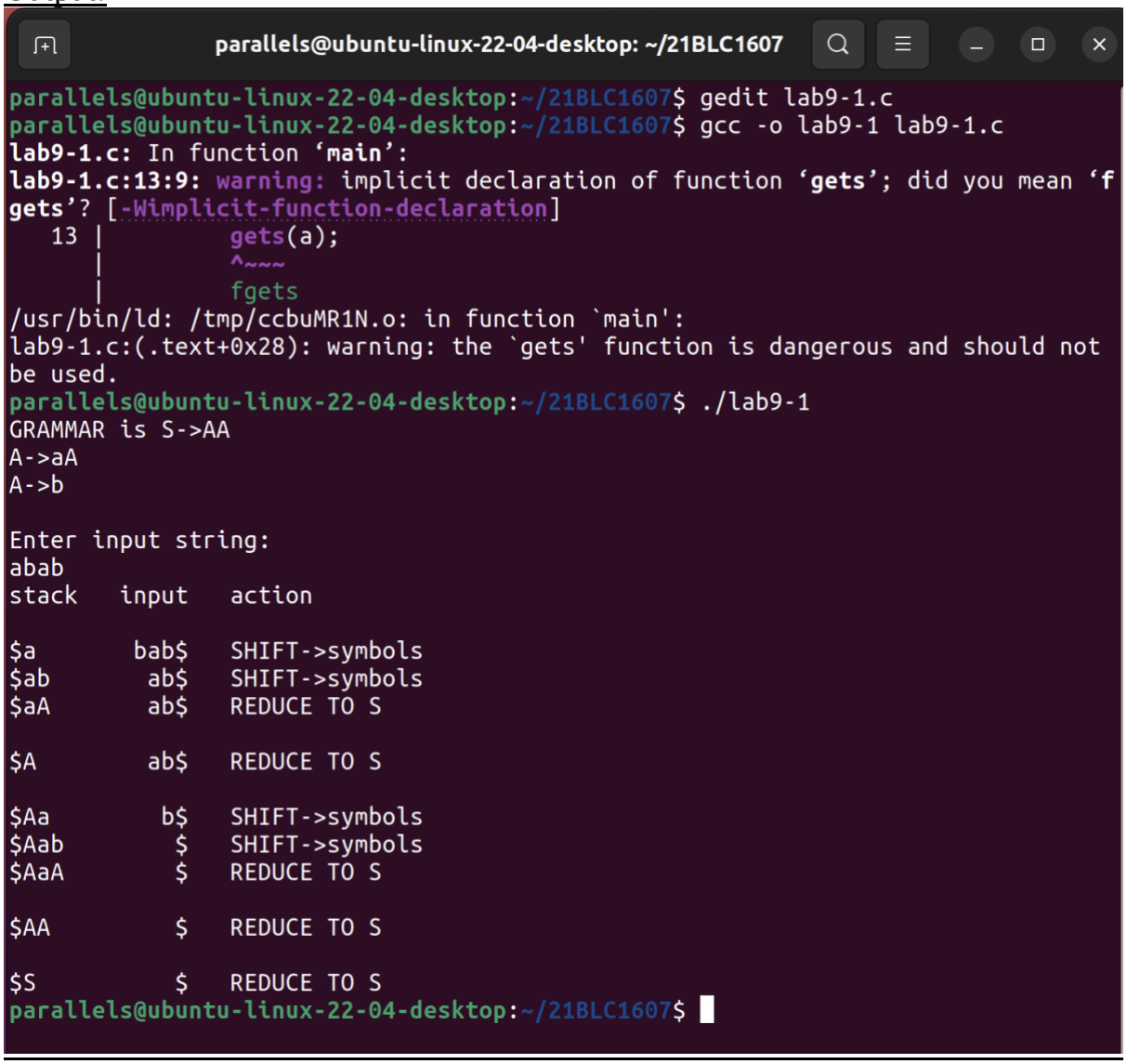
Code:

```
#include <stdio.h>
#include <string.h>

int k=0, z=0, i=0, j=0, c=0;
char a[16], ac[20], stk[15], act[10];

void check();

int main() {
	puts("GRAMMAR is S->AA\nA->aA\nA->b\n");
	// ip: abab
	puts("Enter input string: ");
	gets(a);
	c = strlen(a);
	strcpy(act, "SHIFT->");
	puts("stack\tinput\taction");
	for (k=0, i=0; j<c; k++, i++, j++) {
		if (a[j] == 'i' && a[j+1] == 'd') {
			stk[i] = a[j];
```

```
                stk[i+1] = a[j+1];
                stk[i+2] = '\0';
                a[j] = ' ';
                a[j+1] = ' ';
                printf("\n$%s\t%s$\t%sid", stk, a, act);
                check();
          }
          else {
                stk[i] = a[j];
                stk[i+1] = '\0';
                a[j] = ' ';
                printf("\n$%s\t%s$\t%ssymbols", stk, a, act);
                check();
          }
     }
}

void check() {
     strcpy(ac, "REDUCE TO S\n");
     for (z=0; z<c; z++) {
          if (stk[z] == 'b') {
                stk[z] = 'A';
                printf("\n$%s\t%s$\t%s", stk, a, ac);
          }
     }
     for (z=0; z<c; z++) {
```

```
            if (stk[z] == 'a' && stk[z+1] == 'A') {
                  stk[z] = 'A';
                  stk[z+1] = '\0';
                  printf("\n$%s\t%s$\t%s", stk, a, ac);
                  i = i-1;
            }
      }
      for (z=0; z<c; z++) {
            if (stk[z] == 'A' && stk[z+1] == 'A') {
                  stk[z] = 'S';
                  stk[z+1] = '\0';
                  printf("\n$%s\t%s$\t%s", stk, a, ac);
                  i = i-1;
            }
      }
}
```

Output:



Result:

Thus, the experiment has been successfully executed and verified.