

# System Programming

2조


권태현  
김규진  
김상준  
김영민  
김영웅

강경태 교수님

## | 역할분담

권태현	Main programmer    team leader
김상준	Main programmer
김영민	Sub programmer
김영웅	Sub programmer & Report writer
김규진	Main Report writer

## Github

 [Kwontaehwon](#) / [System\\_teamproject\\_2](#)

👁 Watch ▾0

★ Star0

🍴 Fork0

↔ Code

🔔 Issues0

🔗 Pull requests0

📁 Projects0

📖 Wiki

📊 Insights

⚙ Settings

No description, website, or topics provided.

Edit

[Manage topics](#)

📦 44 commits

🌿 2 branches

🏷 0 releases

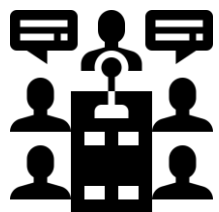
👤 4 contributors

System_teamproject_2	stack.c	28 days ago
4.c	11/28 taehwon	19 days ago
README.md	first commit	28 days ago
base.c	가장 앞의 '.'를 처리 + cal함수와 GreaterOpr함수를 스위치처리함.	3 days ago
calculator.c	파일로 입출력	15 days ago
get_input.c	get_input	27 days ago
i_of_list_clear.c	부호로 쓰인 '-'가 list->i를 늘리지 않아요!	7 days ago
intDLL.c	11/28 taehwon	19 days ago
last_calculator.c	Our last_calculator	9 hours ago
multiply_test.c	cal 스위치	3 days ago
posftfix_5.c	99%완료	15 days ago
postfix(RE).txt	Add postfix	18 days ago
postfix.c	괄호 없는 후위표기	17 days ago
postfix_2.c	11/30	17 days ago
postfix_3.c	11/30 다음	17 days ago
postfix_4.c	posftfix후위표기법 오류	17 days ago
postfix_last.c	1.후위표기법에서 괄호를 아직 없애지 못함 2.PushOrPop에서 사용하는 stack->size 를 ssize...	17 days ago
postfix_new.c	괄호를 없애는 DEL_DLL(DLL *list_1)를 구현하면 후위표기법이 끝난다	17 days ago
postpix.c	후위표기법에 괄호가 남아있음. 연산자순서는 맞음.	17 days ago
q.c	곱하기 기초구현에 요구사항이 적용됨 (main함수에서 plus_change 주석처리함)	7 days ago
ssize 추가 필요.txt	dll에 ssize를 추가해야함.	14 days ago
stack_calculator.c	stack_calculator	26 days ago
test1.c	11/28 taehwon	19 days ago
코드의 변경사항 및 알아낸 점.txt	calculator.c에 추가한 코드입니다. 텍스트 파일 확인해 주세요.	12 days ago

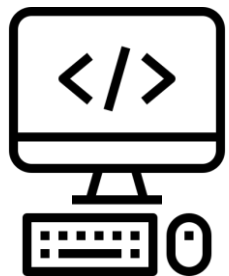
## 개발과정



1. 후위표기법 알고리즘에 대한 학습



2. 알고리즘 토의



3. 개발



1. 숫자 입력 및 후위표기법 변환



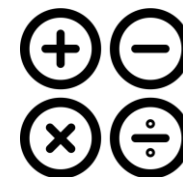
2. + 더하기 구현

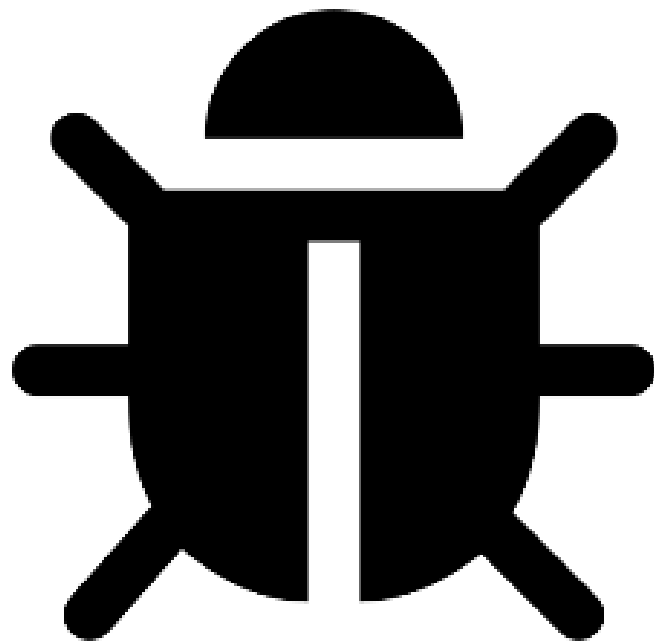


3. - 마이너스 구현



4. X 곱하기 구현





# 코드 설명



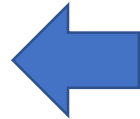
```
void G_POP(DLL *stack, DLL *list_1);
void stack_append(DLL *list, Node *newnode);
void append(DLL *list, Node *newnode);
void print(DLL *list);
void postfix(DLL *list, DLL *list_1);
void getnumber(DLL *list);
void cal(DLL *list, DLL *stack_3); // 기본적인 계산을 해주는 메인함수
void reverse(DLL *list, DLL *list_1); // 노드의 값을 반대로 다시 넣어주는 함수
int GreaterOpr(char opr1, char opr2);
void POP_all(DLL *stack, DLL *list_1);
void PushOrPop(DLL *stack, char input_opr, DLL *list_1);
void insert(DLL *list_1, DLL *list_3); // 계산한값과 기존에 있던 식을 합쳐주는 함수
void zero(DLL *stack_1, DLL *stack_2); // 소수점의 자릿수를 맞춰주는 함수
int insertAt(DLL *stack_3, int index, Node *newnode); // 특정 index에 노드를 삽입하는 함수
void copy_1(DLL *list_1, DLL *list_3); // list_1에 list_3의 노드를 복사해주는 함수
int deleteAt(DLL *list, int index);
void delete_all(DLL *list1); // 노드를 전체 비워주는 함수
void size_check(DLL *list_1); // 총 길이를 재주는 함수
void plus_zero(DLL *list_3); // .1 -.1 인 경우 .앞에 0을 추가해주는 함수
void write(DLL* list, FILE *ofp);
void free_1(DLL* stack);
```

```

int main(){
    getElapsedTime(0);
    DLL *list = newDLL(); // 입력을 받을 list
    DLL *list_2 = newDLL(); // 계산할 값을 넣어줄 list_2
    DLL *list_3 = newDLL(); // reverse한 값을 임시로 넣어줄 list_3
    getnumber(list); // 입력을 받아오는 함수
    DLL *list_1 = newDLL(); // 후위표기법으로 바뀐값을 넣어줄 list_1
    postfix(list, list_1); // 후위표기법으로 바뀐 list_1
    int a = list_1->i;
    if(list->swh == 2){ // -3+5 예제처리를 위해, 맨앞에 -나온경우 나중에 -를 붙여줌.
        list->swh = 0;
        insertAt(list_1, 0, newnode('-'));
    }
    cal(list_1, list_2); // 계산된 list_2
    reverse(list_2, list_3);
    for (int i = 1; i < a; i++){
        list_1->swh = 0; // 부호 변동을 방지
        list_2->swh = 0;
        list_3->swh = 0;
        insert(list_1, list_3); // list_1에 모든걸 저장
        size_check(list_1);
        size_check(list_3);
        delete_all(list_3);
        size_check(list_2);
        delete_all(list_2);
        cal(list_1, list_2);
        reverse(list_2, list_3);
        plus_zero(list_3);
    }
    printf("\nanswer\n");
    print(list_3);
    printf("Elapsed Time: %lld\n", getElapsedTime(1));
}

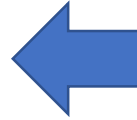
```

```
int main(){  
    getElapsedTime(0);  
    getnumber(list, ifp);  
    postfix(list, list_1);  
    cal(list_1, list_2);  
    reverse(list_2, list_3);  
}
```



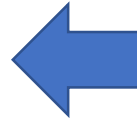
**$(233.14 + 234.3) * 9.123$**

```
int main(){  
    getElapsedTime(0);  
    getnumber(list, ifp);  
    postfix(list, list_1);  
    cal(list_1, list_2);  
    reverse(list_2, list_3);  
}
```



**$(233.14 + 234.3) * 9.123$**

```
int main(){  
    getElapsedTime(0);  
    getnumber(list, ifp);  
    postfix(list, list_1);  
    cal(list_1, list_2);  
    reverse(list_2, list_3);  
}
```



**(233.14+234.3)\*9.123**

**233.14 234.3 + 9.123 \***

```
int main(){  
    getElapsedTime(0);  
    getnumber(list, ifp);  
    postfix(list, list_1);  
    cal(list_1, list_2);  
    reverse(list_2, list_3);
```



**(233.14+234.3)\*9.123**

**233.14 234.3 + 9.123 \***

```
int main(){  
    getElapsedTime(0);  
    getnumber(list, ifp);  
    postfix(list, list_1);  
    cal(list_1, list_2);  
    reverse(list_2, list_3);
```



**(233.14+234.3)\*9.123**

**233.14 234.3 + 9.123 \***

**44.764**

```
int main(){  
    getElapsedTime(0);  
    getnumber(list, ifp);  
    postfix(list, list_1);  
    cal(list_1, list_2);  
    reverse(list_2, list_3);
```

**(233.14+234.3)\*9.123**

**233.14 234.3 + 9.123 \***

**44.764**

**467.44**





연산자(+,-,\*) 의 갯수



```
for (int i = 1 ; i < a; i++){
```

```
insert(list_1,list_3);    list_1 : 233.14 234.3 + 9.123 *
```

```
delete_all(list_3);
```

```
delete_all(list_2);
```

list\_3 : 467.44

```
cal(list_1,list_2);
```

```
reverse(list_2,list_3);
```

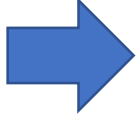
```
}
```

```
printf("\nanswer\n");
```

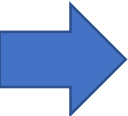
```
print(list_3);
```

```
printf("Elapsed Time: %lld\n", getElapsedTime(1));
```

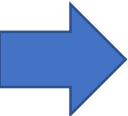
```
}
```



```
for (int i = 1 ; i < a; i++){  
    insert(list_1,list_3);    list_1 : 467.44 9.123 *  
    delete_all(list_3);  
    delete_all(list_2);  
    cal(list_1,list_2);  
    reverse(list_2,list_3);  
}  
printf("\nanswer\n");  
print(list_3);  
printf("Elapsed Time: %lld\n", getElapsedTime(1));  
}
```



```
for (int i = 1 ; i < a; i++){  
    insert(list_1,list_3);    list_1 : 467.44 9.123 *  
    delete_all(list_3);  
    delete_all(list_2);  
    cal(list_1,list_2);    list_1 : 467.44 9.123 *  
    reverse(list_2,list_3);  
}  
printf("\nanswer\n");  
print(list_3);  
printf("Elapsed Time: %lld\n", getElapsedTime(1));  
}
```



```
for (int i = 1 ; i < a; i++){
insert(list_1,list_3);    list_1 : 467.44 9.123 *
delete_all(list_3);
delete_all(list_2);
cal(list_1,list_2);  list_2 : 21554.4624
reverse(list_2,list_3); list_2 : 21554.4624
}
printf("\nanswer\n");
print(list_3);
printf("Elapsed Time: %lld\n", getElapsedTime(1));
}
```

```
for (int i = 1 ; i < a; i++){  
    insert(list_1,list_3);  
    delete_all(list_3);  
    delete_all(list_2);  
    cal(list_1,list_2);  
    reverse(list_2,list_3); list_3 : 4264.45512  
}  
printf("\nanswer\n");  
print(list_3);  
printf("Elapsed Time: %lld\n", getElapsedTime(1));  
}
```

```
for (int i = 1 ; i < a; i++){  
    insert(list_1,list_3);  
    delete_all(list_3);  
    delete_all(list_2);  
    cal(list_1,list_2);  
    reverse(list_2,list_3);  
}
```

```
printf("\nanswer\n");
```

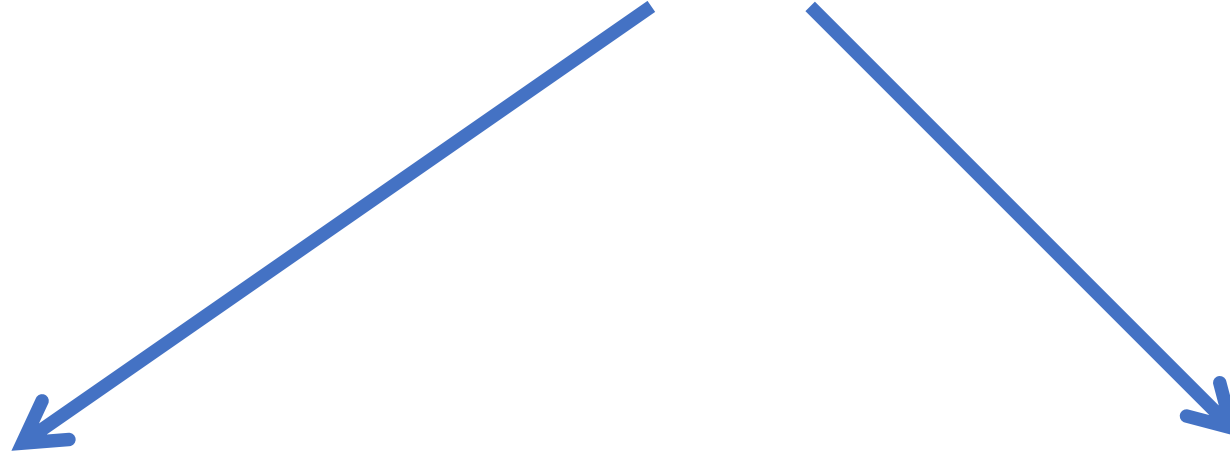
```
print(list_3);
```

**list\_3 : 4264.45512**

```
printf("Elapsed Time: %lld\n", getElapsedTime(1));  
}
```

```
cal(list_1,stack_3);
```

17 123 +



Stack\_1

Stack\_2

17

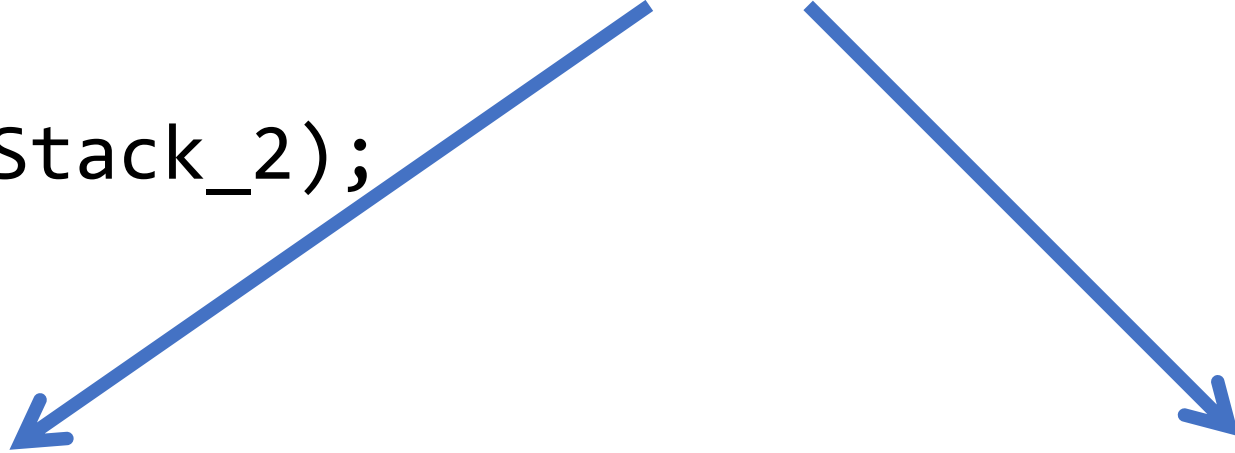
123

cal(list\_1,stack\_3);



Zero(Stack\_1,Stack\_2);

17 123 +



Stack\_1

Stack\_2

17

123



```
cal(list_1, stack_3);
```



```
Zero(Stack_1, Stack_2);
```

Stack\_1

**17**



Stack\_2

**123**

```
cal(list_1, stack_3);
```



```
Zero(Stack_1, Stack_2);
```

Stack\_1

**123**

Stack\_2

**017**



```
cal(list_1, stack_3);  
Switch( '+' )
```



Count = 0

123  
+017

---

3+7 = 10

```
cal(list_1, stack_3);  
Switch( '+' )
```



Count = 0

123  
+017

---

Count++

3+7 = 10

10 - 10 = 0

```
cal(list_1, stack_3);  
Switch( '+' )
```



Count = 1

123  
+017



0

```
cal(list_1, stack_3);  
Switch('+')
```



Count = 1

123  
+017

---

2+1 = 3      0

```
cal(list_1, stack_3);  
Switch(' + ')
```



Count = 0

123  
+017

---

2 + 1 = 3      0

3 + 1 = 4

```
cal(list_1, stack_3);  
Switch( '+' )
```



Count = 0

123  
+017



40



```
cal(list_1, stack_3);  
Switch( '+' )
```



Count = 0

123  
+017



40

```
cal(list_1, stack_3);  
Switch( '+' )
```



Count = 0

123  
+017

---

1 + 0 = 1    40

```
cal(list_1, stack_3);  
Switch( '+' )
```



Count = 0

123  
+017



140

```
cal(list_1,stack_3);
```

**list\_2 :**      **041**

```
reverse(list_2,list_3);
```

**list\_3 : 140**

```
cal(list_1, stack_3);  
Switch('-',)
```



Count = 0

123  
- 017

---

3-7 = -4

```
cal(list_1, stack_3);  
Switch('-',)
```



Count = 1

123  
- 017

---

$$3 - 7 = -4$$

$$-4 + 10 = 6$$

```
cal(list_1, stack_3);  
Switch('-',)
```



Count = 1

$$\begin{array}{r} 123 \\ - 017 \\ \hline 6 \end{array}$$



```
cal(list_1, stack_3);  
Switch('-',)
```



Count = 1

123  
- 017

---

2-1=1

6

```
cal(list_1, stack_3);  
Switch('-',)
```



Count = 0

123  
- 017

---

2-1=1

1-1=0

6

```
cal(list_1, stack_3);  
Switch('-',)
```



Count = 0

123  
- 017

---

06

1-1=0

```
cal(list_1, stack_3);
```

```
Switch('-',)
```

```
Reverse(list_2, list_3)
```



Count = 0

123

- 017



106

1-0=1

```
cal(list_1, stack_3);  
Switch( '*' )
```

**list\_1 : 123 17 \***

**Stack\_3 : 0**

```
cal(list_1, stack_3);  
Switch('*)
```

**Count = 0**

**Stack\_3 = 0**

123  
X 017

---

**3X7=21**

```
cal(list_1, stack_3);  
Switch('*')
```

Count = 2  
Stack\_3 = 0

123  
X 017

---

3X7=21

21-20=1    Count + 2

```
cal(list_1, stack_3);  
Switch('*)
```

**Count = 2**  
**Stack\_3 = 0**

123  
X 017  
-----  
1



```
cal(list_1, stack_3);  
Switch( '*' )
```

$$\begin{array}{r} 123 \\ \times 017 \\ \hline 1 \end{array}$$

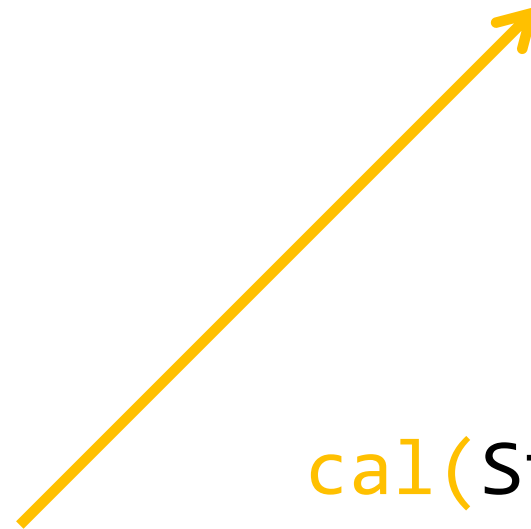
Count = 2  
Stack\_3 = 0

Stack\_4 : 1 0 +

```
cal(list_1, stack_3);  
Switch( '*' )
```

$$\begin{array}{r} 123 \\ \times 017 \\ \hline 1 \end{array}$$

Count = 2  
Stack\_3 = 0



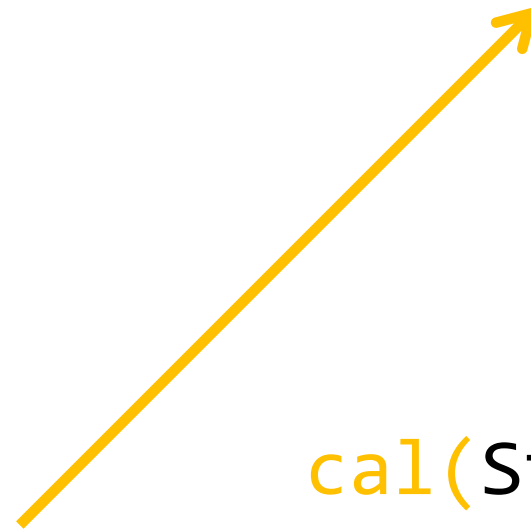
```
cal(Stack_3, Stack_4);
```

Stack\_4 : 1 0 +

```
cal(list_1, stack_3);  
Switch( '*' )
```

$$\begin{array}{r} 123 \\ \times 017 \\ \hline 1 \end{array}$$

Count = 2  
Stack\_3 = 1



```
cal(Stack_3, Stack_4);
```

Stack\_4 : 1 0 +

```
cal(list_1, stack_3);  
Switch('*')
```

Count = 2

123  
X 017

---

2X7=14    14+Count = 16

```
cal(list_1, stack_3);  
Switch( '*' )
```

Count = 0

123  
X 017

---

2X7=14    14+Count = 16

16-10=6    Count + 1

```
cal(list_1, stack_3);  
Switch('*')
```

Count = 1

$$\begin{array}{r} 123 \\ \times 017 \\ \hline 61 \end{array}$$

```
cal(list_1, stack_3);  
Switch( '*' )
```

$$\begin{array}{r} 123 \\ \times 017 \\ \hline 61 \end{array}$$

Count = 1

int Stack\_3 = 1

Stack\_4 : 60 1 +

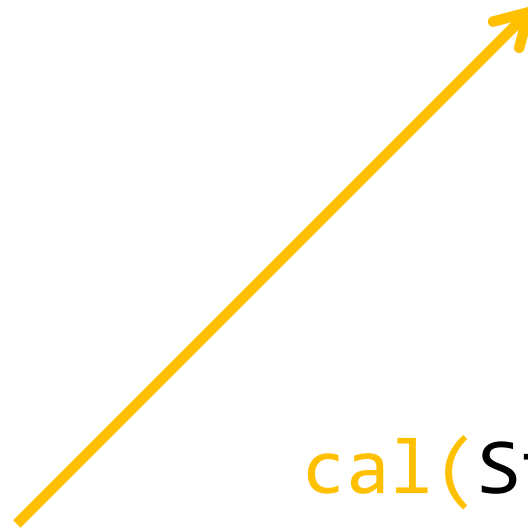
```
cal(list_1, stack_3);  
Switch( '*' )
```

$$\begin{array}{r} 123 \\ \times 017 \\ \hline 61 \end{array}$$

Count = 1  
Stack\_3 = 1

Stack\_4 : 60 1 +

cal(Stack\_3, Stack\_4);





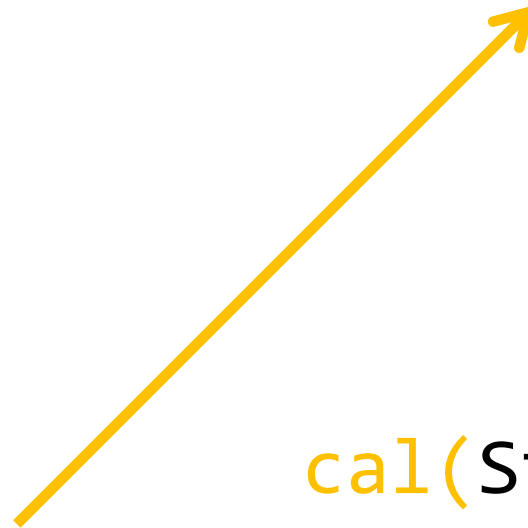
```
cal(list_1, stack_3);  
Switch( '*' )
```

$$\begin{array}{r} 123 \\ \times 017 \\ \hline 61 \end{array}$$

Count = 1  
Stack\_3 = 61

Stack\_4 : 60 1 +

cal(Stack\_3, Stack\_4);



```
cal(list_1, stack_3);  
Switch( '*' )
```

Count = 0  
Stack\_3 = 861

$$\begin{array}{r} 123 \\ \times 017 \\ \hline 861 \end{array}$$

```
cal(list_1,stack_3);
```

```
Switch('*')
```

$$\begin{array}{r} 123 \\ \times 017 \\ \hline 861 \\ + 1230 \end{array}$$

```
cal(list_1, stack_3);
```

```
Switch('*')
```

123

X 017

---

2091

Stack\_3 = 2091