1. 프로그램 소스코드

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#define MAX 64

int num;
int arrNum = 0;
int isFloat = 0;
int floatPoint = 0;
int triggerMix = 0;
float floatNum = 0.0;
char ch = ' ';
char arr[MAX];

typedef enum TOKEN {null, NUMBER, PLUS, STAR, LP, RP, END} To;
typedef enum TYPE {INTTYPE, FLOATTYPE} Ty;
typedef union UNITYPE { int i; float f;} Uni;
typedef struct {
        Ty type;
        Uni value;
}st;

To token;

void get_token();
st* expression();
st* term();
st* factor();
void error(int i);

void get_token(){
        ch = arr[arrNum];
        if(ch == '+'){
                token = PLUS;
        }
```

```
        else if(ch == '*'){
                token = STAR;
        }
        else if(ch == '('){
                token = LP;
        }
        else if(ch == ')'){
                token = RP;
        }
        else if(ch == '\n'){
                token = END;
        }
        else if((ch >= '0') && (ch <= '9')){
                isFloat = 0;
                floatNum = 0.0;
                floatPoint = 0;
                token = NUMBER;
                num = ch - '0';
                while(((arr[arrNum+1] >= '0') && (arr[arrNum+1] <= '9')) || arr[arrNum+1]
== '.')
                {
                        if(arr[arrNum+1] == '.'){
                                isFloat = 1;
                                floatNum = num;
                                arrNum++;
                                continue;
                        }
                        if(isFloat == 1){
                                arrNum++;
                                ch = arr[arrNum];
                                floatPoint++;
                                floatNum = floatNum + (ch-'0')*pow(0.1,floatPoint);
                        }
                        else{
                                arrNum++;
                                ch = arr[arrNum];
                                num = num*10 + (ch - '0');
```

```
                }
            }
        }
        else{
            token = null;
        }
        arrNum++;
}

st* expression(){
        st* result;
        result = term();
        while(token==PLUS)
        {
            if((result->type == FLOATTYPE) && (triggerMix == 0)){
                triggerMix = 1;
            }
            get_token();
            st* temp = term();
            if(temp->type == INTTYPE){
                if(triggerMix>0){
                    triggerMix = 2;
                    temp->type = FLOATTYPE;
                    temp->value.f = temp->value.i;
                }
                else{
                    result->type = INTTYPE;
                    result->value.i = result->value.i + temp->value.i;
                }
            }
            if(temp->type == FLOATTYPE){
                if((triggerMix==0) || (result->type == INTTYPE)){
                    triggerMix = 2;
                    result->value.f = result->value.i;
                }
                result->type = FLOATTYPE;
                result->value.f = result->value.f + temp->value.f;
```

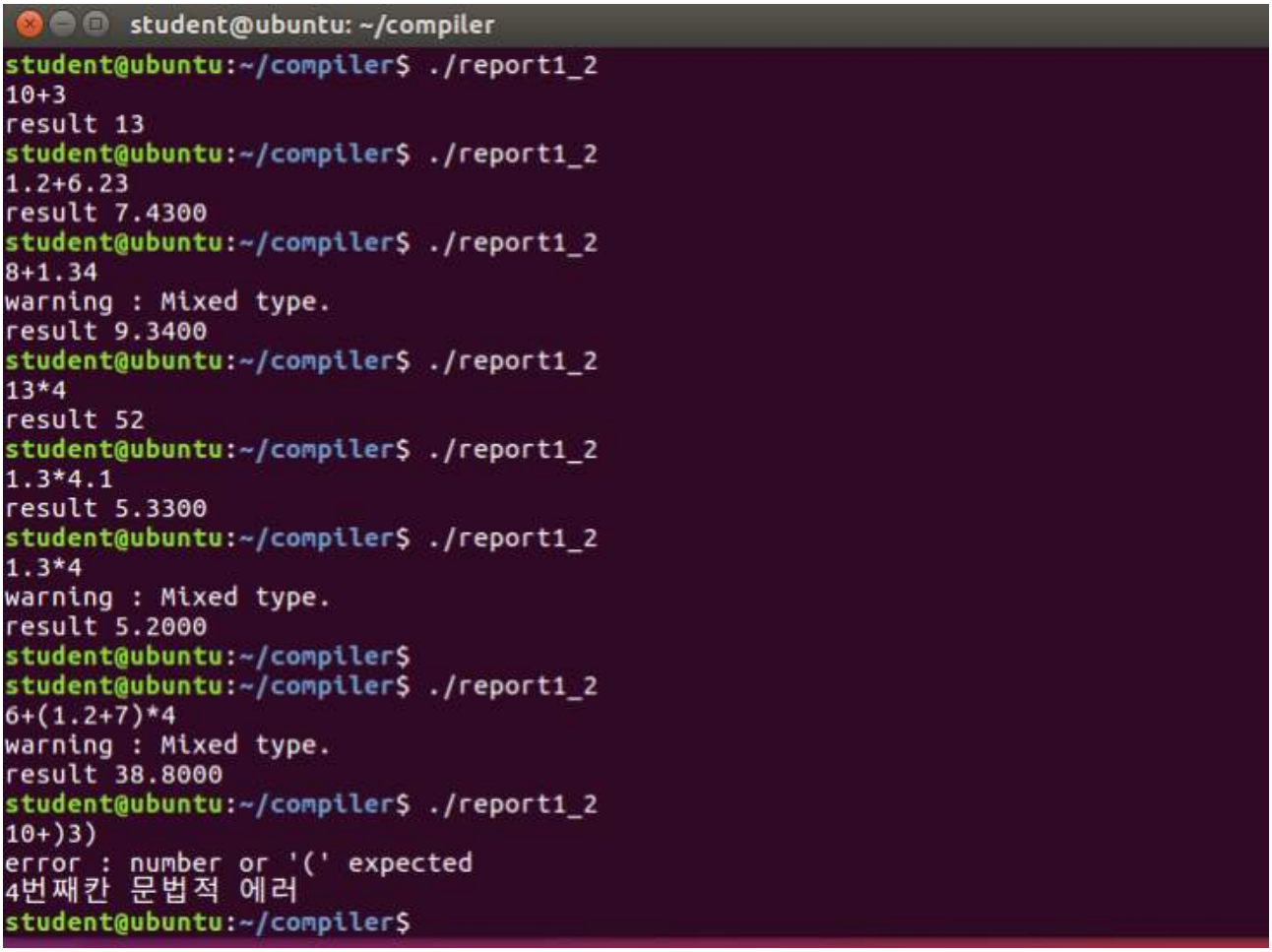
```
                }
                free(temp);
        }
        return (result);
}

st* term(){
        st* result;
        result=factor();
        while(token==STAR)
        {
                if((result->type == FLOATTYPE) && (triggerMix == 0))
                        triggerMix = 1;
                get_token();
                st* temp = factor();
                if(temp->type == INTTYPE){
                        if(triggerMix > 0){
                                triggerMix = 2;
                                temp->type = FLOATTYPE;
                                temp->value.f = temp->value.i;
                        }
                        else{
                                result->type = INTTYPE;
                                result->value.i = result->value.i * temp->value.i;
                        }
                }
                if(temp->type == FLOATTYPE){
                        if((triggerMix == 0) || (result->type == INTTYPE)){
                                result->value.f = result->value.i;
                                triggerMix = 2;
                        }
                        result->type = FLOATTYPE;
                        result->value.f = result->value.f * temp->value.f;
                }
                free(temp);
        }
        return (result);
```

```
}

st* factor(){
	st* result = (st*)malloc(sizeof(st));
	if(token == NUMBER)
	{
		if(isFloat == 1){
			result->value.f = floatNum;
			result->type = FLOATTYPE;
		}
		else{
			result->value.i = num;
			result->type = INTTYPE;
		}
		get_token();
	}
	else if(token==LP){
		get_token();
		result=expression();
		if(token == RP)
			get_token();
		else
			error(2);
	}
	else
		error(1);
	return (result);
}

void main(){
	st* result;
	int i = 0;
	while(i<MAX){
		arr[i] = getchar();
		if(arr[i] == '\n')
			break;
		i++;
```

```
        }
        get_token();
        result=expression();
        if(token!=END){
                error(3);
        }
        else{
                switch(triggerMix){
                        case 0:
                                printf("result %d\n", result->value.i);
                                break;
                        case 1:
                                printf("result %.4f\n", result->value.f);
                                break;
                        case 2:
                                error(4);
                                printf("result %.4f\n", result->value.f);
                                break;
                }
        }
}

void error(int i){
        switch(i)
        {
                case 1:
                        printf("error : number or '(' expected\n");
                        printf("%d번째칸 문법적 에러\n", arrNum);
                        exit(1);
                        break;
                case 2:
                        printf("error : ')' expected\n");
                        printf("%d번째칸 문법적 에러\n", arrNum);
                        exit(1);
                        break;
                case 3:
                        printf("error : EOF expected\n");
```

```
                exit(1);
                break;
        case 4:
                printf("warning : Mixed type.\n");
                break;
    }
}
```

2. 실행 결과


student@ubuntu: ~/compiler
student@ubuntu:~/compiler$ ./report1_2
10+3
result 13
student@ubuntu:~/compiler$ ./report1_2
1.2+6.23
result 7.4300
student@ubuntu:~/compiler$ ./report1_2
8+1.34
warning : Mixed type.
result 9.3400
student@ubuntu:~/compiler$ ./report1_2
13*4
result 52
student@ubuntu:~/compiler$ ./report1_2
1.3*4.1
result 5.3300
student@ubuntu:~/compiler$ ./report1_2
1.3*4
warning : Mixed type.
result 5.2000
student@ubuntu:~/compiler$
student@ubuntu:~/compiler$ ./report1_2
6+(1.2+7)*4
warning : Mixed type.
result 38.8000
student@ubuntu:~/compiler$ ./report1_2
10+)3)
error : number or '(' expected
4번째칸 문법적 에러
student@ubuntu:~/compiler$