

int main (argc, argv)

- 예외처리
- 동적할당
- 정수변환
- 계산
- 배열 출력
- 할당해제

int ft_handle_exception (char * str, int argc)

예외처리 하는 함수

입력 : 인자로 들어오는 문자열, 인자 개수

return : 예외처리 결과

0 : Error

1 : Success

- (예외)
- 배열길이가 != 16일 때
 - 정수가 1 ≤ ≤ 4 범위인지
 - argc != 2
 - " " 공백이 아닌 경우
- } 출력

int ** ft_atoi (char * str)

문자열 자르고 정수 배열에 저장하는 함수

입력 : 인자로 들어오는 문자열
"col1up col2up ..."

return : 배열의 주소값

void ft_print_box (int ** ans)

답을 출력하는 함수

```
| v | v | v | /n
| v | v | v | /n
| v | v | v | /n
| v | v | v | /n
```

void ft_stack_box (int index , int ** udlr , int * ans)

상자 쌓는 함수 (재귀로 구현)

→ 옳은 자리인지 확인 : ft_check (index , udlr , ans)

void ft-check (int index , int ** udlr , int * ans)

상자 개수가 맞는지 확인하는 함수

- 상자 개수를 udlr 과 비교
- 중복이 있는지 확인



void ft-free (udlr , ans)

메모리 할당 해제하는 함수

```
int main ( int argc, char* argv[] )
```

메인함수

```
int result  
int ** udlr  
int * ans
```

입력 여의치리

```
if ( ft_handle_exception (argc, argv[1]) == 0 )  
    return 0
```



입력 정수로 변경해서 저장

```
ans = (int *) malloc ( sizeof(int) * 16 )
```

```
udlr = ft_atoi( argv[1], udlr )
```



계산

```
result = ft_stack_box(0)
```



배열 출력

```
if (result)  
    ft_print_box(ans)  
else  
    write("Error")
```



해당 해제

```
ft_free( udlr, ans )
```