

【수업 준비】

1. 바탕화면 오른쪽 위 **출석확인** : 자기이름 쓰기

2. 깃허브 접속하기

<https://github.com/swKyungbock>

3. 깃허브 링크 **진단평가** 풀기

4. 깃허브에서 **수업자료** 살펴보기

['정보처리와 관리' 평가계획]

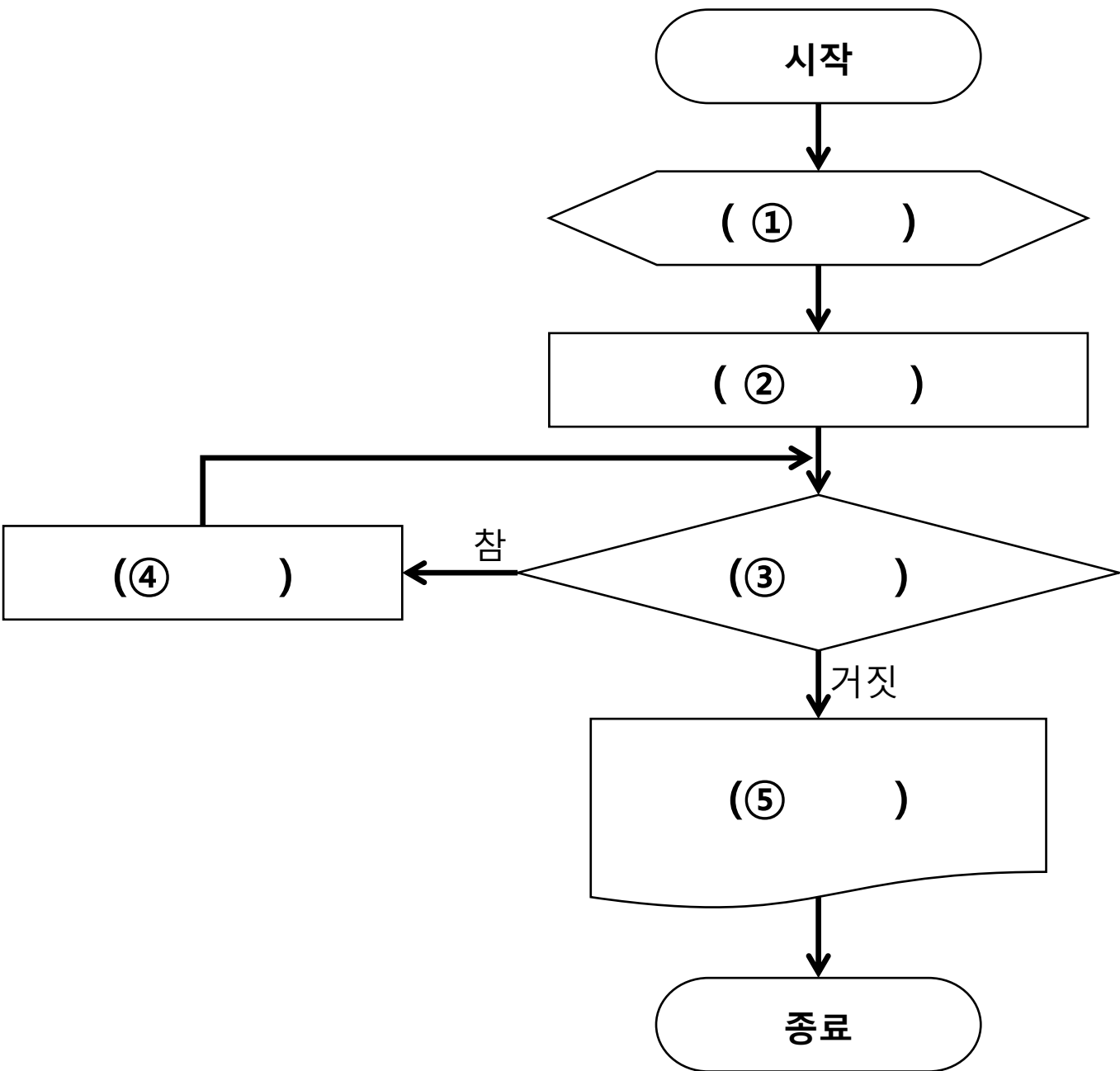
평가항목	배점	평가내용	평가준거
프로젝트	40	<ul style="list-style-type: none"> - 계획 문제 발견, 데이터 수집, 문제 정의 - 완성 지속적인 개선 과정, 협력적 문제해결, 자기주도적 문제해결, 알고리즘 설계, 데이터 구조 설계, 과정에 대한 기록 - 발표 및 보고서 설계하고 완성시킨 프로젝트의 효율적 표현법 	<ul style="list-style-type: none"> - A(40점) : 프로젝트의 계획 및 설계, 프로젝트 완성, 발표 및 보고서를 작성하는 모든 과정을 완벽히 수행함. - B(38점) : 프로젝트의 계획 및 설계, 프로젝트 완성, 발표 및 보고서를 작성하는 과정을 80% 이상 수행함. - C(36점) : 프로젝트의 계획 및 설계, 프로젝트 완성, 발표 및 보고서를 작성하는 과정을 60% 이상 수행함. - D(34점) : 프로젝트의 계획 및 설계, 프로젝트 완성, 발표 및 보고서를 작성하는 과정을 40% 이상 수행함. - E(32점) : 프로젝트의 계획 및 설계, 프로젝트 완성, 발표 및 보고서를 작성하는 과정을 20% 이상 수행함. - F(30점) : 프로젝트의 계획 및 설계, 프로젝트 완성, 발표 및 보고서를 작성하는 과정을 20% 미만 수행함. <p>※ 프로젝트 미수행 학생 0점 처리 가능</p>
문제해결실습	40	-다양한 문제 상황이 주어졌을 때 컴퓨팅적 사고를 통한 문제해결	<ul style="list-style-type: none"> - A(40점) : 제시한 문제 상황을 완벽하게 이해하여 해결함 - B(38점) : 제시한 문제 상황에 대해 80% 이상 해결함 - C(36점) : 제시한 문제 상황에 대해 60% 이상 해결함 - D(34점) : 제시한 문제 상황에 대해 40% 이상 해결함 - E(32점) : 제시한 문제 상황에 대해 20% 이상 해결함 - F(30점) : 제시한 문제 상황에 대해 20% 미만 해결함 <p>※ 실습 미수행 학생 0점 처리 가능</p>
수업참여도	20	<ul style="list-style-type: none"> - 교재준비 미비 • 무단결과 - 휴대전화, MP3등 전자기기 이용 - 수업에 방해되는 행동 및 잠을 잘 경우 - 교사의 지시 사항을 따르지 않는 경우 	누가 기록하여 환산 후 감점

[지난 시간 배운 내용 : RUR-PLE]

문제를 풀어 보세요!

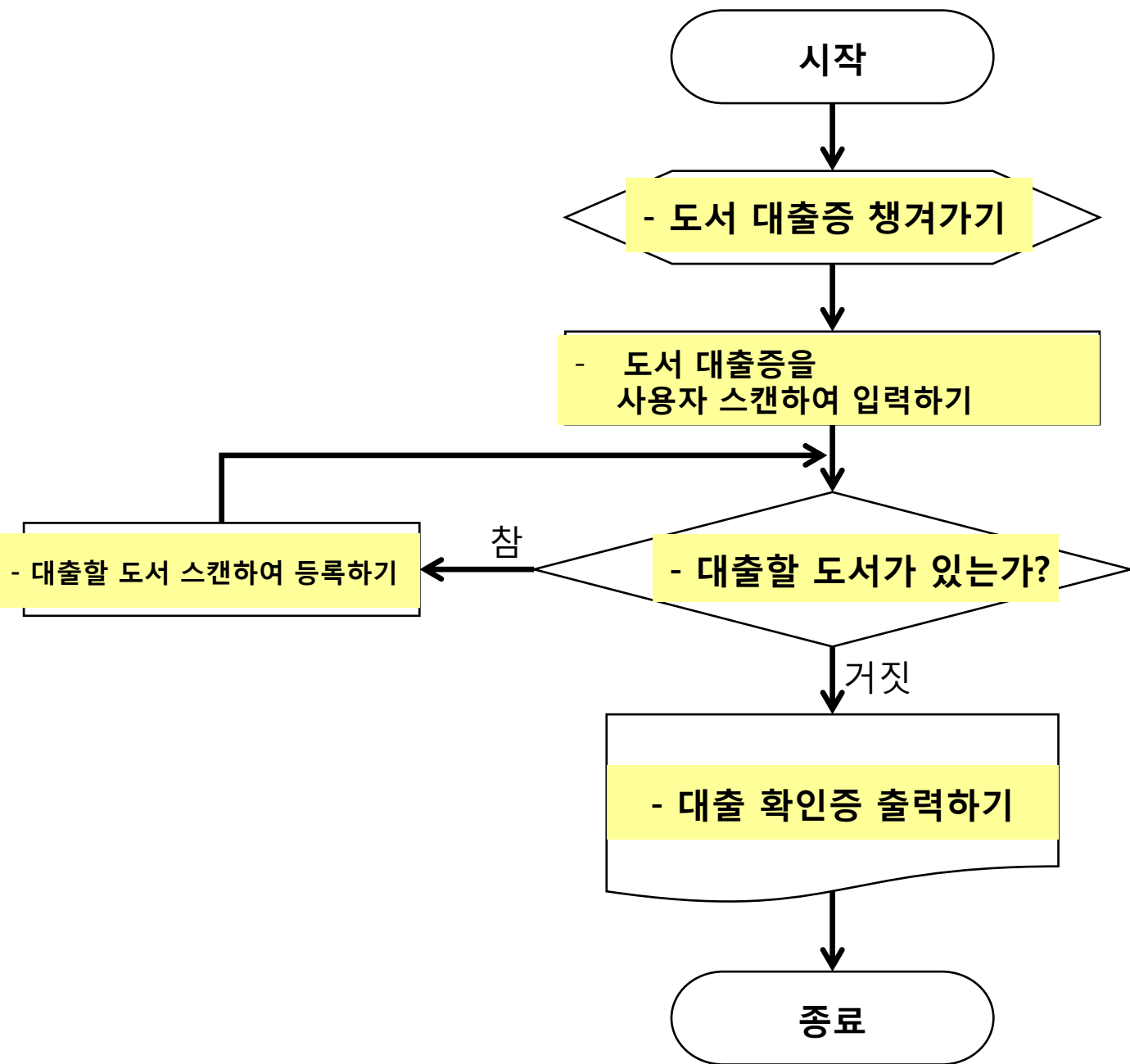
<러플 비퍼, 벽 다루기 진단평가> :

<https://forms.gle/6oUoshQWRDmJxWda8>



<도서대출하기>

- 도서 대출증을 사용자 스캔하여 입력하기
- 도서 대출증 챙겨 가기
- 대출할 도서가 있는가?
- 대출 확인증 출력하기
- 대출할 도서 스캔하여 등록하기



<도서대출하기>

[Python? RUR-PLE?]

"Life is too short, You need python."
(인생은 너무 짧으니, 파이썬이 필요해.)

-Guido van Rossum

[Python? RUR-PLE?]

Rossum's Universal Robots
– *Python Learning Environment*

【지난 시간 설문】 #가장 어려웠던 점?



【오늘의 수업 목표!】

똑같은 명령어 반복을 없애보자!

[오늘의 수업내용 : RUR-PLE(러플)]

#1. RUR-PLE(러플) 사용자 정의 함수

#2. 사용자 정의 함수를 활용한 프로그래밍

- 1) 시계방향으로 사각형을 그리며 도는 로봇프로그래밍**
- 2) 뒤로 도는 로봇프로그래밍**
- 3) 사각형 벽을 반시계방향으로 돌아 나오는 로봇프로그래밍**
- 4) 비퍼를 놓았다가 수거하는 로봇 프로그래밍**
- 5) 허들을 넘는 로봇 프로그래밍**

【파이썬의 함수】

내장 함수 : **move(), turn_left(), turn_off()**
pick_beeper(), put_beeper() 등

사용자 정의 함수 : 얼마든지 “내가” 새로 만들어요!

【사용자 정의 함수】 : 내 마음대로 함수를 만들어요!!

사용자 정의 함수 만드는 형식

def 사용자_함수_이름() :

명령-1

명령-2

....

[사용자 정의 함수] : 내 마음대로 함수를 만들어요!!

사용자 정의 함수 만드는 형식

def 사용자_함수_이름() :

① : (콜론)을 추가해요!

명령-1

명령-2

② : 공백4칸
들여쓰기의 칸은 꼭 똑같이!

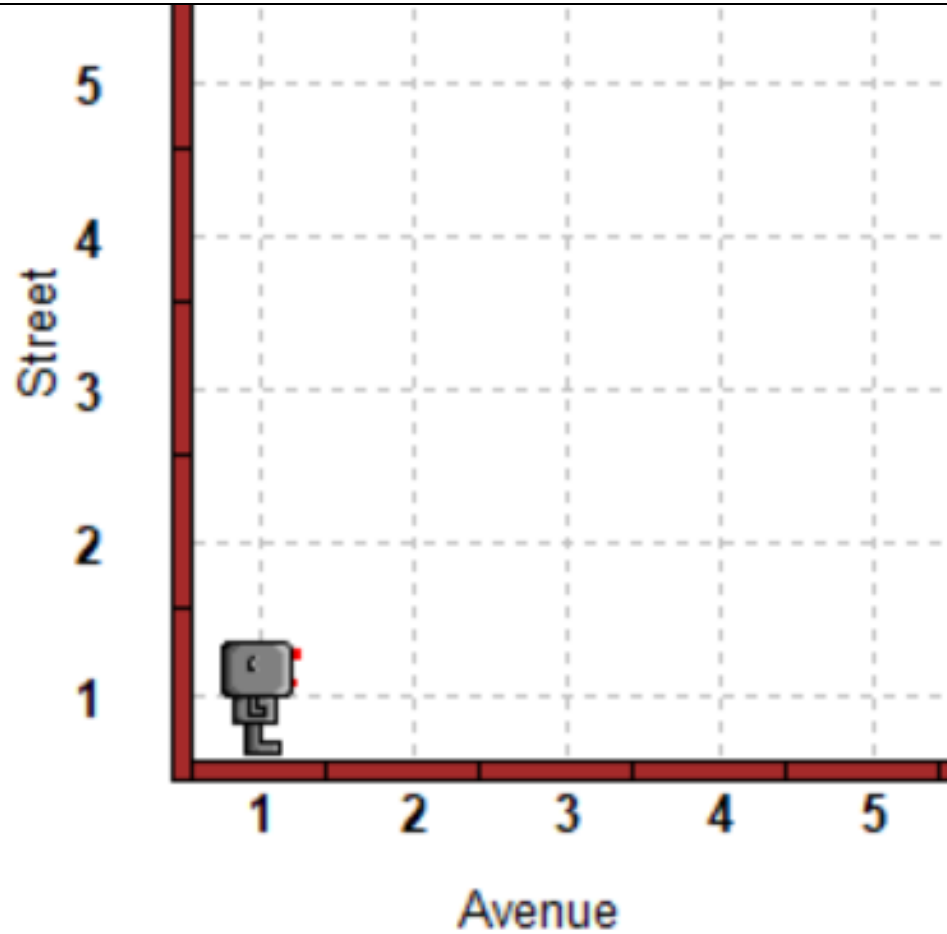
....

【사용자 정의 함수】 : 함수 이름 만드는 규칙!

- 1) 함수 이름은 알파벳, 숫자, _(언더바)만으로 정의**
- 2) 함수 이름은 숫자로 시작할 수 없어요!**
- 3) 함수 이름에 공백은 안되요!**
- 4) 러플에서 이미 사용하는 예약어(def, if, while등)는 사용할 수 없어요!**

【사용자 정의 함수】 : 왼쪽으로 3번 돌기 ??

```
1 turn_left()  
2 turn_left()  
3 turn_left()  
4  
5 turn_off()  
6
```



【사용자 정의 함수】 : turn_right()를 만들어 보세요!

```
turn_left()
```

```
turn_left()
```

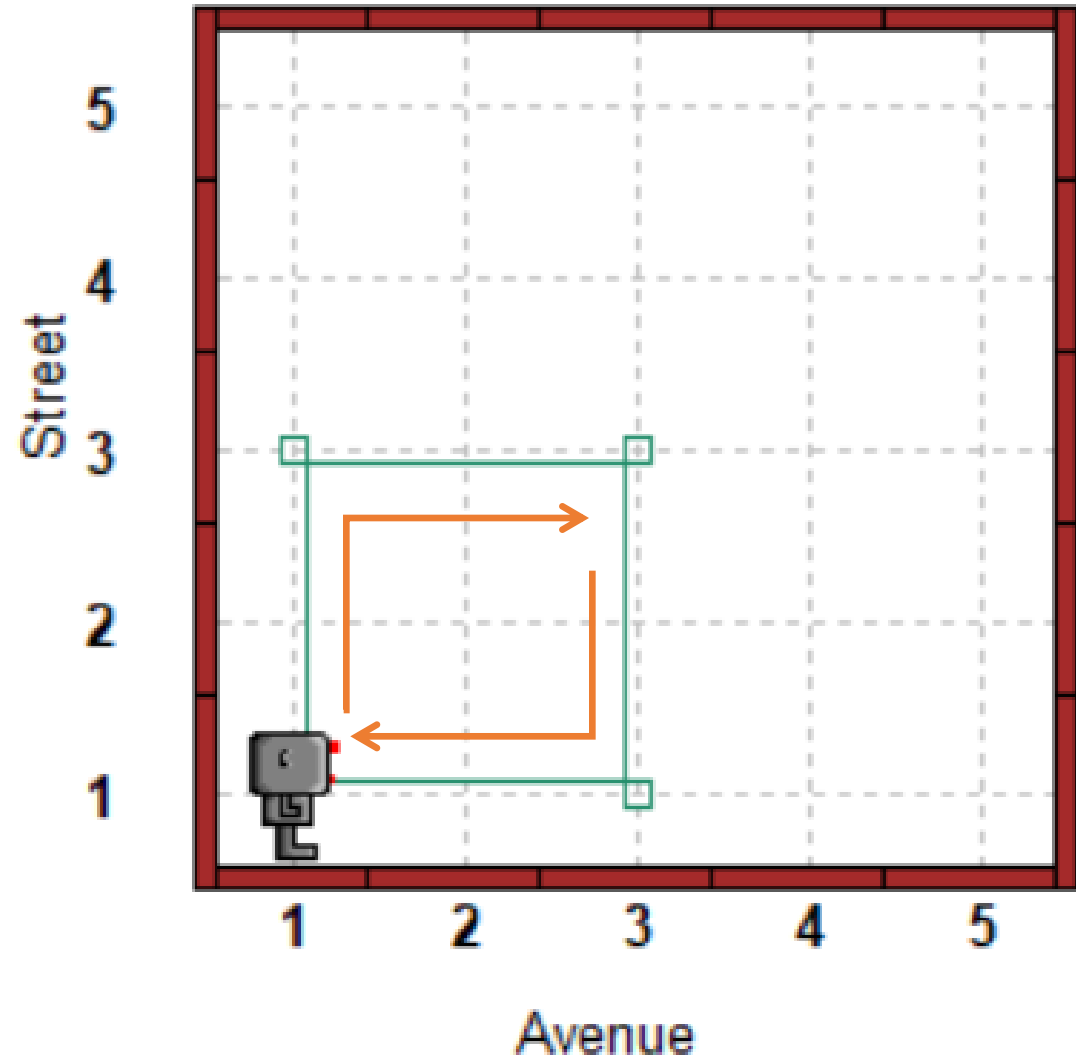
```
turn_left()
```


[사용자 정의 함수] : turn_right()를 만들어 보세요!

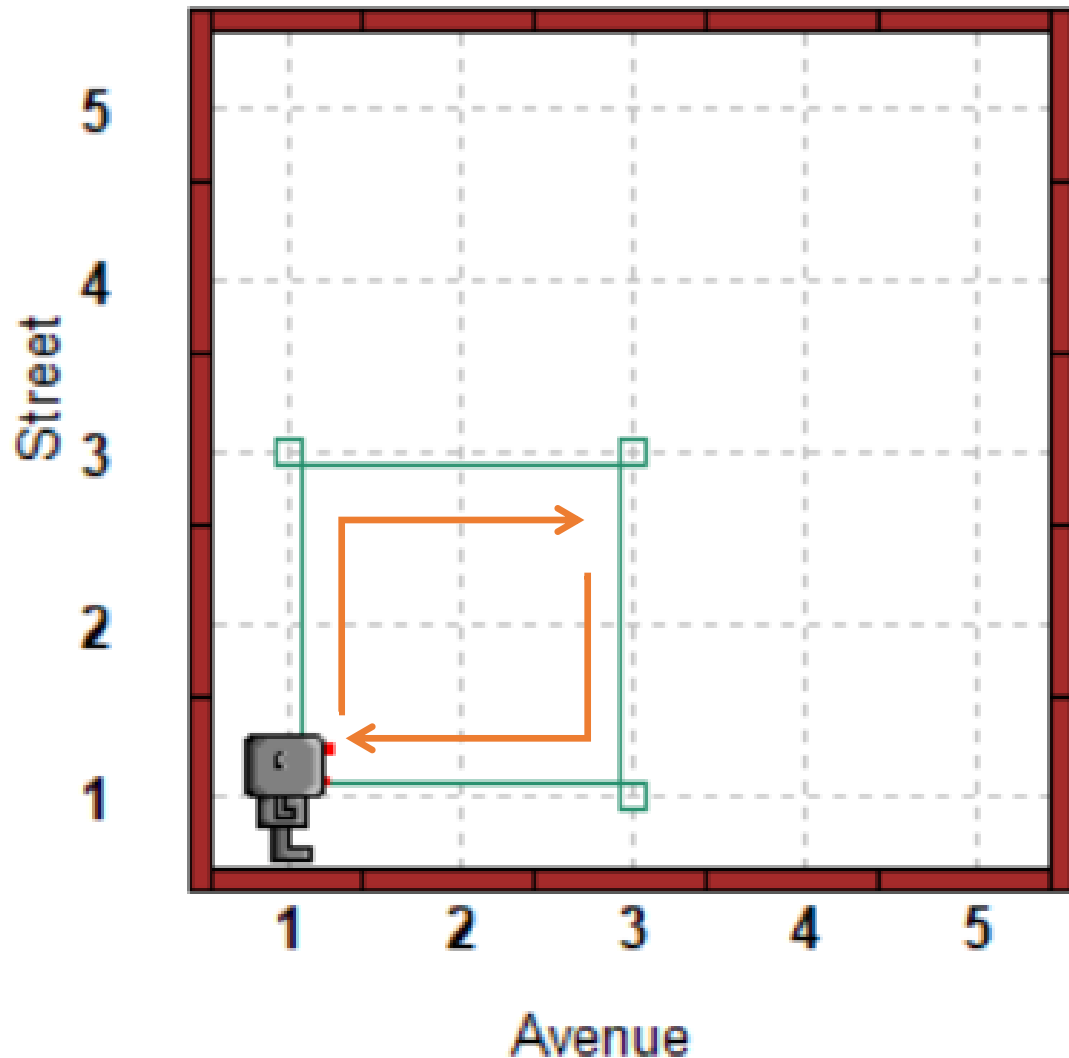
```
def turn_right( ) :
```

```
    turn_left( )  
    turn_left( )  
    turn_left( )
```

[Ex1] : turn_right()를 이용해 로봇이
시계방향으로 사각형을 그리며 이동하도록 프로그램을
완성해 보세요.



[Ex1] : 완성



```
#Ex1.  
#define turn_right  
def turn_right():  
    turn_left()  
    turn_left()  
    turn_left()  
#use turn_right()  
turn_left()  
move()  
move()  
turn_right()  
move()  
move()  
turn_right()  
move()  
move()  
turn_left()  
turn_left()  
turn_off()
```

[Ex1]turn_right()를 정의하기 전과 비교해 볼까요?

<함수를 사용하지 않았을 때>

```
1 move()
2 turn_left()
3 move()
4 turn_left()
5 turn_left()
6 turn_left()
7 move()
8 move()
9 turn_left()
10 turn_left()
11 turn_left()
12 move()
13 turn_left()
14 move()
15 turn_off()
```

<함수를 사용했을 때>

```
1 def turn_right() :
2     ... turn_left()
3     ... turn_left()
4     ... turn_left()
5 move()
6 turn_left()
7 move()
8 turn_right()
9 move()
10 move()
11 turn_right()
12 move()
13 turn_left()
14 move()
15 turn_off()
```

[Ex1]turn_right()를 정의하기 전과 비교해 볼까요?

<함수를 사용하지 않았을 때>

```
1 move()
2 turn_left()
3 move()
4 turn_left()
5 turn_left()
6 turn_left()
7 move()
8 move()
9 turn_left()
10 turn_left()
11 turn_left()
12 move()
13 turn_left()
14 move()
15 turn_off()
```

<함수를 사용했을 때>

```
1 def turn_right() :
2     ... turn_left()
3     ... turn_left()
4     ... turn_left()
5 move()
6 turn_left()
7 move()
8 turn_right()
9 move()
10 move()
11 turn_right()
12 move()
13 turn_left()
14 move()
15 turn_off()
```

[Ex1] 함수 정의 Vs. 함수 호출

함수정의

```
1 def turn_right() :  
2     ... turn_left()  
3     ... turn_left()  
4     ... turn_left()  
5 move()  
6 turn_left()  
7 move()  
8 turn_right()  
9 move()  
10 move()  
11 turn_right()  
12 move()  
13 turn_left()  
14 move()  
15 turn_off()
```

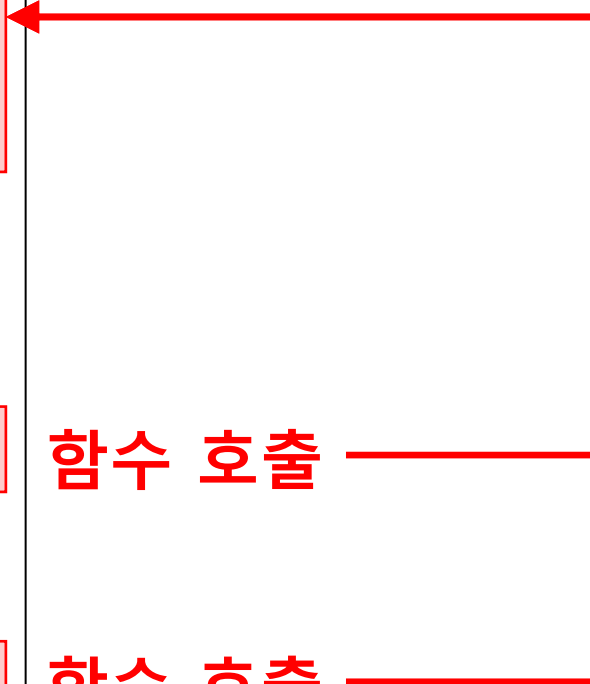
[Ex1] 함수 정의 Vs. 함수 호출

함수정의

```
1 def turn_right() :  
2     ... turn_left()  
3     ... turn_left()  
4     ... turn_left()  
5 move()  
6 turn_left()  
7 move()  
8 turn_right()  
9 move()  
10 move()  
11 turn_right()  
12 move()  
13 turn_left()  
14 move()  
15 turn_off()
```

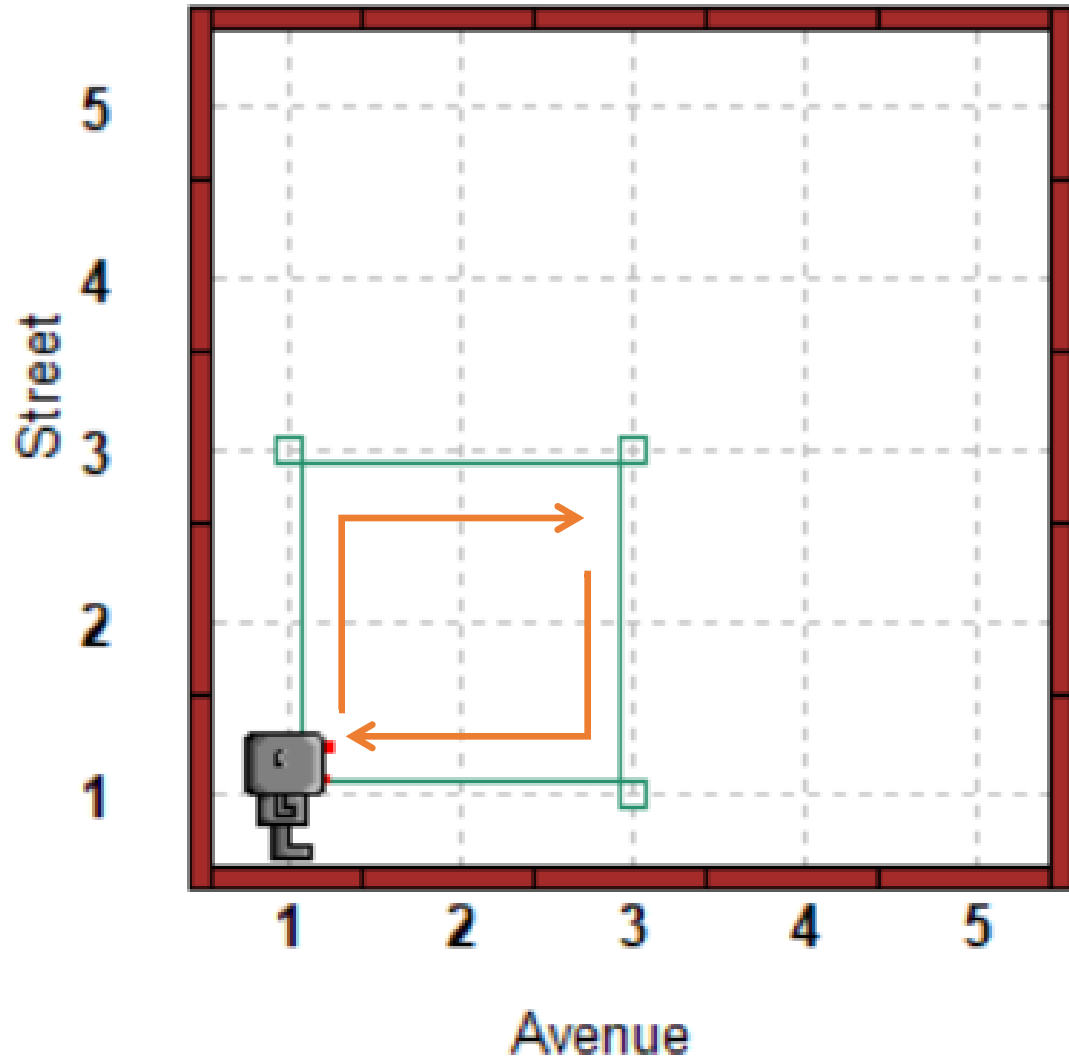
함수 호출

함수 호출



[Ex1] : 하나만 더!

함수정의와 함수호출의 순서가 바뀌면 어떻게 될까요?



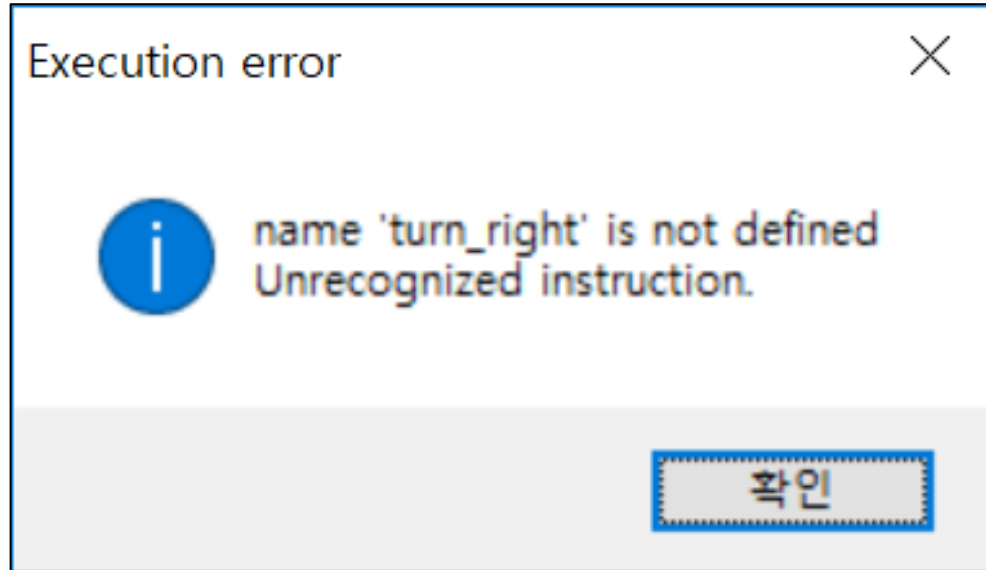
```
#Ex1.  
#define turn_right  
def turn_right():  
    turn_left()  
    turn_left()  
    turn_left()  
#use turn_right()  
turn_left()  
move()  
move()  
turn_right()  
move()  
move()  
turn_right()  
move()  
move()  
turn_left()  
turn_left()  
turn_off()
```

함수정의

함수호출

[Ex1] : 하나만 더!

함수정의와 함수호출의 순서가 바뀌면 어떻게 될까요?



네! 그렇습니다!
turn_right가 정의되지 않았다고 error!!!

꼭!! 함수를 정의한 후 그 아래에서
함수를 호출해 주세요!!!

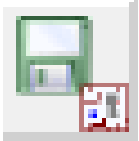
```
#use turn_right()  
turn_left()  
move()  
move()  
turn_right()  
move()  
move()  
turn_right()  
move()  
move()  
turn_left()  
turn_left()  
turn_off()  
#define turn_right
```

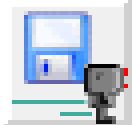
```
def turn_right():  
    ...turn_left()  
    ...turn_left()  
    ...turn_left()
```

함수호출

함수정의

[Ex1] 완성된 프로그램 저장하기

1) 월드 저장하기  Ex1_turn_right.wld

2) 코드 저장하기  Ex1_turn_right.rur



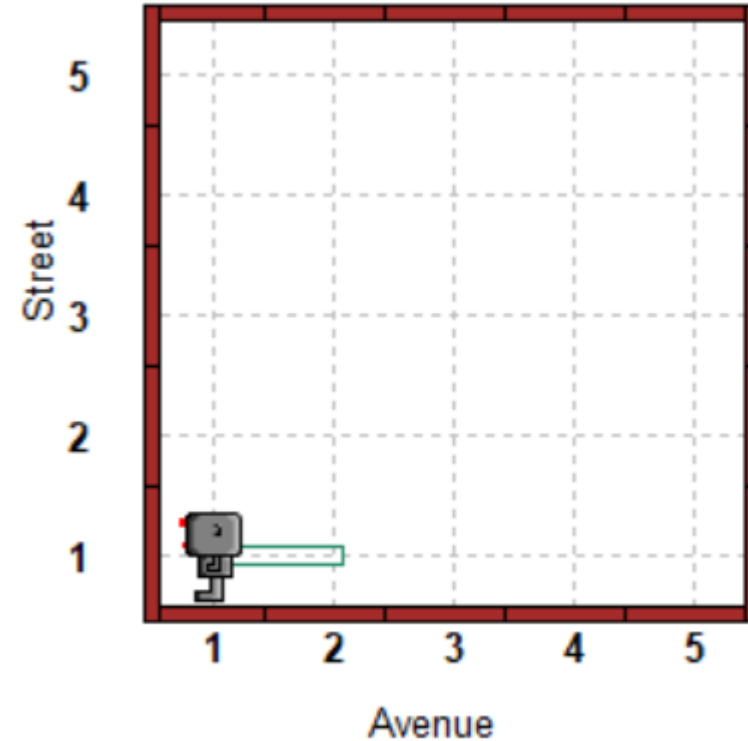
코드 저장 시, 'Code' 영역에 마우스 커서가 있어야 한다.

코드 중 한글이 있으면 안된다(주석 포함).

[Ex2] turn_around() 명령문을 정의하세요!

(A)부분에 명령어를 넣어 로봇이 뒤돌아서는 동작을 할 수 있게 해 주세요 :)

```
1 #Ex2.  
2 #define turn_around()  
3 def turn_around():  
4     ... #... (A) ...  
5 move()  
6 turn_around()  
7 move()  
8 turn_off()
```

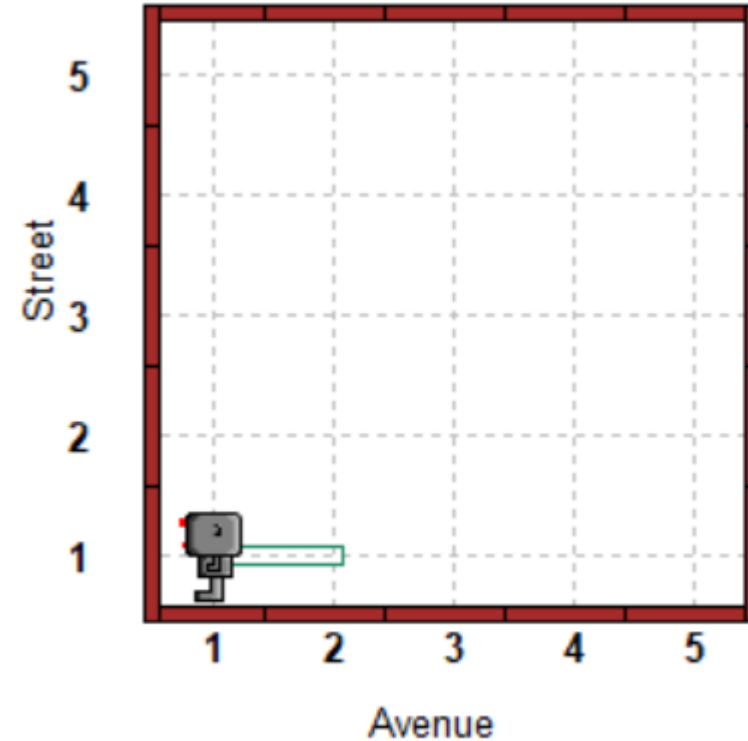


여기에 명령어를 넣어주세요! 1줄이든 여러 줄이든 상관없어요!

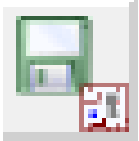
[Ex2] (완성)turn_around() 명령문을 정의하세요!

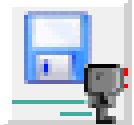
(A)부분에 명령어를 넣어 로봇이 뒤돌아서는 동작을 할 수 있게 해 주세요 :)

```
1  #Ex2.  
2  #define turn_around()  
3  def turn_around():  
4      ....turn_left()  
5      ....turn_left()  
6  move()  
7  turn_around()  
8  move()  
9  turn_off()
```



[Ex2] 완성된 프로그램 저장하기

1) 월드 저장하기  Ex2_turn_around.wld

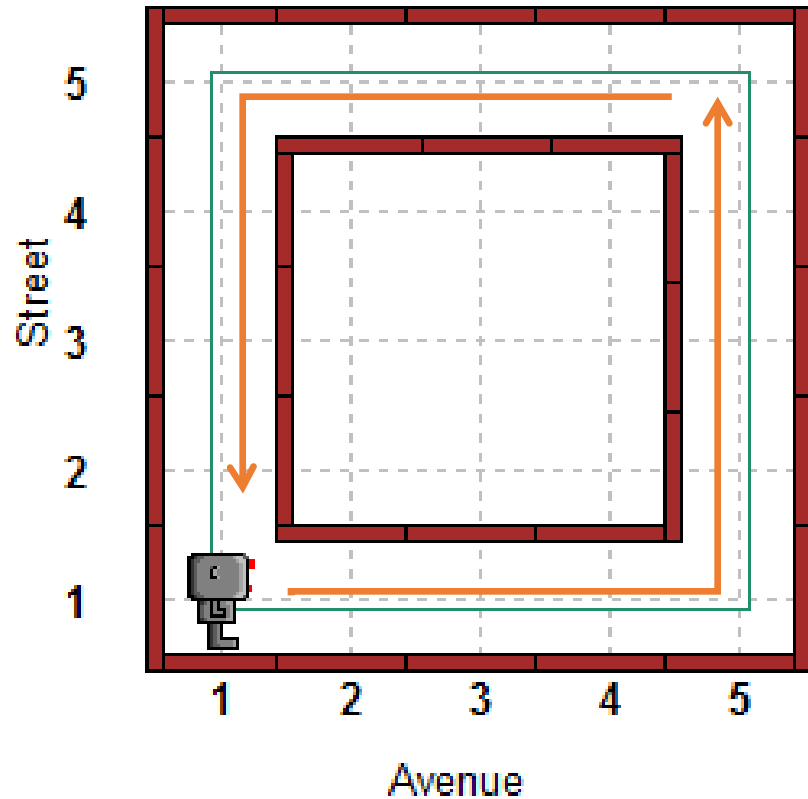
2) 코드 저장하기  Ex2_turn_around.rur



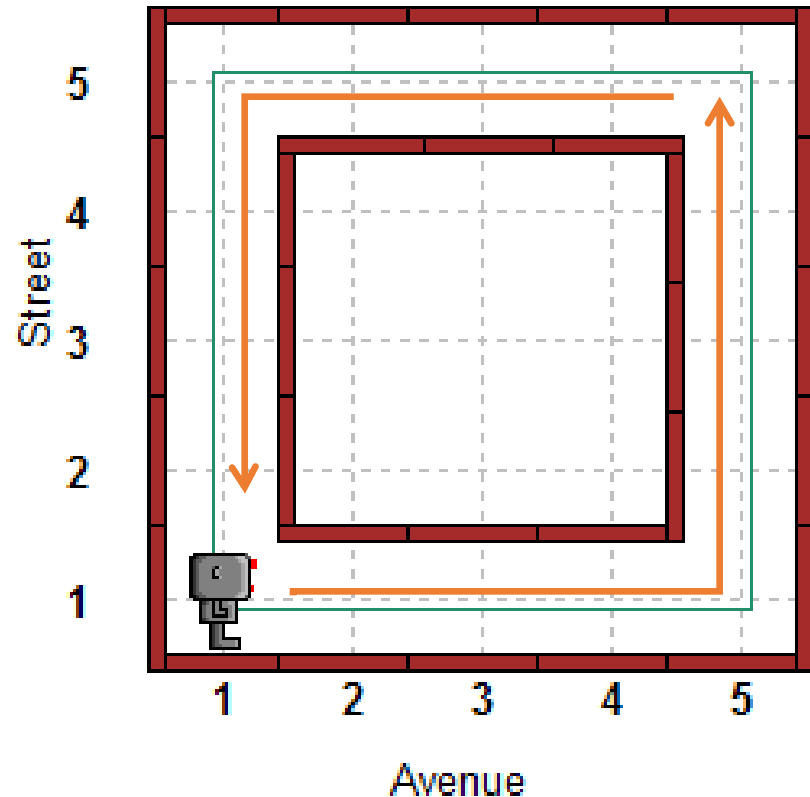
코드 저장 시, 'Code' 영역에 마우스 커서가 있어야 한다.

코드 중 한글이 있으면 안된다(주석 포함).

[Ex3] : 4칸 이동하는 go()함수를 작성하고, 로봇이 반시계방향으로 사각형을 그리며 이동하도록 프로그램을 완성해 보세요.



[Ex3] : 4칸 이동하는 go()함수를 작성하고, 로봇이 반시계방향으로 사각형을 그리며 이동하도록 프로그램을 완성해 보세요.(완성)



```
1 def go() :  
2     ... move()  
3     ... move()  
4     ... move()  
5     ... move()  
6 go()  
7 turn_left()  
8 go()  
9 turn_left()  
10 go()  
11 turn_left()  
12 go()  
13 turn_left()  
14 turn_off()
```

[Ex3] 완성된 프로그램 저장하기

1) 월드 저장하기  Ex3_go.wld

2) 코드 저장하기  Ex3_go.rur



코드 저장 시, 'Code' 영역에 마우스 커서가 있어야 한다.

코드 중 한글이 있으면 안된다(주석 포함).

【우리가 만든 함수를 다시 살펴보아요!】

```
def turn_right() :  
    . . . turn_left()  
    . . . turn_left()  
    . . . turn_left()
```

```
def turn_around() :  
    . . . turn_left()  
    . . . turn_left()
```

```
def go() :  
    . . . move()  
    . . . move()  
    . . . move()  
    . . . move()
```

우리가 만든 함수, 이상한 점 없나요?

【우리가 만든 함수를 다시 살펴보아요!】

```
def turn_right() :  
    turn_left()  
    turn_left()  
    turn_left()
```

```
def turn_around() :  
    turn_left()  
    turn_left()
```

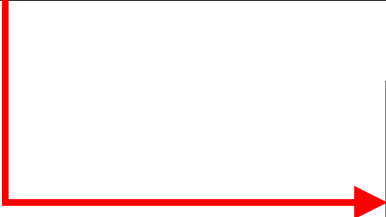
```
def go() :  
    move()  
    move()  
    move()  
    move()
```

네! 그렇습니다!! ·' (° ▽ °) ' · * ∴ ∴ ∴ ∴ ∴ * ∴ ∴ ∴ ∴ ∴ *

【내장함수】 **repeat(A,B)**

A명령어를 B번 반복하게 하는 함수

```
def turn_right() :  
    . . . turn_left()  
    . . . turn_left()  
    . . . turn_left()
```



```
def turn_right() :  
    . . . repeat(turn_left, 3)
```

[내장함수] repeat(A,B)

A명령어를 B번 반복하게 하는 함수

```
def turn_right():  
    ... repeat(turn_left, 3)
```




주의사항!!!! 함수 이름만 써주면 되요! 괄호()는 쓰지 않아요!!!

[내장함수] repeat(A,B)


A명령어를 B번 반복하게 하는 함수

```
def turn_around():  
    turn_left()  
    turn_left()
```



```
def turn_around():  
    repeat(turn_left, 2)
```

```
def go():  
    move()  
    move()  
    move()  
    move()
```

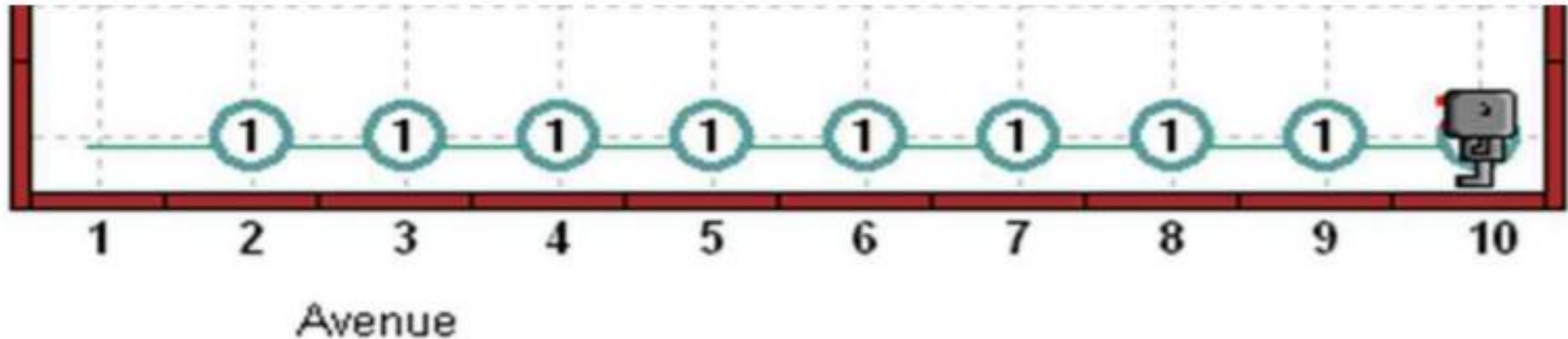


```
def go():  
    repeat(move, 4)
```

[Ex4] : 비퍼를 벽까지 놓았다가 다시 수거해 오세요!

<처리조건>

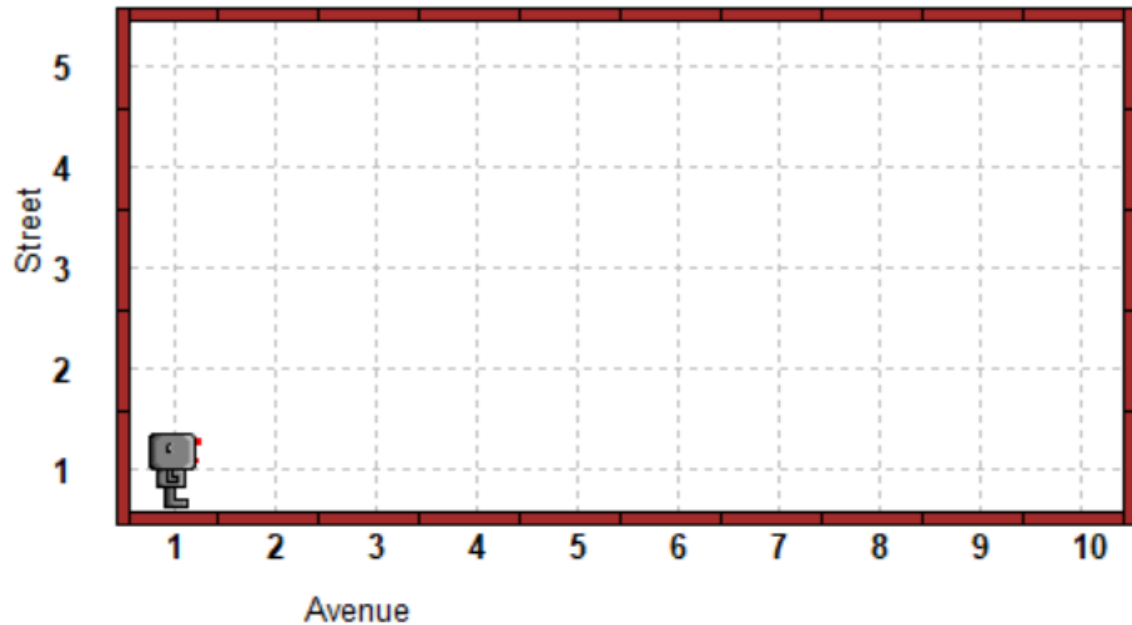
1. def와 repeat()를 사용해서 프로그램을 작성하세요!
2. 프로그램을 작성할 때 자신을 반복하지 마세요!



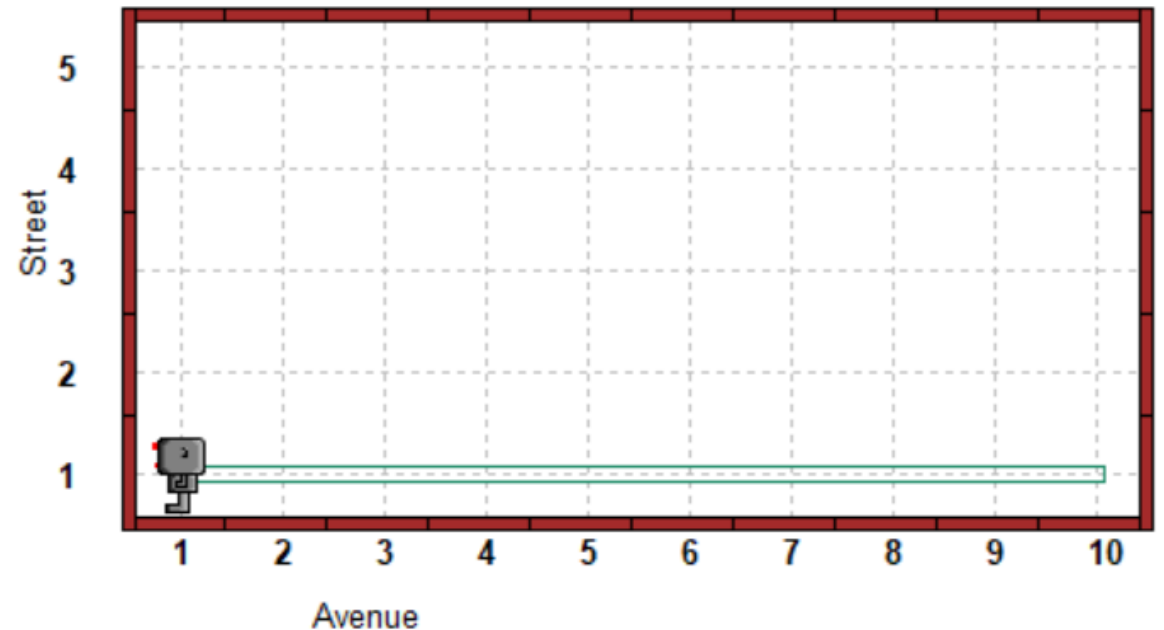
[Ex4] : (Hint) 비퍼를 벽까지 놓았다가 다시 수거해 오세요!

<처리조건>

1. def와 repeat()를 사용해서 프로그램을 작성하세요!
2. 프로그램을 작성할 때 자신을 반복하지 마세요!



<실행 전>



<실행 후>

[Ex4] : (Hint) 비퍼를 벽까지 놓았다가 다시 수거해 오세요!

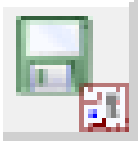
♡ 아래의 주석문 (A)~(C)를 완성시키면 됩니다.
단, (A)~(C)는 1줄이든 여러 줄이든 상관없어요!

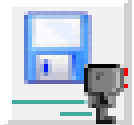
```
def move_and_put() :  
    ... #... (A) ...  
def pick_and_move() :  
    ... #... (B) ...  
def turn_around() :  
    ... #... (C) ...  
repeat(move_and_put, 9)  
turn_around()  
repeat(pick_and_move, 9)  
turn_off()
```


[Ex4] : (완성)비퍼를 벽까지 놓았다가 다시 수거해 오세요!

```
def move_and_put() :  
    ... move()  
    ... put_beeper()  
def pick_and_move() :  
    ... pick_beeper()  
    ... move()  
def turn_around() :  
    ... repeat(turn_left, 2)  
  
repeat(move_and_put, 9)  
turn_around()  
repeat(pick_and_move, 9)  
turn_off()
```

[Ex4] 완성된 프로그램 저장하기

1) 월드 저장하기  Ex4_pick_up.wld

2) 코드 저장하기  Ex4_pick_up.rur



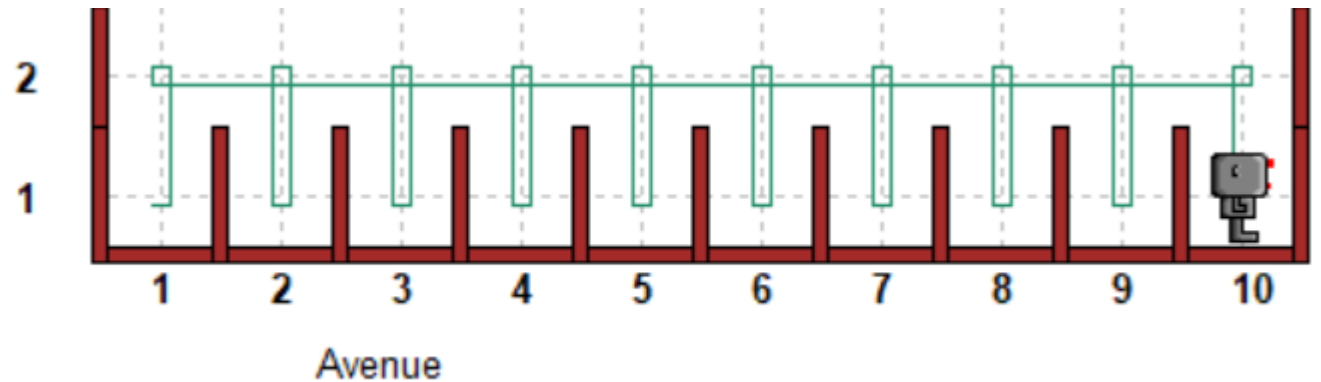
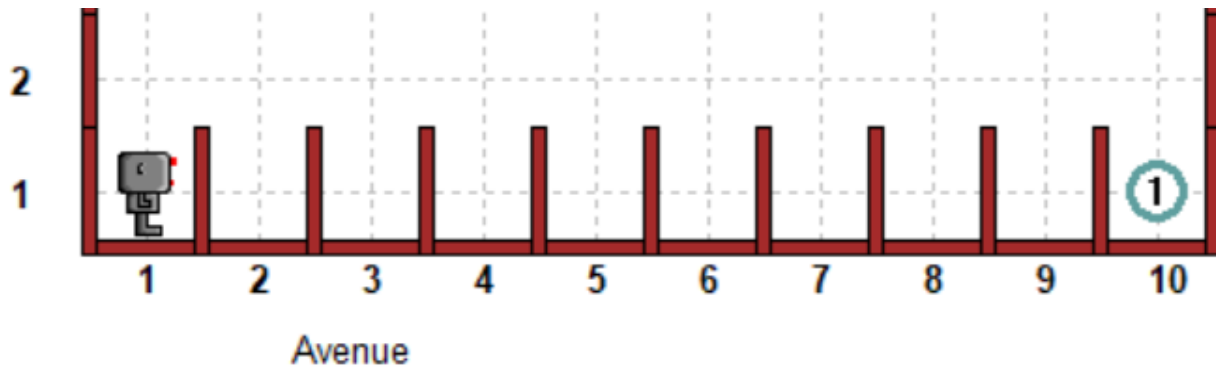
코드 저장 시, 'Code' 영역에 마우스 커서가 있어야 한다.

코드 중 한글이 있으면 안된다(주석 포함).

[Ex5] : 로봇이 허들을 넘어요

리보그가 허들 넘기 경주에 참가합니다. 허들을 넘어 결승점(비퍼가 놓인 곳)에 리보그가 도착하도록 프로그램을 작성하세요!

<처리조건> `jump_hurdle()`를 정의하여 활용하기!



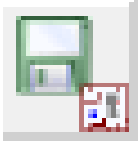
[Ex5] : (Hint)로봇이 허들을 넘어요

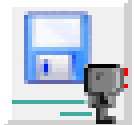
♡아래의 주석문 (A)를 완성시키면 됩니다.
단, (A)는 1줄이든 여러 줄이든 상관없어요!

```
def turn_right():  
    ... repeat(turn_left, 3)  
def jump_hurdle():  
    ... #... (A) ...  
repeat(jump_hurdle, 9)  
pick_beeper()  
turn_off()
```



[Ex5] 완성된 프로그램 저장하기

1) 월드 저장하기  Ex5_hurdle.wld

2) 코드 저장하기  Ex5_hurdle.rur



코드 저장 시, 'Code' 영역에 마우스 커서가 있어야 한다.

코드 중 한글이 있으면 안된다(주석 포함).

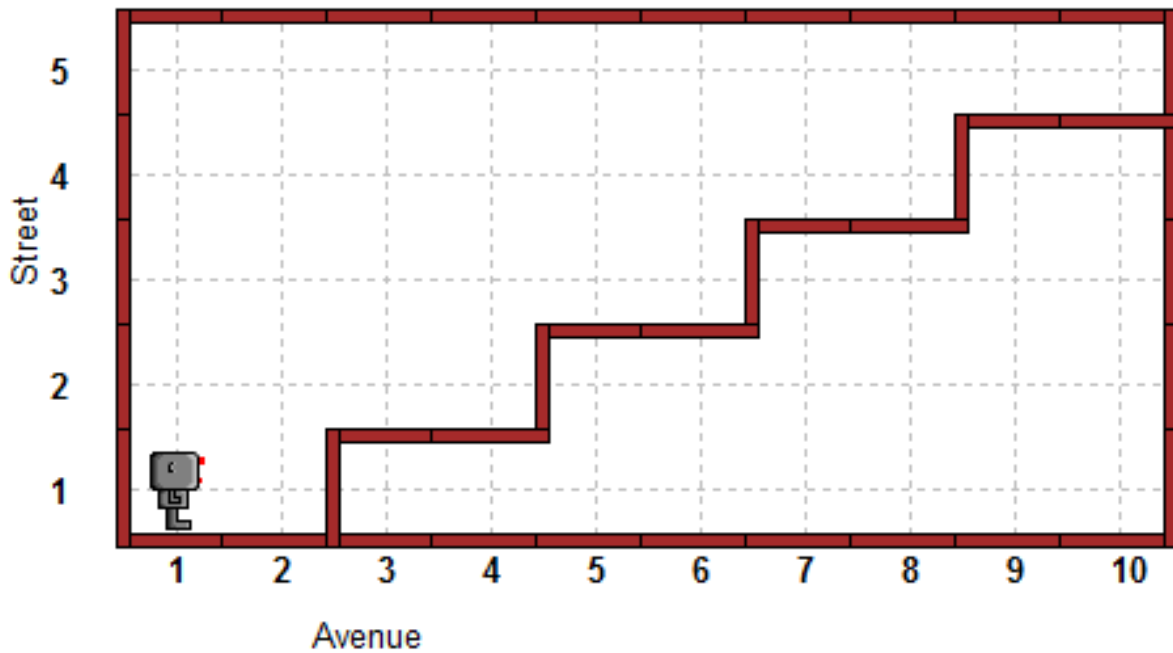
【잠깐! 지난 시간 설문조사】 #가장 인상적이었던 것?



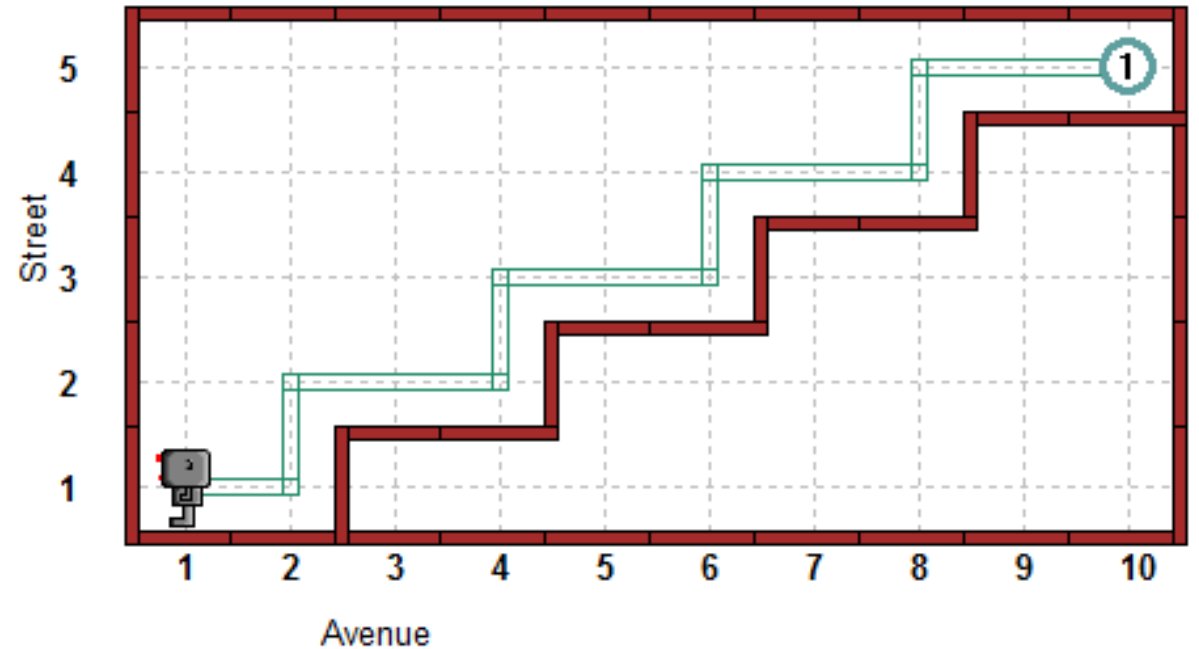
[Ex6] : 다시 돌아온 신문 배달!

<처리조건>

1. def와 repeat()를 사용해서 프로그램을 작성하세요!
2. 프로그램을 작성할 때 자신을 반복하지 마세요!



<실행 전>

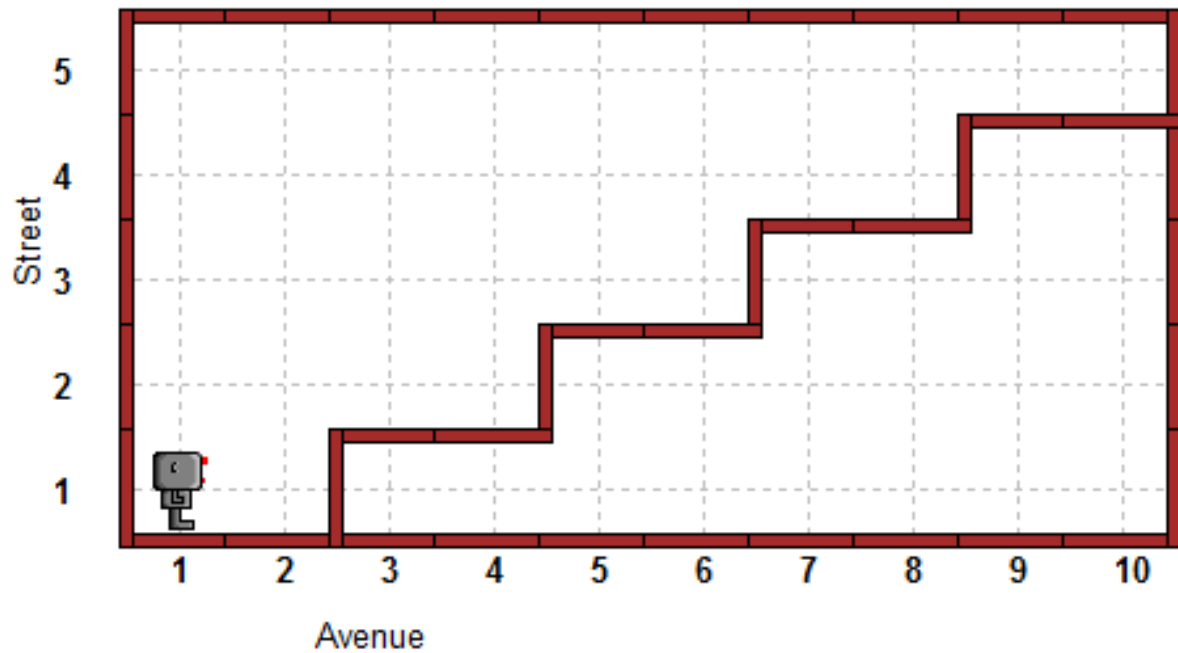


<실행 후>

[Ex6] : (Hint) 다시 돌아온 신문 배달!

<처리조건>

1. def와 repeat()를 사용해서 프로그램을 작성하세요!
2. 프로그램을 작성할 때 자신을 반복하지 마세요!



로봇은 이런 행동을 해요.

1. 4계단 오르기
2. 신문 내려놓기
3. 돌기
4. 4계단 내려오기

[Ex6] : (Hint)다시 돌아온 신문 배달!

<처리조건>

1. def와 repeat()를 사용해서 프로그램을 작성하세요!
2. 프로그램을 작성할 때 자신을 반복하지 마세요!

로봇은 이런 행동을 해요.

1. 4계단 오르기
2. 신문 내려놓기
3. 돌기
4. 4계단 내려오기

파이썬 코드스럽게 바꾸기

- 1.climb_up_four_stairs()
- 2.put_beeper()
- 3.turn_around()
- 4.climb_down_four_stairs()

[Ex6] : (Hint) 다시 돌아온 신문 배달!

<처리조건>

1. def와 repeat()를 사용해서 프로그램을 작성하세요!
2. 프로그램을 작성할 때 자신을 반복하지 마세요!

파이썬 코드스럽게 바꾸기

```
1.climb_up_four_stairs( )  
2.put_beeper( )  
3.turn_around( )  
4.climb_down_four_stairs( )
```

```
1.def climb_up_four_stairs( ) :
```

```
    climb_up_one_stairs( )
```

```
    climb_up_one_stairs( )
```

```
    climb_up_one_stairs( )
```

```
    climb_up_one_stairs( )
```

```
    def climb_up_one_stairs( ) :
```

```
        turn_left( )
```

```
        move( )
```

```
        turn_right( )
```

```
        move( )
```

```
        move( )
```

[Ex6] : (Hint) 다시 돌아온 신문 배달!

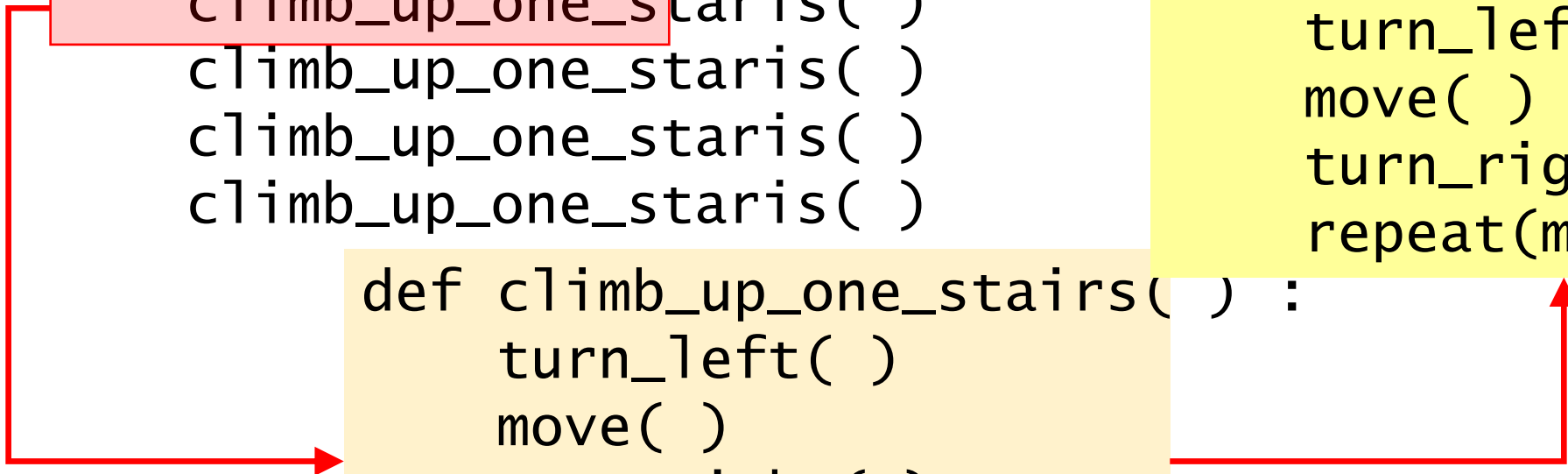
<처리조건>

1. def와 repeat()를 사용해서 프로그램을 작성하세요!
2. 프로그램을 작성할 때 자신을 반복하지 마세요!

```
1. def climb_up_four_stairs( ) :  
    climb_up_one_stairs( )  
    climb_up_one_stairs( )  
    climb_up_one_stairs( )  
    climb_up_one_stairs( )
```

```
def climb_up_one_stairs( ) :  
    turn_left( )  
    move( )  
    turn_right( )  
    move( )  
    move( )
```

```
def climb_up_one_stairs( )  
    turn_left( )  
    move( )  
    turn_right( )  
    repeat(move,2)
```



[Ex6] : (짱짱 Hint) 다시 돌아온 신문 배달!

아래의 코드 블록들을 순서대로 나열하세요! :)

```
def climb_down_one_stairs():  
    repeat(move,2)  
    turn_left()  
    move()  
    turn_right()
```

```
def turn_around():  
    repeat(turn_left,2)
```

```
def climb_up_one_stairs():  
    turn_left()  
    move()  
    turn_right()  
    repeat(move,2)
```

```
def turn_right():  
    repeat(turn_left,3)
```

```
def climb_down_four_stairs():  
    repeat(climb_down_one_stairs,4)
```

```
def climb_up_four_stairs():  
    repeat(climb_up_one_stairs,4)
```

```
move()  
climb_up_four_stairs()  
put_beeper()  
turn_around()  
climb_down_four_stairs()  
move()  
turn_off()
```

[Ex6] : (더 짱짱 Hint) 다시 돌아온 신문 배달!

```
def turn_right():  
    repeat(turn_left,3)
```

```
def climb_up_one_stairs():  
    turn_left()  
    move()  
    turn_right()  
    repeat(move,2)
```

```
def climb_up_four_stairs():  
    repeat(climb_up_one_stairs,4)
```

```
def climb_down_one_stairs():  
    repeat(move,2)  
    turn_left()  
    move()  
    turn_right()
```

```
def climb_down_four_stairs():  
    repeat(climb_down_one_stairs,4)
```

```
def turn_around():  
    repeat(turn_left,
```

```
move()  
climb_up_four_stairs()  
put_beeper()  
turn_around()  
climb_down_four_stairs  
(  
move()  
turn_off()
```

[Ex6] : (더 짱짱 Hint) 다시 돌아온 신문 배달!

```
def turn_right():  
    repeat(turn_left,3)
```

```
def climb_up_one_stairs():  
    turn_left()  
    move()  
    turn_right()  
    repeat(move,2)
```

```
def climb_up_four_stairs():  
    repeat(climb_up_one_stairs,4)
```

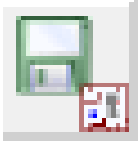
```
def climb_down_one_stairs():  
    repeat(move,2)  
    turn_left()  
    move()  
    turn_right()
```

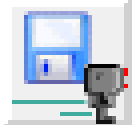
```
def climb_down_four_stairs():  
    repeat(climb_down_one_stairs,4)
```

```
def turn_around():  
    repeat(turn_left,
```

```
move()  
climb_up_four_stairs()  
put_beeper()  
turn_around()  
climb_down_four_stairs  
(  
move()  
turn_off()
```

[Ex6] 완성된 프로그램 저장하기

1) 월드 저장하기  Ex6_newspaper.wld

2) 코드 저장하기  Ex6_newspaper.rur



코드 저장 시, 'Code' 영역에 마우스 커서가 있어야 한다.

코드 중 한글이 있으면 안된다(주석 포함).

【수업 정리】

1. 바탕화면 오른쪽 위 **과제제출**
2. 이번 시간에 실습한 **모든 파일을 ZIP파일로 압축**하여 제출(총 **12개**)

학번_이름.zip

Ex1~Ex6까지 wld파일, rur파일

3. 수업 피드백 작성 :

<https://forms.gle/PZTUZ4cEUvkysorA8>

【다음 시간에는】

RUR-PLE 프로젝트

[RUR-PLE프로젝트 안내]

#. <가상 시나리오 만들기>

비퍼, 벽을 모두 활용할 수 있는 가상 상황을 만들어요!

리보그가 허들 넘기 경주에 참가합니다. 허들을 넘어 결승점(비퍼가 놓인 곳)에 리보그가 도착하도록 프로그램을 작성하세요!

1. 비퍼 : 결승점
2. 벽 : 허들

로봇이 신문배달을 합니다. 집 앞 계단에 올라가서 신문을 마지막 계단에 놓고, 다시 처음 시작 지점으로 돌아오는 프로그램을 완성해 보세요!

1. 비퍼 : 신문
2. 벽 : 계단