

컴퓨터 프로그래밍 II : Python

4주차 : 조건문

조현준(Chris)

cho1475@korea.ac.kr



3주차 복습 : 함수란?

기능(function) 혹은 행동(Action) → 코드의 묶음 → 로직의 흐름

함수 없이, 중복된 코드를 실행하는 경우

```
1 a = 1
2
3 a = 2
4
5 a = 3
6
7 a = 2
8 b = 3
9 c = 4
10
11 result = ((-b + (b**2 - (4*a*c))**0.5) / (2*a))
12 print(result)
13
14 executed in 3ms, finished 00:42:45 2021-10-02
15 (-0.749999999999999+1.1989578808281798j)
```

함수를 활용하여 중복 코드 제거하고 실행하는 경우

```
: 1 #function definition(or declaration)
2 def calculte_complex_formula(a,b,c):
3     return (-b + (b**2 - (4*a*c))**0.5) / (2*a)
4
5
6 #function call(or excution)
7 calculte_complex_formula(1, -2, 1)
8 calculte_complex_formula(1, 2, 3)
9 calculte_complex_formula(3, 4, 5)
10 result = calculte_complex_formula(a = 2, b = 3, c = 4)
11 print(result)
12
13 executed in 5ms, finished 00:51:11 2021-10-02
14 (-0.749999999999999+1.1989578808281798j)
```

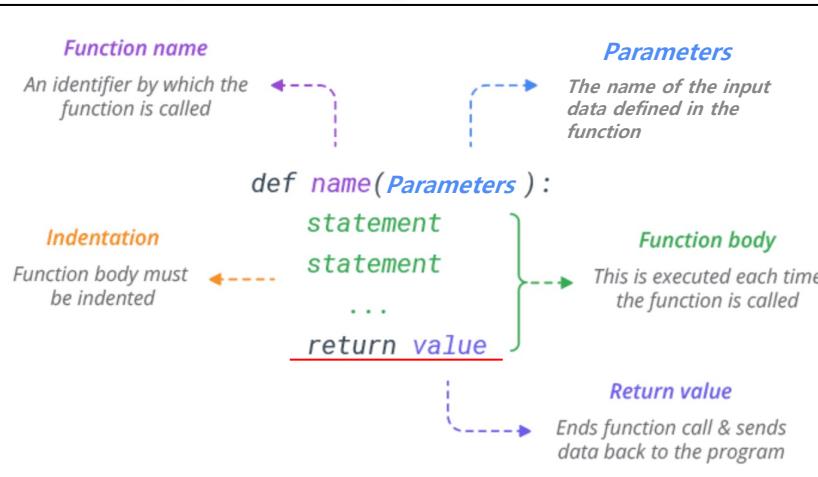
코드의 묶음(재활용) == 추상화

→ 가독성 향상 및 유지보수 용이

3주차 복습 : 함수란?

함수(function) 기본 : 선 정의(Definition), 후 실행(Call)

선 정의(Definition)



선 정의(Definition)

```
1 or declaration
2 def calculte_complex_formula(a,b,c):
3     return (-b + (b**2 - (4*a*c))**0.5) / (2*a)
4
5 후 실행(Call)
6
7 calculte_complex_formula(1, -2, 1)
8 calculte_complex_formula(1, 2, 3)
9 calculte_complex_formula(3, 4, 5)
10 result = calculte_complex_formula(a = 2, b = 3, c = 4)
11 print(result)

executed in 5ms, finished 00:51:11 2021-10-02
(-0.749999999999999+1.1989578808281798j)
```

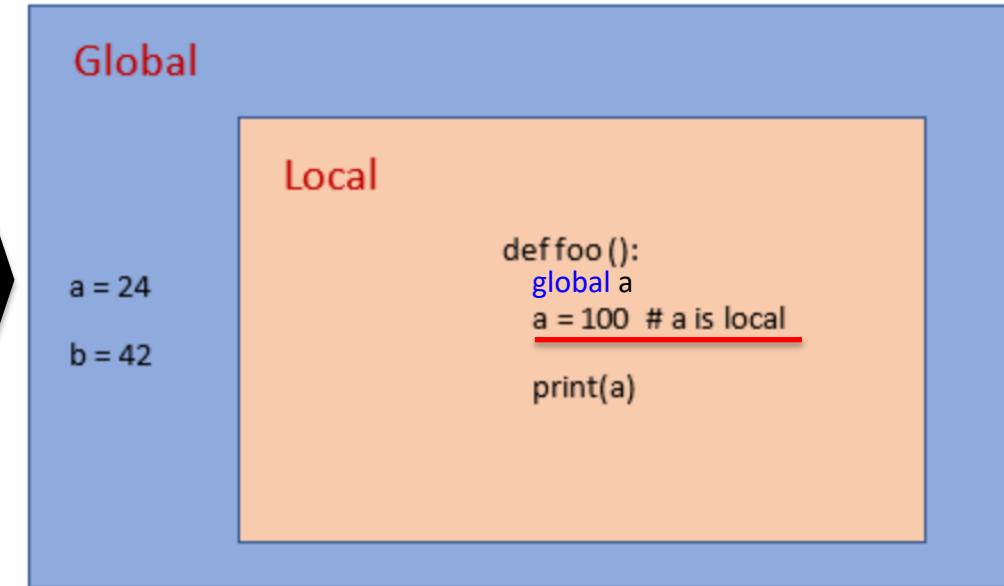
후 실행(Call)시, 결과 값을 받기 위해서는 반환(Return)이 필요하며,

def 를 사용하지 않음

3주차 복습 : 함수란?

함수(function) 심화 : variable scope

```
1  a = 24 # global
2  b = 42 # global
3  print(a)
4
5  def foo(): # function foo is global
6      a = 100 # local
7      print(a)
8
9  foo()
10
11 -----
12
13 24
14 100
```



1. 함수 내에서, 일반적인 우선 순위: local > global

2. 함수 내에서, global keyword 적용시, 우선 순위: local < global

4주차: 조건문 (Conditional Statement)

1. 개념 및 기초

2. 심화

3. 실습 예제

조건문 개념



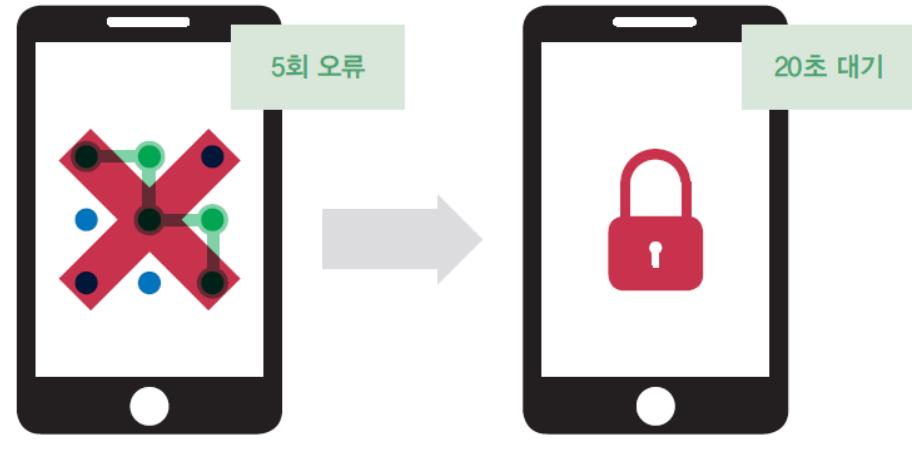
조건문 개념

조건문의 개념

- 조건문(conditional statement): 조건에 따라 특정 동작을 하도록 하는 프로그래밍 명령어이다.
- 조건 여부에 따라 실행 구문이 달라지는 **분기적(branch)** 실행이 가능하다.
- 파이썬에서는 조건문을 사용하기 위해 **if, else, elif** 등의 명령 키워드를 사용한다.



[조건문의 예시1: 미세 먼지 농도에 따른 마스크 착용 여부]



조건의 기준

실행할 명령

[조건문의 예시2: 스마트폰 잠금 해제 패턴]

조건문 개념: 기본 if 문

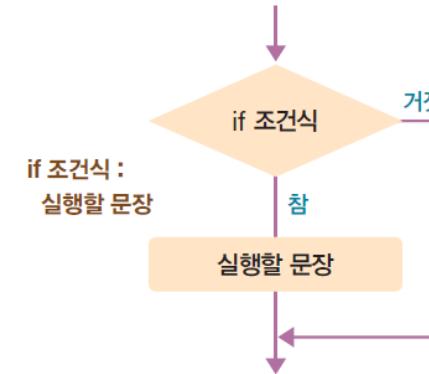
일반 순차적 프로그램 구성 vs 일반적인 조건문이 포함된 프로그램 구성



```
1 x = 1
2 y = 2
3 print(f"x={x}, y={y}")
4 print(f"[실행문1]x+y = {x+y}")
5 print(f"[실행문2]x**2 = {x**2}")
6 print(f"[실행문3]x+y**2 = {x+y**2}")
```

executed in 5ms, finished 21:50:18 2021-08-05

```
x=1, y=2
[실행문1]x+y = 3
[실행문2]x**2 = 1
[실행문3]x+y**2 = 5
```



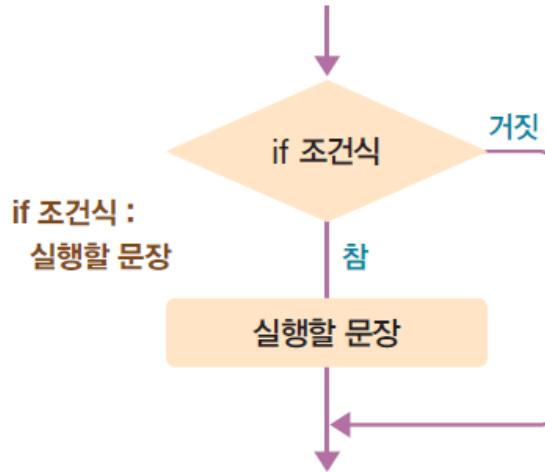
```
1 x = 1
2 if x >= 1:
3     print(f"실행할문장, x={x}")
4
```

executed in 3ms, finished 21:52:35 2021-08-05

실행할문장, x=1

조건문 개념: 기본 if 문

일반적인 조건문이 포함된 프로그램 구성 vs 분기별 실행에 대한 조건문이 포함된 프로그램 구성



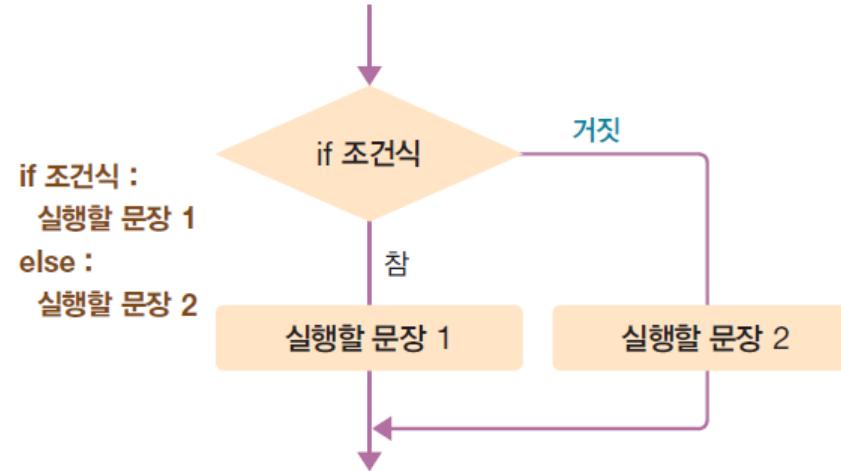
```
if 조건식 :  
    실행할 문장
```

실행할 문장

```
1 x = 1  
2 if x >= 1:  
3     print(f"실행할문장, x={x}")  
4
```

executed in 3ms, finished 21:52:35 2021-08-05

실행할문장, x=1



```
if 조건식 :  
    실행할 문장 1  
else :  
    실행할 문장 2
```

실행할 문장 1

실행할 문장 2

```
1 x = -1  
2 if x >= 1:  
3     print(f"실행할문장1, x={x}")  
4 else:  
5     print(f"실행할문장2, x={x}")
```

executed in 3ms, finished 21:54:35 2021-08-05

실행할문장2, x=-1

조건문 기초: if-else

if -else문

- if-else문의 기본 문법은 다음과 같다.

```
if <조건>:          # if를 쓰고 조건 삽입 후 ':' 입력  
    <수행 명령 1-1>  # 들여쓰기 후, 수행 명령 입력  
    <수행 명령 1-2>  # 같은 조건에서 실행일 경우 들여쓰기 유지  
else:                # 조건이 불일치할 경우 수행할 명령  
    <수행 명령 2-1>  # 조건 불일치 시 수행할 명령 입력  
    <수행 명령 2-2>  # 조건 불일치 시 수행할 명령 들여쓰기 유지
```

- ① if 뒤에는 참과 거짓을 판단할 수 있는 조건문이 들어가야 하고, 조건문이 끝나면 반드시 콜론(:)을 붙여야 한다.
- ② 들여쓰기를 사용하여 해당 조건이 참일 경우 수행할 명령을 작성한다.
- ③ if의 조건이 거짓일 경우 else문이 수행된다. else문 생략도 상관없다. 만약 조건에 해당하지 않는 경우에 따로 처리해야 한다면 else문을 넣으면 된다.

조건문 기초: if-else

if -else문

코드 4-1 if-else.py

```
1 print("Tell me your age?")
2 myage = int(input())          # 나이를 입력받아 myage 변수에 할당
3 if myage < 30:                # myage가 30 미만일 때
4     print("Welcome to the Club.")
5 else:                         # myage가 30 이상일 때
6     print("Oh! No. You are not accepted.")
```

Tell me your age?	← 입력 대기
20	← 사용자 입력
Welcome to the Club.	← 출력

조건문 기초: if-else

if -else문 : [코드 4-1] 해석

- 1~2행 : 먼저 나이를 입력하는 메시지가 나타나면 사용자가 나이를 입력한다.
- 3~6행 : 만약, 나이가 30세 미만일 경우 클럽에 입장할 수 있다는 메시지를 출력하고, 30세 이상일 경우에는 클럽에 입장할 수 없다는 메시지를 화면에 출력한다. 여기서 핵심은 myage < 30이라는 조건이다. 사용자가 입력한 값은 myage 변수에 정수형으로 바꿔 할당된다. 다음으로 그 값이 30보다 작다는 조건이 True일 때, 즉 myage < 30이 참일 경우 print("Welcome to the Club")이라는 구문이 동작하고, 해당 조건이 False일 때는 else로 묶인 구문이 동작한다.

조건문 기초: 비교 연산

조건의 판단 : 비교 연산자

- 비교 연산자(또는 조건 연산자): 어떤 것이 큰지 작은지 같은지를 비교하는 것으로, 그 결과는 참(True)이나 거짓(False)이 된다.

비교 연산자	비교 상태	설명
$x < y$	~보다 작음	x 가 y 보다 작은지 검사
$x > y$	~보다 큼	x 가 y 보다 큰지 검사
$x == y$	같음	x 와 y 의 값이 같은지 검사
$x \text{ is } y$	같음(메모리 주소)	x 와 y 의 메모리 주소가 같은지 검사
$x != y$	같지 않음	x 와 y 의 값이 같지 않은지 검사
$x \text{ is not } y$	같지 않음(메모리 주소)	x 와 y 의 메모리 주소가 같지 않은지 검사
$x >= y$	크거나 같음	x 가 y 보다 크거나 같은지 검사
$x <= y$	작거나 같음	x 가 y 보다 작거나 같은지 검사

[비교 연산자]

조건문 기초: 비교 연산

조건의 판단 : 비교 연산자

- 같음을 의미하는 = 연산자와 비교하여, 할당의 의미로 같음을 표현할 때는 ==과 같은 새로운 연산자를 사용한다.
- 다음 코드에서 두 값이 모두 같으니 결과는 True이다. 사실 조건문 코드를 볼 때, 언제나 이러한 코드가 True 또는 False로 치환된다고 생각하면 이해하기 쉽다.

```
>>> 7 == 7
```

조건문 기초: 비교 연산

조건의 판단 : 비교 연산자

- **Is** 연산자는 ==처럼 두 변수가 같음을 비교하지만, ==과 다르게 메모리의 주소를 비교한다.
- 파이썬은 처음 인터프리터를 시작할 때 -5~256까지 변하지 않는 메모리(정적 메모리) 주소에 값 **을 할당**한다. 그리고 해당 값을 다른 변수가 사용할 때, 그 메모리 주소를 반환한다.
- 다음 코드에서 a와 b, 둘 다 100일 때는 is와 ==이 모두 같다고 나오지만, 둘 다 300일 경우에는 값만 같고 메모리 주소는 다르다고 나온다. is not도 마찬가지로 사용된다.

```
>>> a = 100
>>> b = 100
>>> a is b
True
>>> a == b
True
>>> a = 300
>>> b = 300
>>> a == b
True
>>> a is b
False
```

조건문 기초: 비교 연산

조건의 판단 : True와 False의 치환

- 컴퓨터는 기본적으로 이진수만 처리할 수 있으며, **True**는 1로, **False**는 0으로 처리한다.
- 아래 코드를 실행하면 True가 출력된다. 그 이유는 앞서 설명한 것처럼 컴퓨터는 존재하면 True, 존재하지 않으면 False로 처리하기 때문이다.

```
>>> if 1: print("True")
... else: print("False")
```

- 아래 코드를 실행하면 True가 출력된다. 먼저 3 > 5는 False이고 False는 결국 0으로 치환된다. 그래서 이것을 다시 치환하면 (0) < 10이 되고, 이 값은 참이므로 True가 반환된다.

```
>>> (3 > 5) < 10
```

조건문 기초: 논리 연산

조건의 판단 : 논리 연산자

- 논리 연산자는 **and** · **or** · **not**문을 사용해 조건을 확장할 수 있다.

연산자	설명	예시
and	두 값이 모두 참일 경우 True, 그렇지 않을 경우 False	$(7 > 5)$ and $(10 > 5)$ 는 True $(7 > 5)$ and $(10 < 5)$ 는 False
or	두 값 중 하나만 참일 경우 True, 두 값 모두 거짓일 경우 False	$(7 < 5)$ or $(10 > 5)$ 는 True $(7 < 5)$ or $(10 < 5)$ 는 False
not	값을 역으로 반환하여 판단	not $(7 < 5)$ 는 True not $(7 > 5)$ 는 False

[논리 연산자]

조건문 기초: 논리 연산

조건의 판단 : 논리 연산자

- and는 둘 다 참이어야 True, or는 둘 중 하나만 참이어도 True, not은 참이면 False이고 거짓이면 True를 출력한다.

```
>>> a = 8  
>>> b = 5  
>>> a == 8 and b == 4  
False  
>>> a > 7 or b > 7  
True  
>>> not (a > 7)  
False
```



조건문 심화

조건문 심화: 중첩 if

if-elif-else문

- 중첩 if문을 간단히 표현하려면 **if-elif-else문**을 사용한다.
- 다음 같은 점수판이 있다고 가정하자.

점수(score)	학점(grade)
98	
37	
16	
86	
71	
63	

[점수판]

조건문 심화: 중첩 if

if-elif-else문

- 점수에 맞는 학점을 주기 위해 [코드 4-2]와 같이 코드를 입력하면, 어떤 학점으로 계산될까?

코드 4-2 grade.py

```
1 score = int(input("Enter your score: "))

2

3 if score >= 90:
4     grade = 'A'
5 if score >= 80:
6     grade = 'B'
7 if score >= 70:
8     grade = 'C'
9 if score >= 60:
10    grade = 'D'
11 if score < 60:
12    grade = 'F'
13

14 print(grade)
```



조건문 심화: 중첩 if

if-elif-else문

- 실제 [코드 4-2]를 실행하면 모든 값이 'D'나 'F'로 나온다.
- 이유는 바로 다음 그림과 같이 코드가 한 줄씩 차례대로 실행되기 때문이다.

```
score = 98
  if score >= 90:  True
    grade = 'A'  grade = 'A'
  if score >= 80:  True
    grade = 'B'  grade = 'A' → 'B'
  if score >= 70:  True
    grade = 'C'  grade = 'B' → 'C'
  if score >= 60:  True
    grade = 'D'  grade = 'C' → 'D'
  if score < 60:   False
    grade = 'F'  grade = 'D' → 'D'
  print(grade)
```

[if문만을 이용한 학점 계산기에 98을 할당했을 때 결과 산출 방식]

조건문 심화: 중첩 if

if-elif-else문

- [코드 4-2]의 문제를 해결하기 위해서는 여러 개의 조건을 하나의 if문에서 검토할 수 있도록 elif를 사용한 **if-elif-else**문으로 작성해야 한다. elif는 else if의 줄임말로, if문과 같은 방법으로 조건문을 표현할 수 있다.

코드 4-3 if-elif-else.py

```
1 score = int(input("Enter your score: "))

2

3 if score >= 90: grade = 'A'          # 90 이상일 경우 A
4 elif score >= 80: grade = 'B'        # 80 이상일 경우 B
5 elif score >= 70: grade = 'C'        # 70 이상일 경우 C
6 elif score >= 60: grade = 'D'        # 60 이상일 경우 D
7 else: grade = 'F'                  # 모든 조건에 만족하지 못할 경우 F
8

9 print(grade)
```

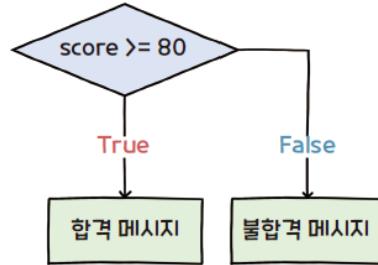


조건문 심화: 중첩 if

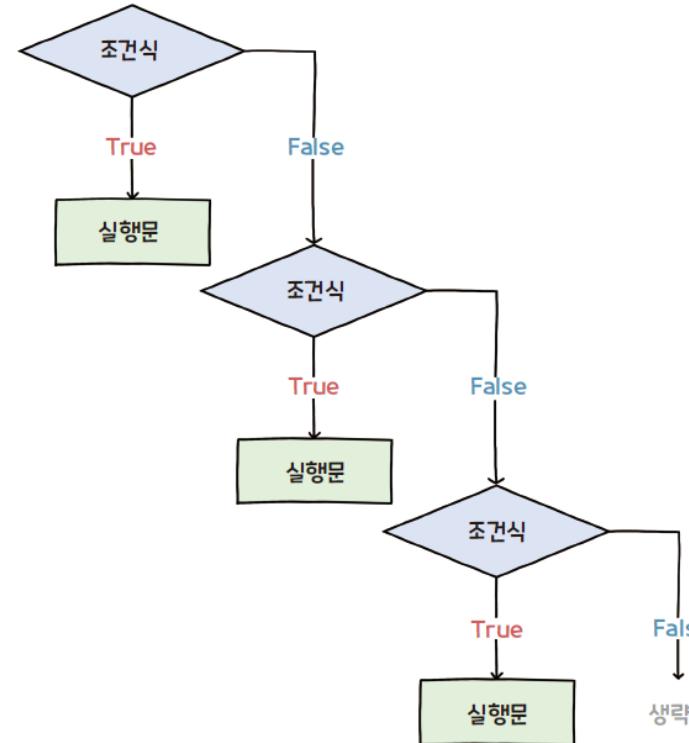
If-else

VS

if-elif



(a) if~else문



if-elif에서
마지막을 생략하지
않고 실행문을
수행하기 위해서는?



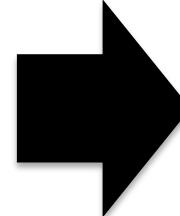
조건문 심화: 삼항 연산자

1줄 미학: variable \leftarrow value1 if conditional statement else value2

variable \leftarrow True이면 value1 할당, False이면 value2 할당

삼항 연산 사용전

```
1 a = 3
2 if a > 1:
3     z = 1
4 else:
5     z = 2
6
7 print(z)
```



삼항 연산 사용 후

```
1 a = 3
2 z = 1 if a > 1 else 2
3 z
```

1

a가 1보다 크면(True) 1로 할당,

a가 1같거나 작으면(False) 2로 할당

1

```

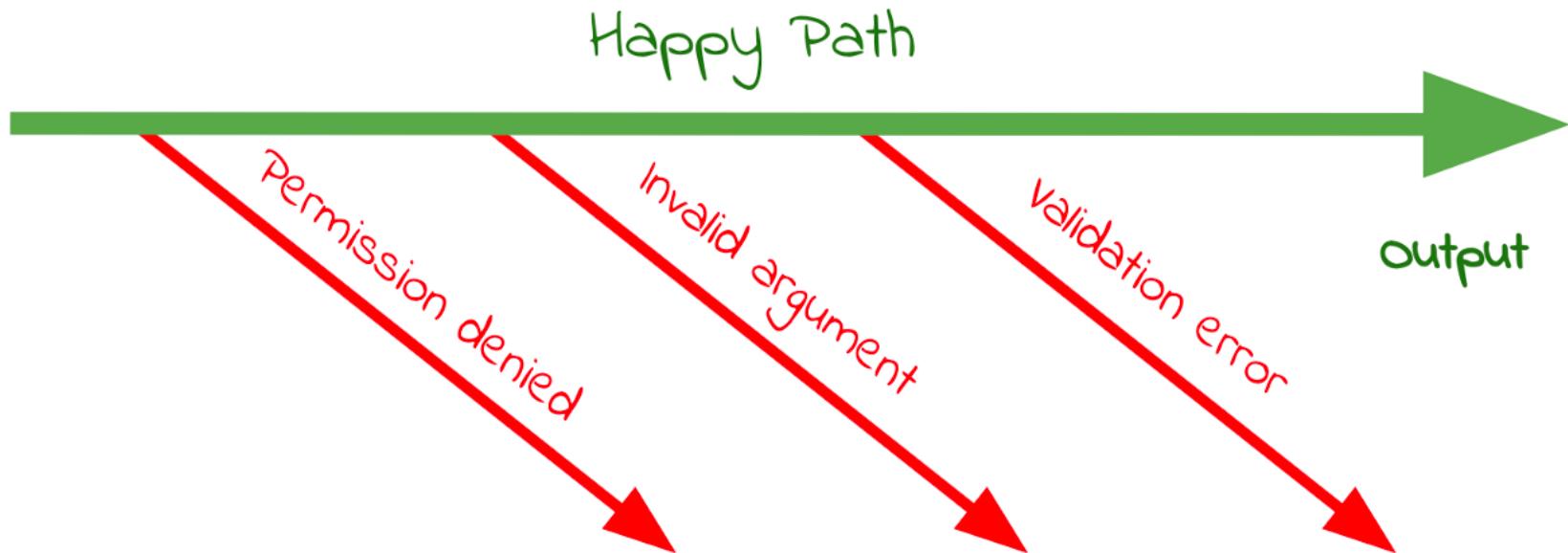
function register()
{
    if (empty($_POST)) {
        $msg = '';
        if (!$_POST['user_name']) {
            if (!$_POST['user_password_new']) {
                if (!$_POST['user_password_repeat']) {
                    if (strlen($_POST['user_password_new']) > 5) {
                        if (strlen($_POST['user_name']) < 6 || strlen($_POST['user_name']) > 30) {
                            if (preg_match('/^(a-zA-Z)([A-Za-z0-9]{5,29})$/i', $_POST['user_name'])) {
                                $user = read_user($_POST['user_name']);
                                if (!isset($user['user_name'])) {
                                    if (!$_POST['user_email']) {
                                        if (strlen($_POST['user_email']) < 65) {
                                            if (filter_var($_POST['user_email'], FILTER_VALIDATE_EMAIL)) {
                                                create_user();
                                                $_SESSION['msg'] = 'You are now registered so please login';
                                                header('Location: ' . $_SERVER['PHP_SELF']);
                                                exit();
                                            } else $msg = 'You must provide a valid email address';
                                        } else $msg = 'Email must be less than 64 characters';
                                    } else $msg = 'Email cannot be empty';
                                } else $msg = 'Username already exists';
                            } else $msg = 'Username must be only a-z, A-Z, 0-9';
                        } else $msg = 'Username must be between 2 and 64 characters';
                    } else $msg = 'Password must be at least 6 characters';
                } else $msg = 'Passwords do not match';
            } else $msg = 'Empty Password';
        } else $msg = 'Empty Username';
        $_SESSION['msg'] = $msg;
    }
    return register_form();
}

```



조건문 심화: early return

종료 조건에 의한 반환(**return**)을 일찍(**early**) 처리하여, 코드 복잡도를 낮추는 디자인 패턴 → 입력 값(None) 체크 등의 방어코드 컨셉에 주로 활용



실습 예제

<http://qj.rightline.kr/contest>



Q&A



감사합니다.

본 강의 교안은 한빛 아카데미(주)에서 제공된
강의 교안을 참고하여 제작되었습니다.

