# python #6

## ▼ 문제

- measure_time이라는 이름의 데코레이터 함수를 작성
- 이 데코레이터는 어떤 함수든 wrapping후 실행 시간을 측정한 뒤,
  - 함수 실행 결과는 그대로 반환하고
  - 실행 시간은 "함수명 took 0.1234 seconds"와 같이 출력
- 임의의 연산 지연이 있는 함수 slow_function()에 적용하여 정상 동작을 확인

## ▼ 코드

### ▼ 전체 코드

```
import time

# TODO: 데코레이터 함수 구현
def measure_time(func):
    def wrapper(*args, **kwargs):
        start = time.perf_counter() # 시작 시간 기록
        result = func(*args, **kwargs) # 원래 함수 호출, 실행 결과를 result에 저장
        end = time.perf_counter() # 종료 시간 기록
        elapsed = end - start # 걸린 시간 계산
        print(f"{func.__name__} took {elapsed:.4f} seconds")  # 함수 이름과 걸린 시간을 출력
        return result # 원래 함수의 반환값을 그대로 반환
    return wrapper

# 테스트용 함수
@measure_time
def slow_function(): # slow_function 위에 @measure_time을 붙이면 slow_function 호출 시 실제로는 wrapp
er가 호출
    time.sleep(1.5)  # time.sleep(1.5)로 1.5초 지연을 준 뒤
    return "완료!" # "완료!" 문자열을 반환

# 실행
result = slow_function() # 실행 시: 실행 시간 로그 출력
print("함수 결과:", result)
```

### ▼ measure_time 데코레이터

```
import time

# TODO: 데코레이터 함수 구현
def measure_time(func):
    def wrapper(*args, **kwargs):
        start = time.perf_counter() # 시작 시간 기록
        result = func(*args, **kwargs) # 원래 함수 호출, 실행 결과를 result에 저장
        end = time.perf_counter() # 종료 시간 기록
        elapsed = end - start # 걸린 시간 계산
        print(f"{func.__name__} took {elapsed:.4f} seconds")  # 함수 이름과 걸린 시간을 출력
        return result # 원래 함수의 반환값을 그대로 반환
    return wrapper
```

- wrapper(*args, **kwargs)

    - 원래 함수를 호출하기 전후로 실행 시간을 측정하고 출력하는 함수
    - 가변인자(*args, **kwargs)를 받아서 모든 매개변수형태의 함수에서 인자를 전달하도록 설정
- time.perf_counter()
    - 시작 시간 기록
- result = func(*args, **kwargs)
    - 원래 함수 호출, 실행 결과를 result에 저장
- time.perf_counter()
    - 종료 시간 기록
- print(f"{func.__name__} took {elapsed:.4f} seconds")
    - func.__name__: 함수 이름 문자열.
    - 함수 이름과 걸린 시간을 출력.
- return result
    - 원래 함수의 반환값을 그대로 반환 (데코레이터를 적용해도 함수의 리턴 값은 변경되지 않는다)

▼ 테스트용 함수, 실행

```
# 테스트용 함수
@measure_time
def slow_function(): # slow_function 위에 @measure_time을 붙이면 slow_function 호출 시 실제로는 wrapp
er가 호출
    time.sleep(1.5)  # time.sleep(1.5)로 1.5초 지연을 준 뒤
    return "완료!" # "완료!" 문자열을 반환

# 실행
result = slow_function() # 실행 시: 실행 시간 로그 출력
print("함수 결과:", result)
```

- @measure_time → def slow_function()
    - slow_function 위에 @measure_time을 붙이면 slow_function 호출 시 실제로는 wrapper가 호출
    - wrapper 안에서 실행 시간 측정 → 시간 출력 → 원래 반환값 반환.
    - slow_function()
        - time.sleep(1.5)로 1.5초 지연을 준 뒤 "완료!"라는 문자열을 반환
- result = slow_function()
    - 실행 시:  실행 시간 로그 출력

▼ **결과**

```
slow_function took 1.5004 seconds
함수 결과: 완료!
```