[파이썬 트랙] 1회차 월말평가 - Python

시험 과목

✔ Python

시험 목적

✔ Python 프로그래밍 기본 문법에 대한 이해

시험 유의 사항

1) 성실하게 테스트에 임할 것 **(부정 행위시 성적 무효 및 중도 퇴소 처리)**
2) **생성형 AI 사용 금지**
3) **VSCode 코파일럿 로그아웃 필수(코파일럿 로그인이 확인될 시 부정행위 처리)**
4) 각 문제별로 스켈레톤 코드가 작성된 파일이 제공되며, 해당 코드 파일을 수정하여 정답 코드를 작성할 것 **(단, 주어진 함수명은 수정불가)**
5) 소스코드 유사도 판단 프로그램 기준 **부정행위**로 판단될 시, **0점 처리** 및 **학사 기준**에 의거 조치 실시 예정
6) 테스트 케이스와는 별도로 채점 케이스가 존재함
7) 내장 함수 활용 여부에 따라 **배점 방식**이 상이하므로 문제별 지시 사항을 **필독**할 것
8) 특정 내장 함수 활용 시 **기본 점수** 부여, **직접 로직 구현** 시 **가산점** 부여
9) **특정 기능 사용 금지**가 명시된 문항에서 해당 기능 사용 시 **감점 처리**
10) 사용자의 입력을 받는 **input 함수는 절대 사용 금지**
11) 요구하는 **형식**과 다른 정답을 반환할 시 **오답**으로 간주함

시험 코드 작성 유의 사항

최종 제출 코드가 다음 항목에 해당하는 경우, 감점 혹은 0점 처리 될 수 있음

1) Syntax Error로 인한 채점이 불가능한 경우
2) input 함수를 사용하는 경우
3) 주석 설명이 없거나 미흡한 경우
4) 출력 결과에 정답과 무관하거나 불필요한 내용이 있는 경우

※ 문제를 풀지 못하였다면 작성한 코드를 주석 처리하거나 삭제 후 pass 키워드 작성

※ 본 평가는 자동 채점 환경에서 채점이 이루어지므로, 상기 유의사항을 철저히 준수할 것

[파이썬 트랙] 1회차 월말평가 - Python

시험 환경

- Visual studio code(이하 vscode)를 이용한다.
- 그 외 (jupyter notebook, Pycharm, …) 사용 불가

코드 실행

- vscode의 터미널창에서 python 실행 명령어로 결과 확인을 **추천**

```
Edwin@LECTURE MINGW64 /d/SSAFY
$ python problem01.py
```

정답 제출 안내

제출 안내 사항 미 준수 시, 감점 혹은 0점 처리 될 수 있음

1) 압축 및 제출 파일 이름

- 지역_0반_홍길동

  ex) 서울_1반_홍길동 / 부울경_2반_김싸피

2) 압축 폴더 구조

- 시험을 진행했던 폴더 구조 그대로 압축하여 제출 진행
- 우측 구조에서 '서울_1반_김싸피' 폴더 선택 후 압축

```
서울_1반_김싸피/
    problem01.py
    problem02.py
    problem03.py
    ...
    problem11.py
```

제출 마감시간에 서버 요청이 집중될 수 있으므로, 미리 제출하는 것을 권장함

(마감 시간 이후 제출 불가)

[파이썬 트랙] 1회차 월말평가 - Python

문제 01 (problem01.py) – 9점

- 스마트팜 관리 시스템을 개발 중인 김싸피는 센서로부터 수집된 온습도 데이터를 분석하려고 합니다.
- 하루 동안 수집된 온도가 **리스트**로 주어질 때, 평균 온도를 **실수(float)** 형태로 반환하는 **calculate_avg** 함수를 완성하시오.

- **데이터 리스트에는 최소 1개 이상의 정수가 담겨 있습니다.**
- **결과값은 별도의 반올림 처리 없이 반환합니다.**

- 예시_01)

  입력: [20, 25, 30, 22, 28]

  출력: 25.0
- 예시_02)

  입력: [10, 10, 10]

  출력: 10.0
- 예시_03)

  입력: [1, 2, 4]

  출력: 2.3333333333333335

- **제한 내장 함수: len, sum**
- **기본 점수 (9점): 제한 내장 함수를 사용한 해결**
- **가산점(+3점): 제한 내장 함수 없이 직접 구현 (총 12점)**

[파이썬 트랙] 1회차 월말평가 - Python

문제 02 (problem02.py) – 9점

❖ 문서 처리 프로그램을 만들고 있습니다. 사용자가 입력한 단어 리스트 중에서, 특정 **길이(N) 이상**인 단어들의 **개수**를 반환하는 **count_long_words** 함수를 완성하시오.

❖ **단어 리스트는 문자열로 구성되어 있습니다.**

❖ **기준 길이 min_length는 양의 정수입니다.**

❖ 예시_01)

입력: ['apple', 'banana', 'cat', 'dog'], 4

출력: 2 ('apple', 'banana'가 길이가 4 이상)

❖ 예시_02)

입력: ['a', 'bb', 'ccc'], 5

출력: 0

❖ **제한 내장 함수: len**

❖ **기본 점수 (9점): 제한 내장 함수를 사용한 해결**

❖ **가산점(+3점): 제한 내장 함수 없이 직접 구현 (총 12점)**

[파이썬 트랙] 1회차 월말평가 - Python

문제 03 (problem03.py) – 9점

❖ 김싸피는 쇼핑몰 앱의 '최근 본 상품' 기능을 구현하고 있습니다. 사용자가 본 상품 번호가 방문한 시간 순서대로(과거→최신) 리스트에 저장되어 있을 때, 이를 **최신순(최신→과거)**으로 보여주기 위해 리스트의 순서를 거꾸로 뒤집어 반환하는 **reverse_list** 함수를 완성하시오.

❖ **단, 방문 기록이 없는 경우 빈 리스트가 주어질 수 있습니다.**

❖ **예시_01)**

입력: [1, 2, 3, 4, 5]

출력: [5, 4, 3, 2, 1]

❖ **예시_02)**

입력: ['A', 'B', 'C ']

출력: ['C', 'B', 'A']

❖ **제한 Python 내장 함수: reversed, list.reverse**

❖ **기본 점수 (9점): 제한 내장 함수를 사용한 해결**

❖ **가산점(+3점): 제한 내장 함수 없이 직접 구현 (총 12점)**

[파이썬 트랙] 1회차 월말평가 - Python

문제 04 (problem04.py) – 9점

❖ 싸피 카페의 키오스크 주문 데이터를 분석하려 합니다. 손님들이 주문한 메뉴 이름이 리스트로 주어질 때, 각 메뉴가 **몇 개씩 주문되었는지** 딕셔너리 형태로 반환하는 **count_menus** 함수를 완성하시오.

❖ **리스트에는 주문된 메뉴 이름이 문자열로 들어있습니다.**

❖ **단, 주문이 없는 경우 빈 리스트가 주어질 수 있습니다.**

❖ **반환 형태는 `{'메뉴이름': 주문수, ...}` 입니다.**

❖ **예시_01)**

입력: ['Ice Americano', 'Latte', 'Ice Americano', 'Mocha', 'Latte']

출력: {'Ice Americano': 2, 'Latte': 2, 'Mocha': 1} (순서는 무관)

❖ **예시_02)**

입력: ['Cocoa', 'Cocoa', 'Cocoa']

출력: {'Cocoa': 3}

❖ **제한 내장 기능: collections.Counter, List.count**

❖ **기본 점수 (9점): 제한 내장 기능을 사용한 해결**

❖ **가산점(+3점): 제한 내장 기능없이 직접 구현 (총 12점)**

[파이썬 트랙] 1회차 월말평가 - Python

문제 05 (problem05.py) – 9점

❖ 창고 관리자 김싸피는 여러 구역(행)으로 나뉜 창고의 물품 현황을 파악하고 있습니다. 구역별 물품 개수가 다를 수 있는 2차원 리스트 warehouse가 주어질 때, 빈 곳(0)을 제외한 실제 물품이 있는 칸의 총개수를 반환하는 **count_total_items** 함수를 완성하시오.

❖ **예시_01)**

입력: [[1, 0, 2], [3, 4, 5], [0]]

출력: 5

❖ **예시_02)**

입력: [[0, 0], [0], [30, 40]]

출력: 2

❖ **제한 내장 함수: sum, len, map**

❖ **기본 점수 (9점): 제한 내장 함수를 사용한 해결**

❖ **가산점(+3점): 제한 내장 함수 없이 직접 구현 (총 12점)**

[파이썬 트랙] 1회차 월말평가 - Python

문제 06 (problem06.py) – 9점

- N x N 2차원 정수 행렬이 주어질 때, 각 행의 최댓값들을 모두 더한 값을 반환하는 **sum_row_maximums** 함수를 완성하시오.
- 행렬의 원소는 절댓값이 10,000 이하인 정수입니다. (음수 포함 가능)
- 행렬의 크기 N은 1 이상 100 이하의 정수입니다.

- **예시_01)**

  입력: [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

  출력: 18 (3 + 6 + 9)

- **예시_02)**

  입력: [[-1, -2, -3], [-10, -5, -1], [-100, -200, -300]]

  출력: -102 ((-1) + (-1) + (-100))

- **예시_03)**

  입력: [[5]]

  출력: 5

- **제한 내장 함수: map, max, sum**
- **기본 점수 (9점): 제한 내장 함수를 사용한 해결**
- **가산점(+3점): 제한 내장 함수 없이 직접 구현 (총 12점)**

[파이썬 트랙] 1회차 월말평가 - Python

문제 07 (problem07.py) – 7점

❖ 도서관 관리자 김싸피는 도서 목록을 카테고리별로 정리하고 싶습니다.

❖ 도서 정보가 담긴 딕셔너리 리스트가 주어질 때, **카테고리(category)**를 **키(key)**로 하고, 해당 카테고리에 속한 **책 제목(title)의 리스트를 값(value)**으로 하는 딕셔너리를 반환하는 **categorize_books** 함수를 완성하시오.

❖ 예시) 입력: books_data = [
{'title': 'Python Basic', 'category': 'IT'},
{'title': 'Java Standard', 'category': 'IT'},
{'title': 'History of World', 'category': 'History'},
{'title': 'Cooking 101', 'category': 'Life'}
]

출력:

```
{
  'IT': ['Python Basic', 'Java Standard'],
  'History': ['History of World'],
  'Life': ['Cooking 101']
}
print(categorize_books(books_data))
```

[파이썬 트랙] 1회차 월말평가 - Python

문제 08 (problem08.py) – 7점

- 숫자 데이터 분석을 위해 리스트에서 두 번째로 큰 수를 찾아야 합니다.
- 정수 리스트 numbers가 주어질 때, 중복 없는 숫자들 중 두 번째로 큰 값을 반환하는 **find_second_largest** 함수를 완성하시오.

- **리스트에는 최소 2개 이상의 서로 다른 정수가 존재합니다.**

- **예시_01)**

  입력: [1, 5, 3, 8, 2]

  출력: 5

- **예시_02)**

  입력: [10, 10, 20, 5]

  출력: 10

[파이썬 트랙] 1회차 월말평가 - Python

문제 09 (problem09.py) – 7점

❖ 김싸피가 운영하는 유료 주차장의 요금 정산 시스템을 구현합니다.

❖ 차량들의 주차 시간(분)이 담긴 리스트가 주어질 때, 총 주차 요금을 계산하여 정수로 반환하는 **calculate_parking_fee** 함수를 완성하시오.

❖ **[요금 산정 규칙]**

1. **기본 요금**: 주차 시간이 **30분 이하**라면 기본 요금 **2000원**이 부과됩니다. (단, 0분 주차는 0원)
2. **추가 요금:** 30분을 초과하면, 기본 요금에 더해 **10분당 500원**씩 추가됩니다.
   - 10분 미만의 자투리 시간은 **올림**하여 10분으로 계산합니다.
   - 예: 31분: (기본 30분 + 추가 1분) → 10분 추가 요금 부과
   - 예: 41분: (기본 30분 + 추가 10분 + 추가 1분) → 20분 추가 요금 부과
3. **일일 최대 요금:** 계산된 요금이 **10000원을 초과**하면, **10000원**만 청구합니다.

❖ **주차 시간은 모두 양의 정수입니다.**

❖ **math 라이브러리(ceil 등) 사용 시 감점**

❖ **예시_1)**

입력: [25, 45, 300]

출력: 15000

계산과정:

1) 25분: 기본 시간 내 → 2000원

2) 45분: 30분(2000) + 15분(10분+5분 → 2단위 추가) = 2000 + 1000 = 3000원

3) 300분: 계산상 금액이 커도 최대 10000원 (15,500원이나 상한 적용)

❖ **예시_2)**

입력: [10, 0]

출력: 2000 # 2000 + 0 = 2000

[파이썬 트랙] 1회차 월말평가 - Python

문제 10 (problem10.py) – 7점

❖ 로봇 청소기가 청소 구역을 이동하려고 합니다. 로봇은 (0, 0) 좌표에서 시작하여 초기에는 **동쪽(East)**을 바라보고 있습니다. 문자열로 된 **이동 명령 commands**가 주어질 때, 모든 명령을 수행한 후의 **최종 좌표 (x, y)를 튜플 형태**로 반환하는 **robot_simulation** 함수를 완성하시오.

❖ **[명령어 규칙]**

- ❖**'G' (Go):** 현재 바라보는 방향으로 1칸 전진
- ❖**'B' (Back):** 현재 바라보는 방향의 반대쪽으로 1칸 후진 (바라보는 방향은 변하지 않음)
- ❖**'R' (Right):** 현재 방향 기준 오른쪽으로 90도 회전
- ❖**'L' (Left):** 현재 방향 기준 왼쪽으로 90도 회전

❖ **[방향 및 좌표 규칙]**

- ❖ **동쪽(East):** x좌표 +1
- ❖ **서쪽(West):** x좌표 -1
- ❖ **북쪽(North):** y좌표 +1
- ❖ **남쪽(South):** y좌표 -1

❖ **예시)**

입력: "GRGLGRG"

출력: (2, -2)

❖ **설명:**

1. **시작(0, 0, 동) → G → (1, 0, 동)**
2. **R회전 → (남쪽을 봄)**
3. **G → (1, -1, 남)**
4. **L회전 → (동쪽을 봄)**
5. **G → (2, -1, 동)**
6. **R회전 → (남쪽을 봄)**
7. **G → (2, -2, 남)**