COMENTO

직무부트캠프

[부천대] 실제 현업 WEB 개발자와 함께 SW 포트폴리오 제작까지!

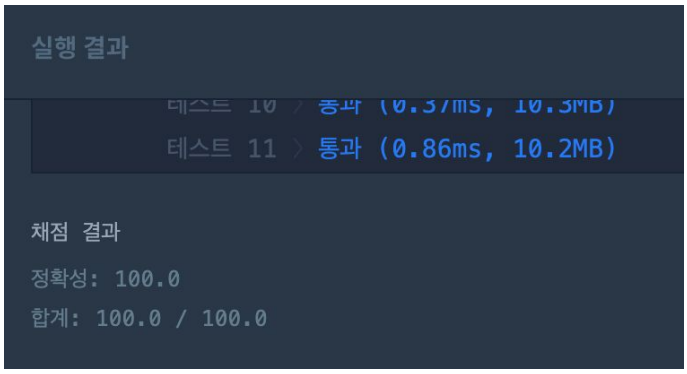
위승빈

1. 기본적인 Python 설치 및 기본 문법에 대해서 익히기

- Download Python 사이트를 이용하여 Python 패키지를 다운로드 후 설치하였습니다.
- Python의 IDE인 PyCharm을 다운로드 PyCharm 사이트를 이용하여 설치하였습니다.
- 주로 점프 투 파이썬 사이트를 이용하여 Python의 문법을 익혔습니다.

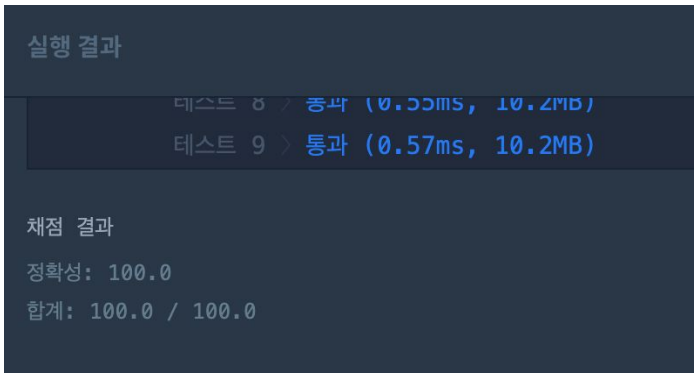
2. Programmers - Python3 Level1 10문제 이상 풀고 소스코드 제출

1. 크레인 인형뽑기 게임



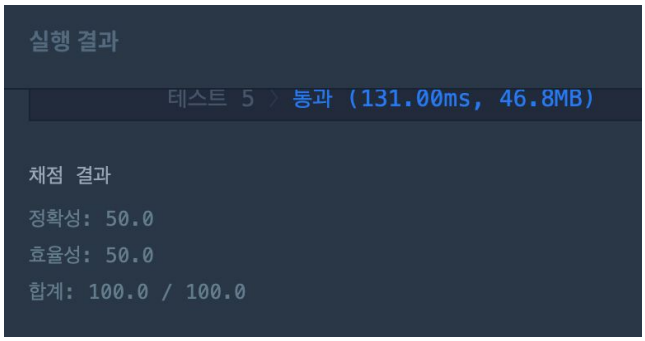

```
def solution(board, moves):  # 크레인 인형뽑기 게임
    answer = 0  # 사라진 인형
    result = []  # 결과 바구니
    nowResult = 0  # 현재 크레인의 결과

    for m in moves:
        for w in board:
            if w[m - 1] != 0:  # 크레인의 위치에 인형이 있다면
                nowResult = w[m - 1]  # 현재 크레인이 잡고있는 인형
                w[m - 1] = 0  # 인형이 있던 칸은 비워진다
                if len(result) > 0 and result[-1] == nowResult:  # 결과 바구니에 있는 인형이 하나 이상이며 결과 바구니의 마지막 인형과 현재 크레인 인형이 같다면
                    del result[-1]  # 바구니의 마지막 인형을 터트린다
                    answer += 2  # 사라진 인형에 터트려진 인형 수를 더한다 (바구니 마지막 인형 + 크레인 인형)
                    break
                result.append(nowResult)  # 인형이 터트려지지 않으면 결과 바구니에 쌓이게 된다
                break

    return answer
```

2. 두 개 뽑아서 더하기



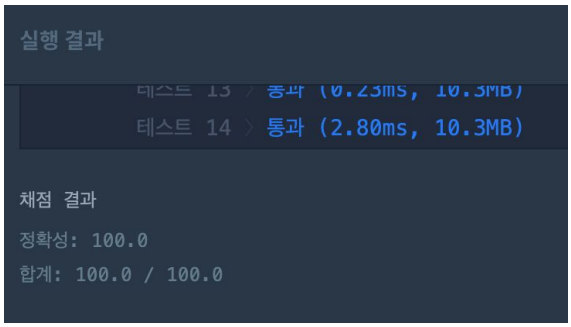

```python
def solution(numbers):  # 두 개 뽑아서 더하기
    answer = []
    for i, n in enumerate(numbers):
        for j in numbers[i + 1:]:  # numbers의 첫 요소와 그 다음 요소부터 numbers 모든 요소 반복
            answer.append(n + j)  # 요소 더하기

    answer = list(set(answer))  # set을 이용하여 중복 요소 제거
    answer.sort()  # 오름차순 정렬

    return answer
```

3. 완주하지 못한 선수



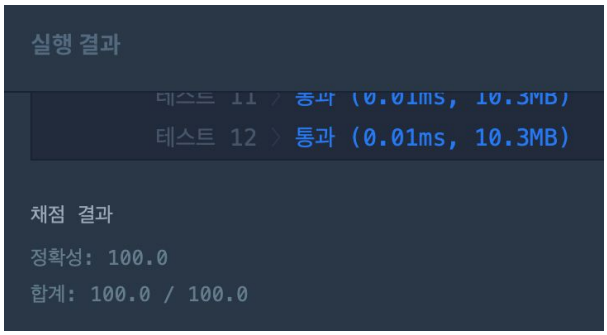

```python
def solution(participant, completion):  # 완주하지 못한 선수
    p = set(participant)  # 참여한 선수 동명이인 제거
    c = set(completion)  # 완주한 선수 동명이인 제거
    result = (list(p - c))  # 완주하지 못한 선수 추출

    if len(result) == 1:  # 완주한 선수가 추출됐다면 (한명이라면)
        answer = result[0]  # 완주하지 못한 선수는 해당 선수
    else:  # 동명이인이 완주하지 못한 선수일 경우
        pDic = {}  # 참여한 선수 딕셔너리
        cDic = {}  # 완주한 선수 딕셔너리

        for i in participant:  # pDic key-value
            try:
                pDic[i] += 1
            except:
                pDic[i] = 1

        for i in completion:  # cDic key-value
            try:
                cDic[i] += 1
            except:
                cDic[i] = 1

        for key in pDic:
            if pDic[key] != cDic[key]:  # 해당 선수의 value값이 다르다면
                answer = key  # 해당 선수가 완주하지 못한 선수 (동명이인)

    return answer
```

4. 모의고사



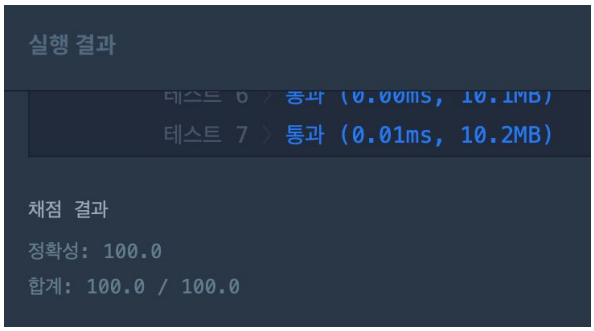

```
def solution(answers):  # 모의고사
    supo1 = [1, 2, 3, 4, 5]  # 1번 수포자가 찍는 방식
    supo2 = [2, 1, 2, 3, 2, 4, 2, 5]  # 2번 수포자가 찍는 방식
    supo3 = [3, 3, 1, 1, 2, 2, 4, 4, 5, 5]  # 3번 수포자가 찍는 방식
    result = [0, 0, 0]  # 각 수포자들 맞힌 문제 개수
    count = [len(supo1), len(supo2), len(supo3)]  # 각 수포자들의 찍는 방식 길이 (pattern Loop)

    for i, a in enumerate(answers):  # 모의고사 문제만큼 반복
        if a == supo1[i % count[0]]:  # 1번 수포자의 찍는 방식의 답과 정답이 같다면 (모의고사 문제번호에 pattern Loop를 적용하여 수포자의 찍는 답 추출)
            result[0] += 1  # 해당 수포자의 맞힌 문제 개수 증가
        if a == supo2[i % count[1]]:  # 2번 수포자
            result[1] += 1
        if a == supo3[i % count[2]]:  # 3번 수포자
            result[2] += 1

    maxResult = max(result)  # 가장 많이 맞힌 문제

return [i + 1 for i, r in enumerate(result) if maxResult == r]  # result의 요소가 maxResult와 같다면 가장 많은 문제를 맞힌 사람에 추가
```

5. 체육복



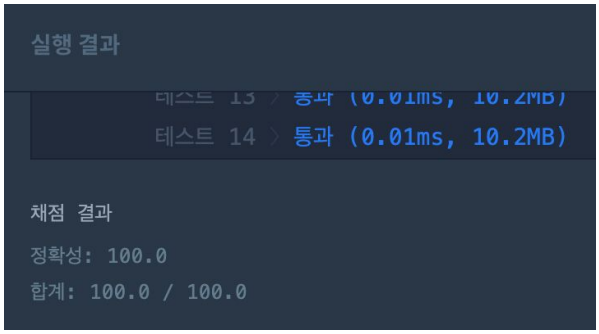

```
def solution(n, lost, reserve):  # 체육복
    clothes = [1] * n  # 전체 학생의 체육복

    for i in lost:
        clothes[i - 1] = 0  # 도난당한 학생의 체육복 제거
    for i in reserve:
        clothes[i - 1] += 1  # 여벌의 체육복을 가져온 학생의 체육복 추가

    for i, c in enumerate(clothes):
        if c > 1 and i != 0 and clothes[i - 1] == 0:  # 여벌의 체육복이 있으며 첫번째 번호가 아니고 앞번호 학생의 체육복이 없을 시
            clothes[i - 1] += 1  # 앞번호 학생에게 체육복 빌려주기
            c -= 1  # 해당 학생의 빌려준 체육복 제거
        if c > 1 and i != n - 1 and clothes[i + 1] == 0:  # 여벌의 체육복이 있으면 마지막 번호가 아니고 뒷번호 학생의 체육복이 없을 시
            clothes[i + 1] += 1  # 뒷번호 학생에게 체육복 빌려주기
            c -= 1  # 해당 학생의 빌려준 체육복 제거

    return n - clothes.count(0)  # 체육수업을 들을 수 있는 학생 수
```

6. K번째수



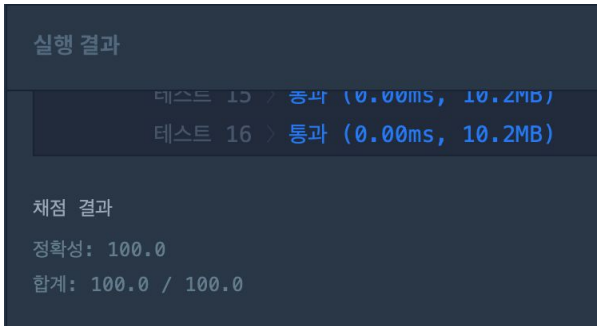

```python
def solution(array, commands):  # K번째수
    answer = []

    for c in commands:
        resultArray = array[c[0] - 1: c[1]]  # 배열의 첫번째 번호부터 두번째 번호까지 array를 슬라이싱
        resultArray.sort()  # 결과 정렬
        answer.append(resultArray[c[2] - 1])  # K번째 수 정답으로 추가

    return answer
```

7. 2016년




```python
def solution(a, b):  # 2016년
    DoW = ["SUN", "MON", "TUE", "WED", "THU", "FRI", "SAT"] # 일요일부터 토요일
    day = 5  # 2016년의 시작하는 요일 초기화 (금요일)

    for i in range(1, a):  # 1월부터 a달 전달까지 날짜 계산
        if i in (1, 3, 5, 7, 8, 10, 12):  # 해당 달이면
            day += 31  # 31일 추가
        elif i in (4, 6, 9, 11):  # 해당 달이면
            day += 30  # 30일 추가
        else:  # 2월이라면
            day += 29  # 29일 추가 (2016년은 윤년)

    day += b  # 날짜에 현재 날짜 추가
    week = day % 7  # 현재 날짜 일주일 단위로 계산

    return DoW[week - 1]
```

8. 가운데 글자 가져오기

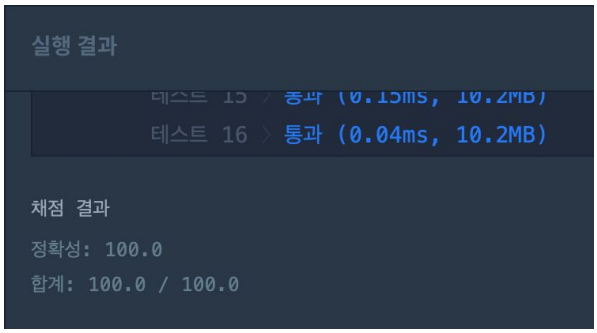



```
def solution(s):  # 가운데 글자 가져오기
    l = len(s)  # 단어 s의 길이
    if l % 2 == 0:  # 짝수라면
        answer = s[int(l / 2) - 1] + s[int(l / 2)]  # 가운데 두 글자
    else:  # 홀수라면
        answer = s[int(l / 2 - 0.5)]  # 가운데 한 글자

    return answer
```

9. 같은 숫자는 싫어

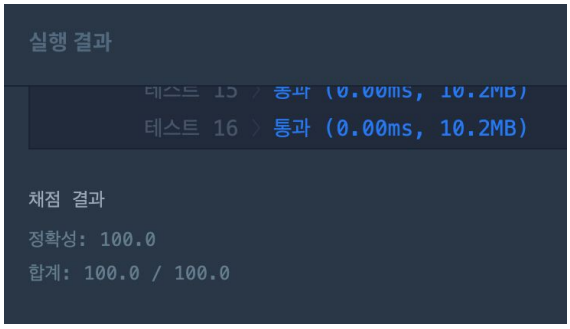

```
def solution(arr):  # 같은 숫자는 싫어
    pre = arr[0]  # arr의 첫번째 요소로 이전 숫자 초기화
    answer = [pre]  # 정답에 pre로 초기화

    for i in arr:
        if pre != i:  # 현재 숫자가 이전 숫자와 다르다면
            answer.append(i)  # 정답에 현재 숫자 추가
            pre = i  # 이전 숫자를 현재 숫자로 변경

    return answer
```

10. 나누어 떨어지는 숫자 배열



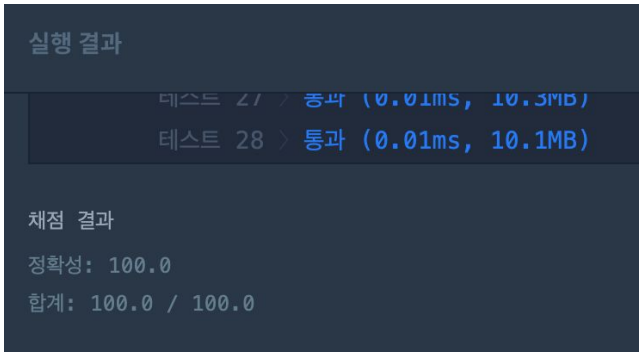

```python
def solution(arr, divisor):  # 나누어 떨어지는 숫자 배열
    answer = [i for i in arr if i % divisor == 0]  # arr의 요소가 divisor로 나누어 떨어지는 경우 정답에 추가
    answer.sort()  # 오름차순 정렬
    if len(answer) < 1:  # 나누어도 떨어지는 숫자가 없다면
        answer.append(-1)

    return answer
```

11. 두 정수 사이의 합



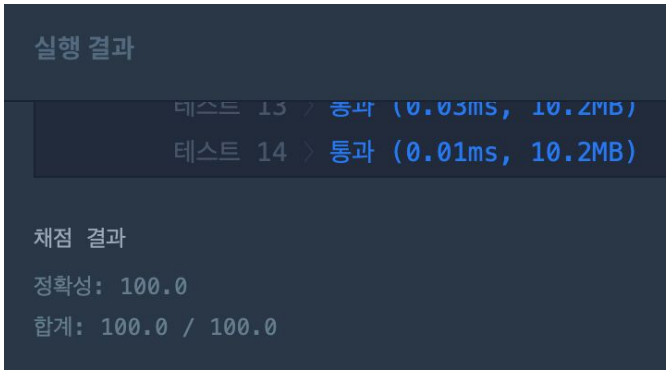

```python
def solution(a, b):  # 두 정수 사이의 합
    if a > b:  # a보다 b가 더 큰 수라면 두 수 교환
        temp = a
        a = b
        b = temp

    count = a - b  # 두 정수 사이의 정수의 개수 - 1
    if count < 0:  # 음수라면 양수로 변환
        count = -count

    answer = a * (count + 1)  # 작은 수 * 두 정수 사이의 정수의 개수
    answer += (count + 1) * count // 2  # 두 정수 사이의 정수의 개수 * count(이미 계산된 a를 제외한 정수의 개수) // 2

    return answer
```

12. 문자열 내 p와 y의 개수



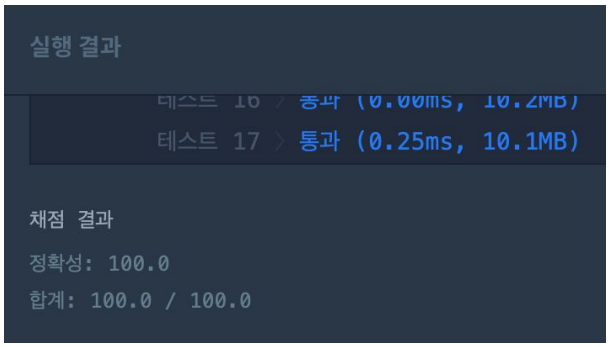

```python
def solution(s):  # 문자열 내 p와 y의 개수
    pCount = yCount = 0  # p와 y의 개수 초기화

    for i in s:
        if i == 'p' or i == 'P':
            pCount += 1
        if i == 'y' or i == 'Y':
            yCount += 1

    return pCount == yCount  # p와 y의 개수가 같다면 True 아니라면 False
```

13. 서울에서 김서방 찾기




```python
def solution(seoul):  # 서울에서 김서방 찾기
    return '김서방은 ' + str(seoul.index("Kim")) + '에 있다'  # 김서방이 있는 인덱스 찾아 출력
```

14. 약수의 합




```python
def solution(n):  # 약수의 합
    answer = 0  # 정답 초기화
    for i in range(1, n + 1):  # 1부터 n까지 반복
        if n % i == 0:  # 약수라면
            answer += i  # 정답에 약수 더하기

    return answer
```

3. Python 기본 문법에 대한 설명

파이썬은 인터프리터 방식, 객체지향, 동적 타입 언어로, 다른 언어에 비해 간결하며 문법이 쉽다는 특징이 있습니다.

아래는 Python 문법의 특징 중 일부입니다.

- 변수는 자료형을 따로 선언하지 않고 변수에 정수, 문자열, 리스트 등을 대입하게 되면 해당 자료형 객체가 됩니다.
- 제어문은 if, for, while 문을 작성하고 ':' 기호 바로 아래 문장부터 같은 너비의 들여쓰기로 수행할 문장을 작성해야 합니다.
- 다중 조건 판단시 else if가 아닌 elif로 작성합니다.
- for문, if문은 간결하게 한 줄로 작성이 가능합니다.
- 함수는 'def 함수명(매개변수):' 아래 수행 문장을 작성합니다.
- 'lambda 매개변수: 수행문장' 처럼 lambda 예약어를 통해 함수를 간결하게 작성할 수 있습니다.
- 클래스는 'class 클래스명:' 아래 클래스 내용을 작성합니다.
- .py로 만든 파일은 모두 모듈이 됩니다.
- 'import 모듈명'을 작성해 모듈을 불러올 수 있습니다. 모듈명에는 확장명을 적지 않습니다.
- import된 모듈의 함수를 이용할 때는 '모듈명.함수명(매개변수)'으로 작성합니다.
- len(), abs(), max(), min(), sum(), range(), enumerate(), split(), str() 등 다양하고 유용한 내장 함수들이 있습니다.

Python의 다양한 모듈 활용을 더불어 장점을 살려 다른 언어에 비해 빠른 구현이 가능하지만 스타일 제한이 엄격해 문법을 신경써서 작성해야 합니다.