자료구조 과제#1 _1712070 김소영

<자체 평가표>

문제	평가 기준	배점	자체 평가 점수
1 장-25 번	시간복잡도 – 정답에 대한 설명이 포함되었는가?	5	5
2 장 10-(2)	희소행렬 – 행렬이 올바르게 출력되는가?	15	15
2 장 12-(1)	볼링 점수 계산 – 미확정 프레임 점수 출력이 보류되는가? 합계가 맞는가?	30	30
3 장-2(최대값)	재귀함수의 실행 결과가 올바르게 출력되는가?	12	12
3 장 3 번	재귀함수의 실행 결과가 올바르게 출력되는가?	13	13
3 장 6 번	순열의 출력 순서가 올바르게 출력되는가?	12	12
3 장 8 번	재귀함수의 실행 결과가 올바르게 출력되는가?	13	13
	합계	100	100

<1 장 연습문제>

25 번

	Steps	**Frequency**	**Total steps**
def isum(num):	0	0	0
total = 0	1	1	1
for k in range (1, n+1, 2):	1	n/2	n/2
for j in range(n):	1	n*(n/2)	$n^2/2$
total = total + num	1	n*(n/2)	$n^2/2$
return total	1	1	1
			전체 합 = $n^2+n/2+2$

정답 2 번 : $n^2+n/2+2$

<2 장 연습문제>

10 번-(2)

튜플의 리스트를 먼저 딕셔너리에 저장해준다. 이중 for 문을 이용해 각 행과 열을 돌면서, get()을 사용하여 해당하는 key 가 없다면, 0 을 출력하도록 했다.

```
In [86]: runfile('C:/Users/USER/.spyder-py3/2장_10(2)번.py', wdir='C:/Users/
USER/.spyder-py3')
0 3 0 2
8 0 4 0
0 0 0 5
```

12 번-(1) : P.42 의 프로그램 2.12 의 score1 점수를 키보드로 입력한 결과를 출력한다.

우선 for 문으로 1 부터 보너스 프레임까지 입력을 받도록 했다. 첫 번째 점수와 두 번째 그리고 총 합을 리스트에 저장하도록 했다. 각 점수의 상황에 맞게 result 에 spare 와 strike 를 저장했다. Result 에 맞게 이전에 입력됐던 점수를 가져와 계산하도록 코드를 작성했다. Strike 가 나온 경우, score[i-1][2] += total 를 써서 이전 점수(total)에 현재 점수를 더해서 보류되었던 점수가 다시 출력이 되도록 했다. spare(/), strike(X), none(-) 상황에 따라 다르게 표시되도록 mark 에 저장했고, print(f"({l}, {m}, '{mark}', {n})", end = "") 이런 식으로 작성해 전체 출력되도록 했다.

```
In [91]: runfile('C:/Users/USER/.spyder-py3/2장_12(1)번.py', wdir='C:/Users/USER/.spyder-py3')

(입력) 1 프레임 : 8 0
(8, 0, '-', 8)
Total =  8
(8, 0, '-', 8)

(입력) 2 프레임 : 4 3
(4, 3, '-', 7)
Total =  15
(8, 0, '-', 8)(4, 3, '-', 7)

(입력) 3 프레임 : 8 2
(8, 2, '/', )
Total =  25
(8, 0, '-', 8)(4, 3, '-', 7)(8, 2, '/' ,  )

(입력) 4 프레임 : 4 6
(4, 6, '/', )
Total =  39
(8, 0, '-', 8)(4, 3, '-', 7)(8, 2, '/', 14)(4, 6, '/' ,  )

(입력) 5 프레임 : 2 6
(2, 6, '-', 8)
Total =  49
(8, 0, '-', 8)(4, 3, '-', 7)(8, 2, '/', 14)(4, 6, '/', 12)(2, 6, '-', 8)

(입력) 6 프레임 : 10 0
(10, 0, 'X', )
Total =  59
(8, 0, '-', 8)(4, 3, '-', 7)(8, 2, '/', 14)(4, 6, '/', 12)(2, 6, '-', 8)(10, 0, 'X' ,  )
```

```
(입력) 7 프레임 : 9 0
(9, 0, '-', 9)
Total =  77
(8, 0, '-', 8)(4, 3, '-', 7)(8, 2, '/', 14)(4, 6, '/', 12)(2, 6, '-', 8)(10, 0, 'X', 19)(9, 0, '-', 9)

(입력) 8 프레임 : 10 0
(10, 0, 'X', )
Total =  87
(8, 0, '-', 8)(4, 3, '-', 7)(8, 2, '/', 14)(4, 6, '/', 12)(2, 6, '-', 8)(10, 0, 'X', 19)(9, 0, '-', 9)(10, 0, 'X' ,  )

(입력) 9 프레임 : 8 2
(8, 2, '/', )
Total =  107
(8, 0, '-', 8)(4, 3, '-', 7)(8, 2, '/', 14)(4, 6, '/', 12)(2, 6, '-', 8)(10, 0, 'X', 19)(9, 0, '-', 9)(10, 0, 'X', 20)
(8, 2, '/' ,  )

(입력) 10 프레임 : 10 0
(10, 0, 'X', )
Total =  127
(8, 0, '-', 8)(4, 3, '-', 7)(8, 2, '/', 14)(4, 6, '/', 12)(2, 6, '-', 8)(10, 0, 'X', 19)(9, 0, '-', 9)(10, 0, 'X', 20)
(8, 2, '/', 20)(10, 0, 'X' ,  )

(입력) 보너스 프레임 : 10 10
(10, 10, '-', 20)
Total =  147
(8, 0, '-', 8)(4, 3, '-', 7)(8, 2, '/', 14)(4, 6, '/', 12)(2, 6, '-', 8)(10, 0, 'X', 19)(9, 0, '-', 9)(10, 0, 'X', 20)
(8, 2, '/', 20)(10, 0, 'X', 30)(10, 10, '-', 20)
```

<3 장 연습문제>

2 번(최대값)

lst = [3, 28, 34, 200, 13, 20, 65] 로 리스트를 만들어 주었고, 함수 Find_Max 를 구현하였다.

리스트의 길이가 1 개일 때와 아닐 때로 구분하였다. 아닐 경우엔 재귀호출을 이용하여 리스트 안의 원소 중 가장 큰 값을 찾아낼 때까지 반복하도록 했다. max_b = Find_Max(array, l-1)에서 재귀호출해서 그 값을 max_b 에 저장했다. 원소끼리 값을 비교했고 리스트의 전체 원소들을 하나씩 비교하며 최대값을 리턴하도록 코드를 작성했다.

```
In [113]: runfile('C:/Users/USER/.spyder-py3/3장_2번.py', wdir='C:/Users/
USER/.spyder-py3')
200
```

3 번

Word 리스트를 만들어 처음과 끝 번호를 지정해주고 같은 지 판단하도록 코드를 작성했다.

Return palindrome(word, s+1, e-1)를 이용해 단어 전체를 확인할 수 있도록 했다.

```
In [114]: runfile('C:/Users/USER/.spyder-py3/3장_3번.py', wdir='C:/Users/
USER/.spyder-py3')

단어를 입력하세요 : abcdedcba
palindrome이 맞습니다.

In [115]: runfile('C:/Users/USER/.spyder-py3/3장_3번.py', wdir='C:/Users/
USER/.spyder-py3')

단어를 입력하세요 : apple
palindrome이 아닙니다.
```

6 번

Permutation 함수를 만들어주고 단어의 처음부터 끝을 돌면서 위치를 바꾸도록 코드를 작성했다.

s[i], s[j] = s[j], s[i]로 위치를 바꾸도록 했고 i 를 하나 늘려서 permutation 을 다시 돌도록 했다. 그리고 그 끝을 만나게 될 경우 if i == length: print(''.join(s), end = ', ') 코드를 이용해 각 알파벳을 합치도록 했다.

```
In [105]: runfile('C:/Users/USER/.spyder-py3/3장_6번.py', wdir='C:/Users/
USER/.spyder-py3')
LANG, LAGN, LNAG, LNGA, LGNA, LGAN, ALNG, ALGN, ANLG, ANGL, AGNL, AGLN, NALG,
NAGL, NLAG, NLGA, NGLA, NGAL, GANL, GALN, GNAL, GNLA, GLNA, GLAN,
```

8 번

harmonic 함수를 만들어서 n<2 일때와 아닐 때로 경우를 나누었다. n 이 2 보다 크다면 return 1/n + (harmonic(n-1))을 통해 1/n + 1/(n-1) +1/(n-2)+…+ 1/2 이렇게 입력 받은 값에서 n 을 2 까지 줄여가며 더한다. 그 후 n=1 이 되면 return 1 이 되니, 결국 조화 수를 구할 수 있게 되는 것이다.

```
In [108]: runfile('C:/Users/USER/.spyder-py3/3장_8번.py', wdir='C:/Users/
USER/.spyder-py3')

n을 입력하세요 : 6
2.4499999999999997

In [109]: runfile('C:/Users/USER/.spyder-py3/3장_8번.py', wdir='C:/Users/
USER/.spyder-py3')

n을 입력하세요 : 2
1.5
```