2023년 2학기

프로그래밍과 문제해결

Assignment #1

담당 교수: 윤은영

학번: 20230499

학과: 무은재학부

이름: 김재환

POVIS ID: carotinoid

명예서약(Honor code)

"나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다."

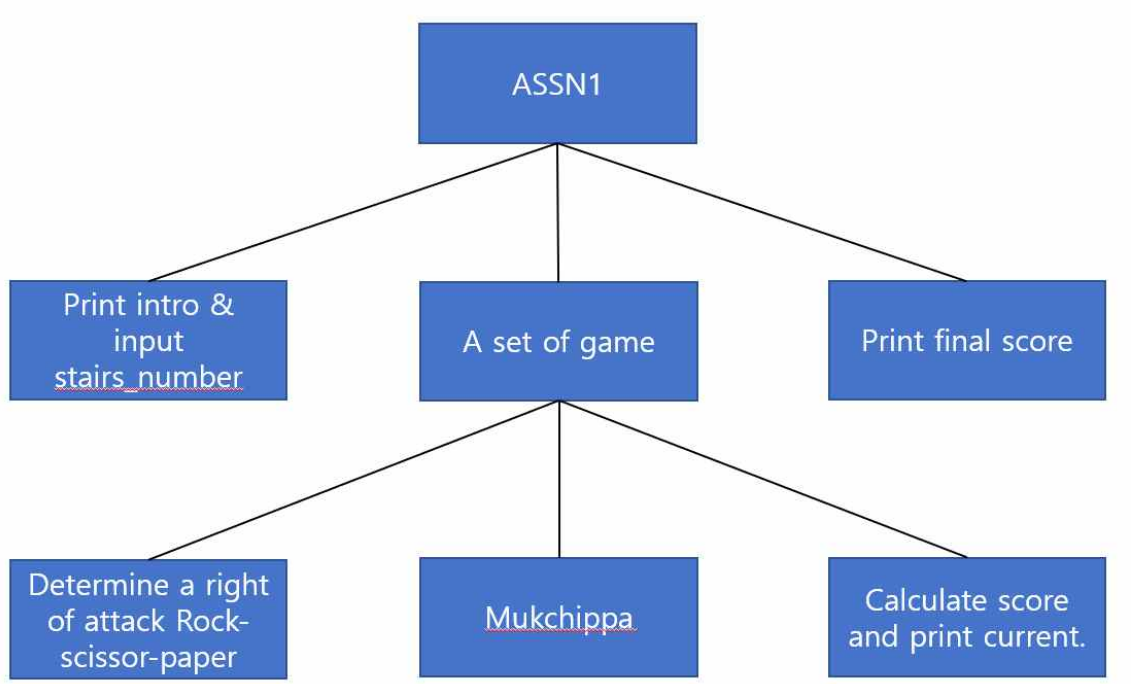
---

1. 문제의 개요

본 프로그램을 간략히 설명하면 다음과 같다.

- 계단 수를 입력받고, 그에 따른 상황을 출력한다.
- 프로그램이 끝날 때까지, 여러 경기를 진행한다.
- 한 경기는, 공격권 결정 가위바위보, 묵찌빠, 점수 계산으로 이루어진다.

이때 사용되는 구조 차트는 다음과 같이 나타낼 수 있다.



본 프로그램은 게임 진행 전, 진행 중, 진행 후 마무리 3단계로 나눌 수 있으며, 게임 진행 중은 공격권 결정 가위바위보, 묵찌빠, 점수 결산 3단계로 다시 나눌 수 있다.

2. 알고리즘

본 프로그램 작성을 위해서 크게 중요한 부분은 세 부분으로 나눌 수 있다. 1. 계단 출력, 2. 공격권 결정 가위바위보, 3. 묵찌빠. 이를 Pseudo 코드 형태로 나타내면 다음과 같다. calculate_result 등은 프로그램 내에서 정의되어있는 함수로, 이름대로의 역할을 수행하면서 코드가 간단하기에 본 Pseudo 코드에서는 나타내지 않고 사용하였다. 또한 엔터 입력 대기 및 화면 지우기 코드는 본 Pseudo 코드에서 생략하였다.

Pseudo-algorithm for Print_stairs

```
// 프로그램에 필요한 변수들은 미리 선언해 놓은 것으로 가정한다.

print stairs_number, player score, computer score
calculate rows and cols of stairs
declare 2-dimension list stairs
for I in range(1, rows)
        for j in range(0, I)
                stairs[i][j] <-- '▨'  # 계단 왼쪽부분
        for j in range(cols - 1, cols-1-i, -1)
                stairs[i][j] <-- '▨'  # 계단 오른쪽 부분
if player <= stairs_number // 2
        set player'○'  on left side of stairs
else
        set player'○'  on right side of stairs
if computer <= stairs_number // 2 #right side of stairs
        if players position and computer are same
                set '◑'
        else
                set '●'
else
if players position and computer are same
                set '◑'
        else
                set '●'
print stairs with nested loop
```

Pseudo-algorithm for Determining the right of attack RSP(function)

```
// 프로그램에 필요한 변수들은 미리 선언해 놓은 것으로 가정한다.

while true
        input player choice
        determine computer choice (random)
        attacker <-- calculate_result
        if attacker is player or computer:
                print result
                return attacker
        if attacker is 'same'
                comtinue
```

Pseudo-algorithm for Mukchippa(function)

```
// 프로그램에 필요한 변수들은 미리 선언해 놓은 것으로 가정한다.

0    get attacker from parameter
1    while true:
2            round += 1
3            input player choice
4            determine computer choice (random)
5            next_attacker <-- calculate_result
6            if next_attacker is 'same':
7                    break
8            attacker <-- next_attacker
9    return attacker, round.
```

Pseudo-algorithm for entire structure

```
// 프로그램에 필요한 변수들은 미리 선언해 놓은 것으로 가정한다.

1    print_intro
2    input_stairs_number
3
4    while True:
5            print_stairs
6            initial_RSP
7            Mukchippa
8            determine winner/loser, calculate score
9            check if game be end
10   print outro
```

3. 프로그램 구조 및 설명

a) 초기화면 출력 및 계단 수 입력

- 묵찌빠 게임 소개를 위한 초기화면을 출력하고, 총 승리에 필요한 점수인 계단의 수에 대한 입력을 요청한다.

b) 계단 출력

- 계단을 그리기에 필요한 계단 수, 플레이어와 컴퓨터의 현재 점수를 매개변수로 받는다.
- 계단을 그리기에 앞서 출력을 위한 2차원 리스트를 선언한다. 이때 행과 열은 각각 (stairs_number + 1) // 2 + 1, stairs_number + 1로 계산된다. 계단의 위치 외에도 플레이어와 컴퓨터 기호의 위치를 고려하여 행의 마지막에 1을 더해준다.
- 중첩 반복문을 2번 사용해서 왼쪽 내려가는 계단, 오른쪽 올라가는 계단(플레이어 기준)

을 각각 리스트에 입력한다.

- 플레이어의 점수를 이용해 플레이어 위치를 리스트에 입력한다. 이때 리스트 인덱스의 시작이 1이 아니라 0임을 고려해야 하며, 점수가 계단 수의 절반을 넘어가는 시점을 기준으로 하여 다른 인덱스를 적용해야 한다.
- 같은 방식으로 컴퓨터의 위치를 리스트에 입력한다. 이때 해당 위치에 플레이어의 기호가 있는지 검사하고, 결과에 따라 다른 기호를 입력해야 한다.

c) 공격권 결정 가위바위보

- 플레이어의 입력을 요청한다. '가위', '바위', '보' 이외의 값을 입력받으면 제대로 된 입력을 받을 때까지 다시 입력을 요청한다.
- random.randint() 함수를 이용해 컴퓨터의 선택을 결정한다.
- 플레이어와 컴퓨터의 선택을 입력받으면 calculate_result() 함수를 호출하여 결과를 계산하고, 공격권자를 출력한다. 둘의 선택이 같으면 처음으로 돌아가 반복한다.

d) 묵찌빠

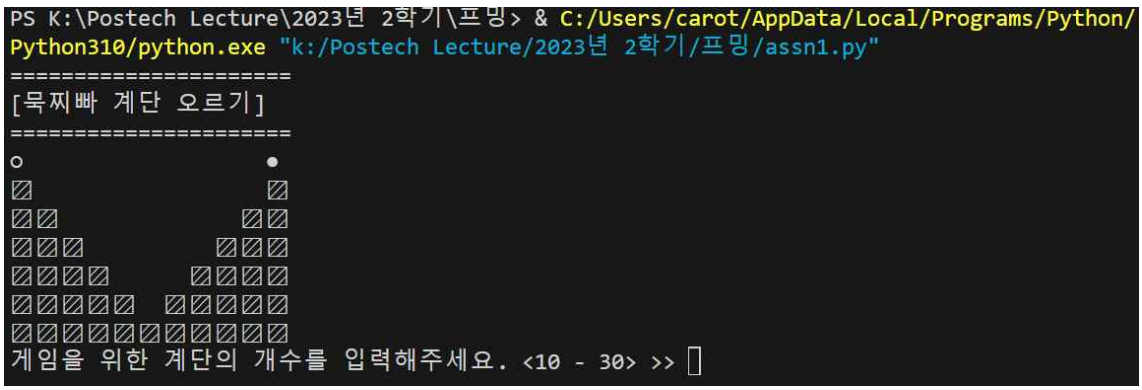
- 공격권 결정 가위바위보로부터 결정된 공격권자를 매개변수로 받는다.
- 공격권 결정 가위바위보와 비슷하게 진행되나, 저번 라운드의 공격권자(이기지 않았을 경우)와 현재 라운드에서 승리한 사람이 같아야 이기므로 이전 값을 저장하고 현재 공격권자와 비교할 필요해야한다..

e) 점수 계산 및 최종 결과 출력

- 묵찌빠가 진행된 라운드 수 만큼 승리자의 점수에 더한다.
- 최종 점수가 계단 수를 초과할 경우, 최종 점수 출력에서는 최종 점수를 계단 수로 표기해야한다.

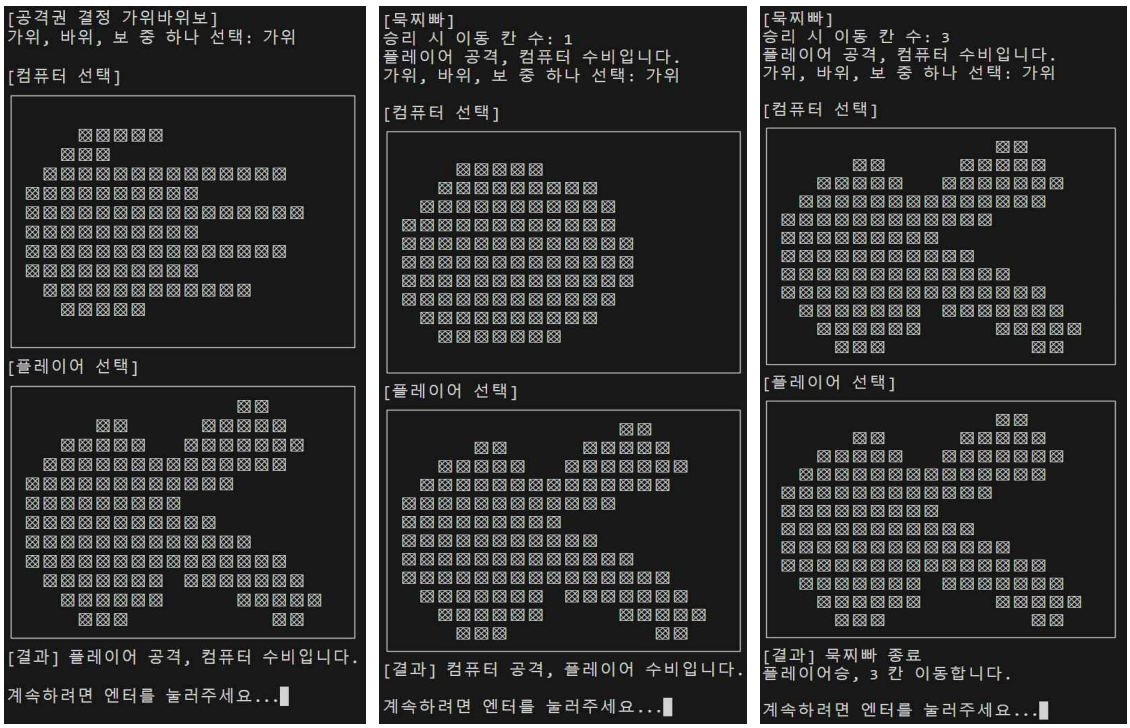
4. 프로그래밍 실행 방법 및 예시

처음에 프로그램을 실행하면 프로그램의 설명(print_intro)가 출력되며, 계단의 수를 입력받는 화면이 나오게 된다.



이때 계단의 수가 10개인 것은 입력받은 계단의 개수에 따른 것이 아니며, 그저 설명을 위한 예시 그림을 초기화면에 출력한 것이다. 계단 수가 10과 30 사이의 값이 아니라면 다시 입력받는다. 해당 예시에서는 13을 입력하였다.

print_stairs 함수를 호출하여 계단 수를 입력하면 입력받은 계단 수와 플레이어와 컴퓨터의 아이콘과 현재 점수, 그리고 계단 수에 맞는 계단을 출력한다. 이후 공격권 결정 가위바위보 / 묵찌빠를 진행한 이후 다시 print_stairs 함수를 호출하게 된다.



각각 공격권 결정 가위바위보, 묵찌빠, 묵찌빠 마지막 출력이다.

공격권 결정 가위바위보에서는 플레이어에게 가위, 바위, 보 중 하나를 입력받고, 랜덤하게 선택된 컴퓨터의 선택과 플레이어의 선택을 출력한다. 그리고 계산된 결과를 출력하며, 무승부일 경우 승패가 결정될 때까지 가위바위보를 반복한다.

공격권 결정 가위바위보에서 결정된 공격권자를 묵찌빠 함수에 전달하여 묵찌빠를 진행한

다. 가위바위보와 비슷하지만, 해당 라운드에서 이긴 사람은 다음 라운드의 공격권자가 되며, 둘의 선택이 같아질 때까지 반복한다. 둘의 선택이 같다면 공격권자가 묵찌빠를 승리하게 되며, 그동안 진행한 라운드 수만큼 점수를 얻는다.

공격권 결정 가위바위보와 묵찌빠 한 세트를 진행한 이후, 다시 현재 상황을 출력한다. 이후 다시 공격권 결정 가위바위보와 묵찌빠, 현재 상황 출력을 계속해서 반복한다.

현재 상황 출력(print_stairs), 가위바위보, 묵찌빠를 반복하면서 진행하다가 누군가의 점수가 계단 수 이상이 되면 결과를 출력하며 프로그램을 종료한다. 이때 점수가 계단 수를 초과하더라도 최종 점수는 계단 수를 초과하지 않는다.

5. 토론

- 변수의 값이나 프로그래밍 코드에 한글을 넣는 것을 선호하지는 않으나, 프로그램의 출력에 대부분 한글을 사용하기 때문에 플레이어와 컴퓨터, 가위, 바위, 보를 한글로 처리하면 변수를 그대로 출력할 수 있어서 유리하다. 마찬가지로 처음에는 3의 배수와 나머지를 이용해 가위, 바위, 보를 처리하면 더 간단하게 코드를 구성할 수 있으나 코드가 난해할 수도 있고, 한글로 대입하는 것이 출력 면에서 가독성이 좋다고 판단하였다.
- 화면을 지우기 위해 실행환경마다 다른 코드가 요구된다. 보고서의 예시 화면은 윈도우

의 vscode에서 실행된 환경이며, 첨부된 코드 파일은 윈도우에 맞는 코드가 사용되어 있다. 다른 환경에서 실행하면 코드 상단 주석에 있는 설명에 따라 주석을 추가하거나 # 기호를 제거해 환경을 맞춘 후 실행해야 한다.

- 기본적으로 기호를 이용해 그림을 그릴 때 가장 힘든 점은 문자마다 너비가 다르다는 점이었다. 이를 해결하기 위해 문자의 특징을 찾아보았고, 고정폭 글꼴을 사용하면 띄어쓰기로 인한 칸수 문제를 해결할 수 있었다.
- 계단을 출력하기 위해 2차원 리스트에 그릴 때, 행의 크기가 n으로 선언했다면 그 인덱스는 0부터 n-1이다. 또한 계단의 윗부분에 플레이어와 컴퓨터 아이콘이 들어갈 한 줄이 띄워져 있어야 하므로 계단의 행 번호는 1부터 잡아야 했다. 이런 점 때문에 반복문이 돌아갈 횟수가 1씩 차이가 나는 것에 어려움을 겪었다.

6. 결론

- 본 과제에서는 프로그램을 작성할 때 함수와 반복, 조건문을 이용함으로써 간결하고 알아보기 쉽게 흐름을 제어하는 방법을 학습할 수 있었다. 특히 비슷한 부분에서 사용되는 코드를 함수화하여 여러 부분에서 한 줄로 불러올 수 있어 코드를 절약함과 동시에 코드의 작동을 이해하기 쉽게 만드는 방법을 알 수 있었다.

7. 개선 방향

- 본 과제를 수행하면서 계단 출력에 대한 반복문 작성에 어려움을 겪었고, 그 과정에서 2차원 리스트의 인덱스를 표기하는 것이 깔끔하지 못하였다.
- 적절한 함수명, 변수명을 설정하는 방법을 더 찾아보고 적용하면 가독성이 좋아질 것이다.