2023년 2학기

프로그래밍과 문제해결

Assignment #1

담당교수: 윤은영

학번: 20230673

학과: 무은재학부

이름: 전재영

명예서약(Honor code)

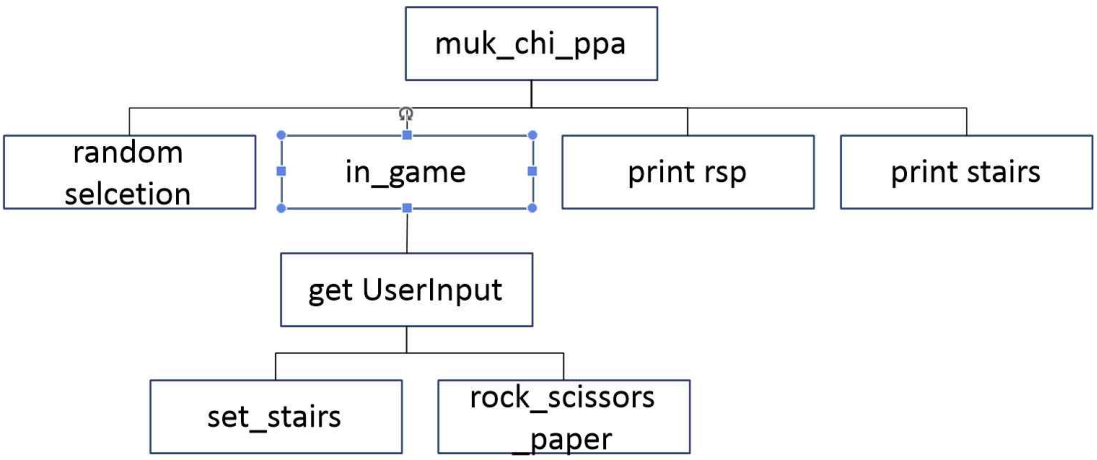
"나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다."

Problem: 묵찌빠 계단오르기

1. 문제의 개요

본 프로그램을 간략히 설명하면 다음과 같다.
- 계단의 개수를 유저의 입력으로 받는다.
- 유저의 입력을 받아 컴퓨터와 가위바위보를 통해 공격권을 정한다.
- 컴퓨터의 가위바위보 선택은 랜덤으로 정한다.
- 묵찌빠를 진행해 (묵찌빠를 한 횟수)만큼 계단을 이동한다.
- 유저와 컴퓨터의 위치를 출력한다.

이때 사용되는 구상 가능한 구조 차트(structure chart)는 아래와 같이 표현될 수 있다.



2. 알고리즘

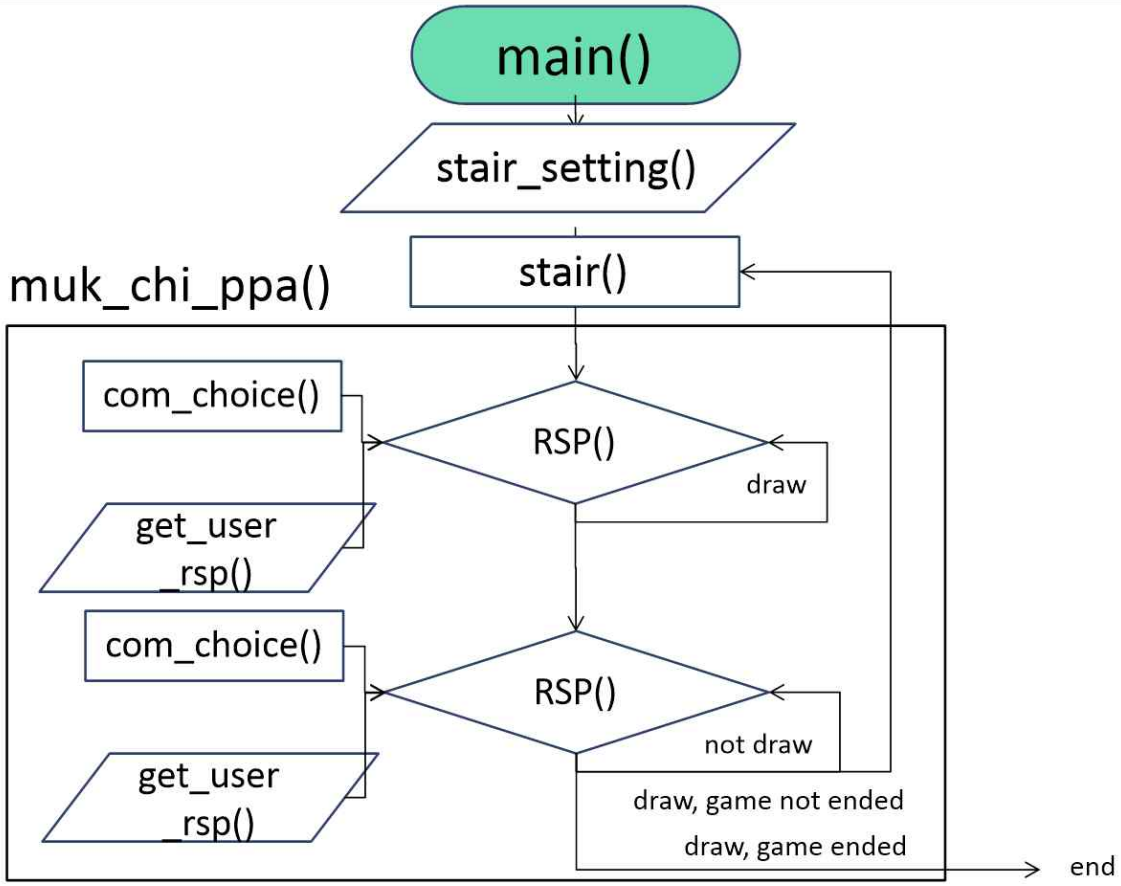
본 프로그램 작성을 위한 알고리즘을 Pseudo 코드 형태로 나타내면 다음과 같다.

Pseudo-algorithm for game
  // 프로그램에 필요한 변수들은 미리 선언해놓은 것으로 가정한다

```
1 include essential module random, os, clear
2 get user's input -> setting the number of stair
3 while (game is end) // muk_chi_ppa
4         print stair & user/computer's place
5         get user's input -> rock/scissors/paper // first rsp
6         if draw, do rsp again
7         if user win, return status : win
8         if user lose, return status : lose
9
```

```
10      while (rsp result == draw)
11              default score = 1
12              get user's input -> rock/scissors/paper // rsp for 묵찌빠
13              if user win, return status : win
14              if user lose, return status : lose
15              if draw:
16                      if status == win -> user get score
17                      if status == lose -> computer get score
18
19              score += 1
20      move user or computer
21
22 if game end, print result
```

위의 의사 알고리즘을 Flowchart를 통해 표현하면 다음과 같다.



3. 프로그램 구조 및 설명

먼저 프로그램을 이루는 주요 변수들과 함수들을 설명하고, 구조를 설명하겠습니다.

변수 :

num_stair : 유저의 입력을 통해 받은 계단의 개수이다.

user_place, com_place : 컴퓨터와 유저의 위치를 나타내는 변수이다. stair_ui[]의 인덱스에 해당한다.

user_r, user_c, com_r, com_c : stair_ui[][]에서의 인덱스로, 계단에서 구체적인 유저와 컴퓨터의 x값과 y값을 나타내는 변수이다.

status : 가위바위보의 결과를 저장하는 변수이다. 각각 -1은 비겼을 때, 0은 이겼을 때, 1은 졌을 때를 나타낸다.

user_win : 가위바위보의 결과를 저장하는 변수이다. 유저가 이겼을 때 True, 졌을 때 False 값을 갖는다.

score : 이겼을 때 계단을 얼마나 이동할지 나타내는 변수다. 묵찌빠에서 승부가 나지 않을 경우 1씩 누적된다.

함수 :

stair_setting() : 유저의 입력을 받아 계단의 개수를 설정하는 함수다.

stair() : 계단과 유저와 컴퓨터의 위치를 출력하는 함수다.

where_is_user() : 유저의 위치를 list에서의 row, column으로 반환하는 함수이다.

where_is_com() : 컴퓨터의 위치를 list에서의 row, column으로 반환하는 함수이다.

print_rock() : 바위을 출력한다.

print_paper() : 보를 출력한다.

print_scissors() : 가위를 출력한다.

com_choice() : 1~3 중의 랜덤한 수를 골라 반환한다.

get_user_rsp() : 유저의 “가위”, “바위”, “보”를 입력받는 함수이다.

RSP() : 유저와 컴퓨터의 가위바위보 결과를 통해 누가 이겼는지를 판단하는 함수이다.

muck_chi_ppa() : RSP() 함수를 이용해 묵찌빠를 진행하는 함수이다.

print_rsp() : 컴퓨터와 유저의 가위바위보 결과를 출력한다.

display_place() : 총 계단수, 플레이어의 위치와 컴퓨터의 위치를 숫자로 표현하는 함수이다.

clear_screen() : 단순히 화면을 clear하는 함수이다.

enter_to_conti() :

구조도는 다음과 같습니다.

a) 계단의 입력

- 초기 계단의 상태(10개)를 출력하고, stair_setting() 함수를 통해 유저의 입력을 받습니다. 그 후, 화면을 지우고 입력받은 수만큼의 계단을 만들어 출력합니다. 계단의 공백을 표현하기 위해서는 띄어쓰기 ‘ ’을 사용하였습니다.

b) 게임 시작

- 어느 한쪽이 다른 한쪽의 계단의 시작점에 도달할 때까지 묵찌빠를 반복합니다.
- muk_chi_ppa() 함수에서 묵찌빠를 진행하고 승자와 이동해야 할 결과를 받아와 유저와 컴퓨터를 이동해 계단을 출력합니다.

c-1) 묵찌빠의 진행 (공격권 결정 가위바위보)

- muk_chi_ppa()가 실행되었을 때, 공격권 결정 가위바위보를 진행합니다.
- RSP() 함수를 실행해 유저의 가위바위보 입력을 받고, 컴퓨터의 랜덤 선택과 비교해 누가 이겼는

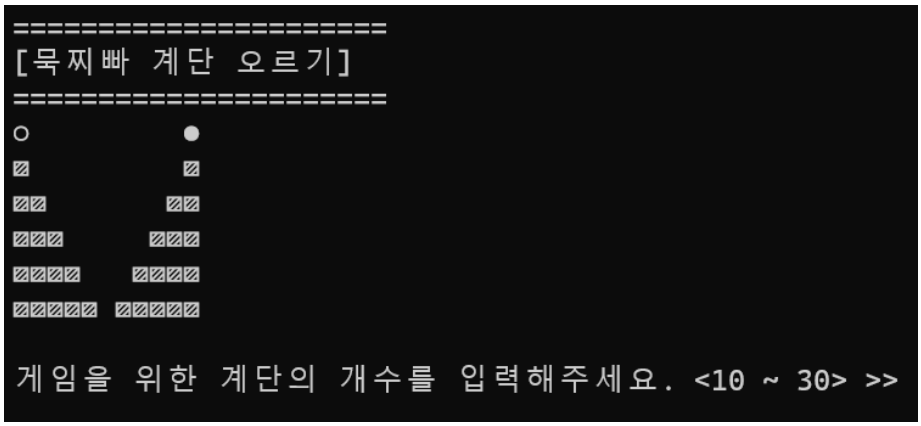
지를 반환합니다. 비겼을 경우 가위바위보를 다시 진행하고, 유저가 이겼을 경우 status = 0, 졌을 경우 status = 1를 반환합니다.
- 공격권이 결정된 경우 묵찌빠 단계로 진행됩니다.

c-2) 묵찌빠의 진행 (묵찌빠 단계)
- 공격권이 결정이 된 후, 다시 가위바위보를 진행합니다.
- 가위바위보를 이겼을 경우, 유저가 공격권을 갖고, 졌을 경우, 컴퓨터가 공격권을 가져갑니다. 그 후, 점수에 1을 더하고, 다시 가위바위보를 진행합니다.
- 만약 비겼을 경우 묵찌빠가 종료됩니다. 공격권을 갖고 있는 사람이 점수를 얻게 되고, 그 사람이 누적된 점수만큼 이동하게 됩니다.

d) 묵찌빠 종료 후 계단 출력
- muk_chi_ppa() 함수에서 묵찌빠를 진행하고 승자와 이동해야 할 결과를 받아와 stair()를 통해 유저 또는 컴퓨터의 위치를 이동시킵니다.
- 만약 유저가 컴퓨터의 시작 위치 (오른쪽의 맨 끝)까지 이동했을 경우, 유저의 승리를 나타내며 프로그램을 종료합니다.
- 만약 컴퓨터가 유저의 시작 위치 (왼쪽의 맨 끝)까지 이동했을 경우, 컴퓨터의 승리를 나타내며 프로그램을 종료합니다.

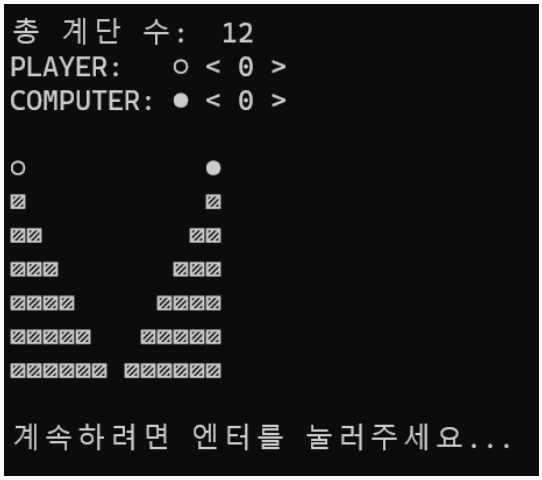
4. 프로그램 실행방법 및 예제
- 윈도우 vscode 환경에서 프로그램을 제작했으며, window prompt를 통해 실행하였다.

[예제 1]



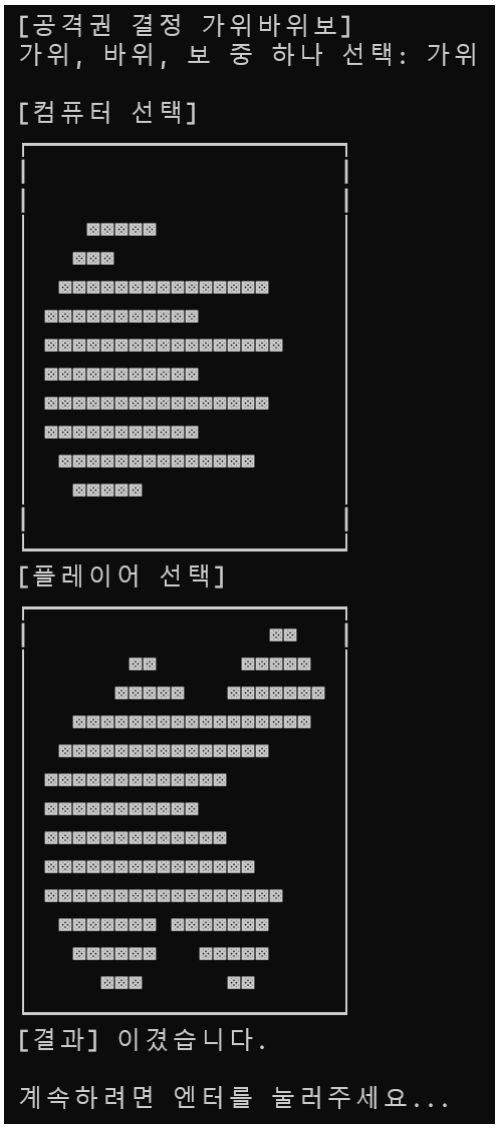
게임을 시작했을 때, 기본적으로 설정된 10개의 계단의 모습을 출력하고, 유저로부터 계단의 수를 입력받는다.

[예제 2]



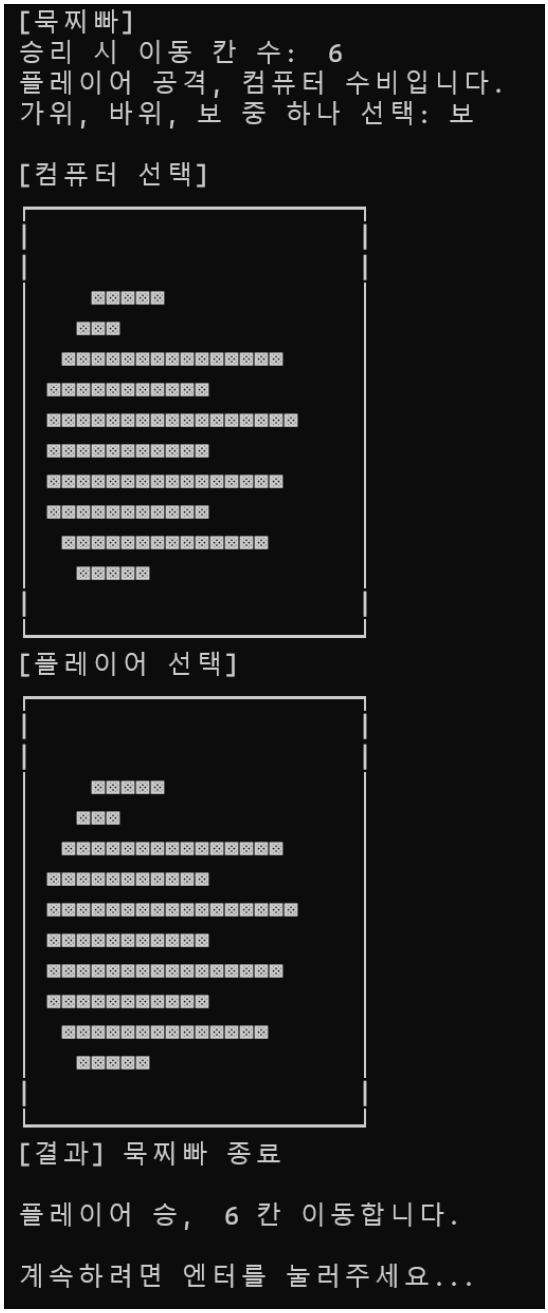
계단을 12개 입력했을 때의 모습을 출력하고, 게임이 진행된다.

[예제 3]



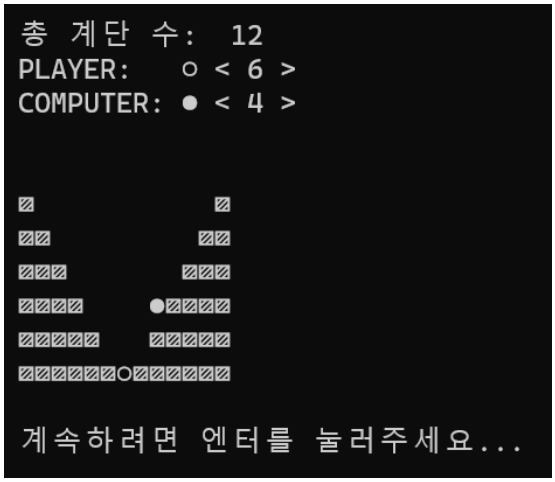
묵찌빠를 시작할 때, 가위바위보를 통해 공격권을 정한다.

[예제 4]



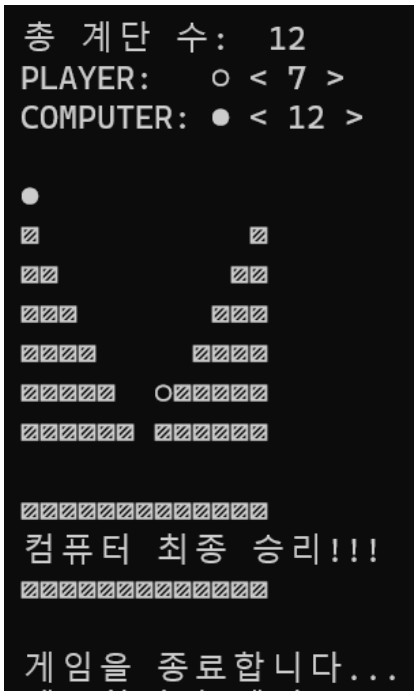
묵찌빠를 진행할 때마다 승리시 이동 칸수가 올라가며, 공격권을 가진 사람이 이겼을 때, 묵찌빠를 종료하고 누적된 점수만큼 계단을 이동한다.

[예제 5]



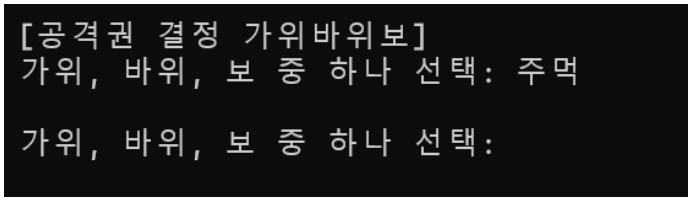
묵찌빠가 끝난 후, 결과를 반영하여 출력한다. 플레이어는 6칸, 컴퓨터는 4칸 이동한 것을 알 수 있다.

[예제 6]



한쪽이 다른 한쪽의 시작 지점에 도달했을 경우, 게임 결과를 출력하고, 프로그램을 종료한다.

[예외]



입력 형식과 상관없는 입력을 받는 경우, 다시 입력을 받는다.

5. 토론

- 사용한 모듈은 컴퓨터의 의사결정을 위한 random과 화면을 clear하기 위한 os와 IPython.display::clear_output를 사용했다.
- 유저의 초기 위치를 0, 컴퓨터의 초기 위치를 (입력받은 계단의 개수)로 설정했는데, 이는 유저와 컴퓨터의 위치를 인덱스로 보아 리스트에 원활히 넣기 위해서였다. 화면에 표시될 때는 컴퓨터의 위치는 (계단의 개수 - 컴퓨터의 좌표)로 계산해, 이동한 거리로 표시했다.
- stair() 함수에서 계단을 구성하는 문자와 플레이어, 컴퓨터의 위치를 나타내는 문자가 모두 한 리스트에 담겨서 출력된다. 이때, where_is_user/com() 함수를 통해 ~_r, ~_c를 return 받는데, 각각의 2중 리스트에서의 row와 column을 나타낸다.
- print_stair(…) -> stair(…)와 비슷하지만, 플레이어의 현재 이동한 계단 수와 이동해야 할 총 계단 수를 매개변수를 받는 것이 아니라 user/com_place로 좌표를 입력받아. 출력한다는 점이 다르다.

6. 결론

- 본 과제를 통해 기본적인 파이썬에서의 입력과 출력, 반복문과 조건문을 익히는데 유용했으며, 여러 함수를 사용하면서 프로그램을 구조화하는 방법을 알 수 있었다. 또한, 입력받은 수를 통해 계단과 유저, 컴퓨터의 위치를 나타내기 위해 리스트를 사용하면서, 이중 리스트와 리스트의 사용법을 익힐 수 있었던 것 같다. 간단한 알고리즘만 주로 다루었던 전과 달리, 여러 함수를 사용해 하나의 제대로 동작하는 프로그램을 만들었다는 경험을 통해, 더 복잡한 프로그램을 만들 수 있을 것 같다는 자신을 얻을 수 있었다.

7. 개선방향

- 묵찌빠를 진행하는 함수는 만들었지만, 묵찌빠를 반복하는 함수는 만들지 않아 main()에서 while(1)을 통해 게임이 종료될 때까지 묵찌빠를 반복하는 코드를 구현하였다. 그래서 메인에서의 코드가 꽤나 난잡해지고, 알기 어려웠던 것 같다.
- 묵찌빠를 진행할 때, 가위바위보 결과를 return 하는 과정에서 status와 user_win이라는 변수가 혼용되어 사용되는데, 사실상 status가 user_win을 포함하고 있지만 둘 다 사용되면서 코드가 길어지고 복잡해졌던 것 같다.

참고 :
점프 투 파이썬 : List