Honor pledge :

I affirm that I will not give or receive any unauthorized help on this exam, and that all work will be my own.

분반 : 03반

학번 : 20202971

이름 : 박윤혜

1. Random, time 모듈 가져오기

```
import random   # random 모듈 호출
import time     # time 모듈 호출

print('\nHangman 게임에 오신 것을 환영합니다!\n')   # 시작 문구
name = input('이름을 입력하세요 : ')               # 이름 입력
print('안녕하세요 {}님!'.format(name))             # 인삿말
time.sleep(1)   # 1초동안 정지
print('기회는 총 5번입니다. 행운을 빌어요 :)\nGame Start!') # 도전 가능 횟수를 알려주고 게임시작을 알림
time.sleep(2)   # 2초동안 정지
```

코드 설명 :

· import random : 리스트에 있는 항목을 무작위로 뽑을 때 사용한다.

· import time : 시간에 관한 정보를 제공하는 모듈이다.

· time.sleep() : 일정 시간동안 프로그램이 멈추도록 하는 데 사용된다.

2. hangman() 함수

```
def hangman() : # hangman()함수 정의
    global count    # count 전역변수 선언
    global limit    # limit 전역변수 선언
    global word_guess   # word_guess 전역변수 선언
    global sep      # sep 전역변수 선언
    global word     # word 전역변수 선언

    word_list = ['march','music','math','yellow','kangaroo','coffee','water',      # Hangman 게임에 사용할 단어들을 list에 넣고 그 리스트를 word_list에 할당
                 'computer','camera','photo','purple','orange','chicken','pizza']
    word = random.choice(word_list) # word_list에 있는 단어들 중 무작위로 하나를 뽑음 >> Ex) word = math
    w_list = list(word)             # word를 리스트 형식으로 바꾸고 w_list에 할당함 >> Ex) w_list = ['m','a','t','h']
    word_length = '_'* len(word)    # word 길이만큼 '_'를 만들고 word_length에 할당함  >> Ex) word_length = '____' (if word = math)
    word_guess = list(word_length)  # word_length를 리스트 형식으로 바꾸고 word_guess에 할당함  >> Ex) word_guess = ['_','_','_','_']
    already_guess = []              # already_guess에 빈 리스트 할당 (already_guess의 초기값 설정)
    sep = ''    # sep에 ''를 할당함
    count = 0   # count에 0을 할당함 (count의 초기값 설정)
    limit = 5   # limit에 5를 할당함 (기회가 5번 있음)
```

코드 설명 :

· global count, global limit, global word_guess, global sep, global word 전역변수를 선언하고 초기값을 할당한다. global 키워드를 사용해서 다른 함수에서도 이 변수를 사용할 수 있도록 한다.

· word_list : Hangman 게임에서 사용할 단어들을 모아놓은 리스트이다.

· word : 사용자가 맞춰야 할 단어이다. Random 모듈에 있는 choice 함수를 사용하여 word_list 내의 임의의 요소가 하나 선택되어 할당된다.

· w_list : word에 포함된 알파벳 하나하나를 원소로 가지는 리스트이다. 사용자가 단어에 포함된 알파벳을 맞추면 그 알파벳을 삭제하기 위해서 사용된다.

· word_length : len()함수를 사용하여 word 길이만큼의 '_'를 만들어서 사용자에게 단어의 길이를 알려주는 데 사용된다.

· word_guess : 사용자가 단어에 포함된 알파벳을 맞추면 그 알파벳이 어느 자리에 들어가는지 알려주기 위해 사용된다.

· already_guess : 사용자가 알파벳을 입력하면 그 알파벳이 리스트의 원소로 추가된다. 중복 입력을 방지하기 위해 사용된다.

· sep : 나중에 word_guess 리스트의 원소들을 하나의 문자열로 나타내기 위해서 join()함수와 함께 사용된다.

· count : 사용자가 한번 틀릴 때마다 1씩 증가한다. 초기값은 0이고, 최대 5까지 올라간다.

· limit : 사용자가 한번 틀릴 때마다 1씩 감소한다. 기회는 5번 있기 때문에 초기값은 5이며, 0이되면 Game Over가 된다.

```
while limit != 0 :  # limit가 0이 아니면(도전 기회가 남아있으면) 반복함
    if sep.join(word_guess) == word :   # sep.join(word_guess)가 정답(word)이랑 같으면 (정답을 맞췄으면)
        print('정답입니다! 축하드려요 :)')  # 정답이라고 알려주고
        break   # 반복문을 나감

    # 정답을 맞추지 않은 상태라면 계속 진행
    guess = input('Hangman 단어입니다 "{}". 알파벳을 하나 입력하세요 : '.format(sep.join(word_guess)))  # 알파벳을 입력받아서 guess에 할당함
    guess = guess.strip()   # guess에 공백을 입력했을 경우를 대비해서 공백을 제거해줌

    while (guess in already_guess) or ((guess.isalpha() and len(guess)==1) == False) :  # 입력한 적이 있었던 알파벳을 입력했거나, 알파벳을 올바르게 입력하지 않은경우 반복함
        while guess in already_guess :  # guess가 already_guess에 포함되어 있으면(입력한 적이 있었던 알파벳을 또 입력했으면)
            guess = input('이미 입력했던 알파벳입니다. 다시 입력해주세요 : ')   # 이미 입력했었던 알파벳이라고 알려주고 다시 입력하라고 함

        while (guess.isalpha() and len(guess)==1) == False :   # guess가 알파벳이 아니거나 guess의 길이가 1이 아니면
            guess = input('잘못 입력하셨습니다. 알파벳 하나로 다시 입력해주세요 : ')    # 잘못 입력했음을 알려주고 다시 입력하라고 함
            guess = guess.strip()   # guess에 공백을 입력했을 경우를 대비해서 공백을 제거해줌
    guess = guess.lower()   # guess의 알파벳을 소문자로 변환 (대문자를 입력했을 때를 대비하기 위함)

    if guess in w_list :    # guess가 w_list에 있으면
        while guess in w_list :  # w_list에 guess가 없을때까지 반복한다
            word_index = w_list.index(guess)  # w_list에 있는 guess의 인덱스를 word_index에 할당
            word_guess[word_index] = guess    # word_guess의 word_index번째 원소를 guess로 바꿈
            w_list[word_index] = '' # w_list의 word_index번째 원소를 ''(빈문자열)로 바꿈
        print(sep.join(word_guess)) # word_guess 리스트의 원소들을 다 합쳐서 문자열 형태로 출력한다.
    else :          # guess가 w_list에 없으면
        count += 1  # count값을 1 증가시키고
        limit -= 1  # limit값을 1 감소시키고
        hangman_picture()   # hangman_picture() 함수 실행
    already_guess.append(guess) # already_guess 리스트에 guess를 원소로 추가
```

· guess : 사용자로부터 input()함수를 통해 알파벳 하나를 입력받은 후, strip()함수를 통해 공백을 제거해주고 lower()함수를 통해 알파벳을 소문자로 변환시킨다.

· guess에 입력받은 문자가 알파벳인지, 하나의 알파벳을 입력했는지, 이전에 이미 입력받았던 알파벳은 아닌지 확인하고 잘못 입력했거나 이미 입력받았던 알파벳이라면 제대로 입력할 때까지 다시 입력하라고 알려준다. 이때 isalpha()는 영어, 한글 등의 글자로만 이루어졌으면 True, 아니면 False를 반환한다.

· 입력받은 알파벳이 단어에 포함되어 있으면 index()함수를 통해 단어에서 그 알파벳의 위치를 찾는다. word_guess에는 그 위치에 있던 문자를 입력받은 알파벳으로 바꾸고, w_list에는 그 위치에 있던 문자를 빈 문자열로 바꾼다. 최종적으로 word_guess에는 사용자가 맞춘 알파벳들이 들어가고 w_list에는 사용자가 맞춘 알파벳들이 없어진다.

· join()함수를 이용하여 word_guess의 원소들을 하나의 문자열로 나타낸다. 사용자가 맞춘 알파벳이 어느 자리에 들어갔고, 남은 자리는 어디인지를 알려준다.

· 입력받은 알파벳이 단어에 포함되어 있지 않으면 count 값을 1 증가시키고, limit 값을 1 감소시킨다. 그 후 hangman_picture()함수를 실행시킨다.

· 입력받은 알파벳은 already_guess에 추가시켜서 중복 입력을 방지시킨다.

· limit가 0이 될 때까지 (기회를 다 쓸 때까지) 이 과정을 반복한다. 만약 사용자가 단어를 맞췄으면 break을 통해 반복문을 나간다.

```
again = input('다시 도전하겠습니까? (Y/N) : ')     # 기회를 다 사용했거나 정답을 맞췄으면 다시 도전할건지 물어봄
while again.strip() not in ['Y', 'N', 'y', 'n'] :  # 정해진 형식('Y','N','y','n')으로 대답하지 않으면
    again = input('Y/N으로 입력해주세요. 다시 도전하시겠습니까? ')  # 다시 입력하라고 함
if again.strip() == ('Y' or 'y') :  # 다시 도전한다고 대답하면
    hangman()   #hangman()함수 실행 (게임 다시 시작)
else :          # 다시 도전을 안 한다고 했으면
    print('게임을 종료합니다.') # 게임을 종료한다고 말하고
    exit()  # 나감
```

· again : 기회를 다 썼거나 (limit = 0) 정답을 맞췄으면 다시 도전할 것인지 물어보고 input()을 통해 대답을 입력받는다.

· 대답이 'Y'나 'y'이면 다시 도전하겠다는 의미이므로 hangman()함수를 다시 실행시키고, 'N' 혹은 'n'이면 다시 도전하지 않겠다는 의미이므로 프로그램을 종료시킨다. 만약 정해진 형식대로 대답하지 않으면 똑바로 입력할 때까지 다시 입력받는다.

3. hangman_picture() 함수

```
def hangman_picture() : # hangman_picture()함수 정의 (hangman 그림을 나타내기 위함)
    global count    # count 전역변수 선언
    global limit    # limit 전역변수 선언
    global word_guess   # word_guess 전역변수 선언
    global sep      # sep 전역변수 선언
    global word     # word 전역변수 선언
    if count == 1 :     # (기회를 1번 써서) count가 1이면
        time.sleep(0.5) # 0.5초동안 정지
        print("  _______ \n"    # 첫번째 그림 출력
                "  |      \n"
                "  |      \n"
                "  |      \n"
                "  |      \n"
                "  |      \n"
                "  |      \n"
                "__|__\n")
        print('틀렸습니다! 기회는 {}번 남았습니다.'.format(limit))  # 틀렸다고 말해주고 남은 기회를 알려줌

    elif count == 2 :   # count가 2이면
        time.sleep(0.5) # 0.5초동안 정지
        print("  _______ \n"    # 두번째 그림 출력
                "  |     | \n"
                "  |     |\n"
                "  |      \n"
                "  |      \n"
                "  |      \n"
                "  |      \n"
                "__|__\n")
        print('틀렸습니다! 기회는 {}번 남았습니다.'.format(limit))  # 틀렸다고 말해주고 남은 기회를 알려줌

    elif count == 3 :   # count가 3이면
        time.sleep(0.5) # 0.5초동안 정지
        print("  _______ \n"    # 세번째 그림 출력
                "  |     | \n"
                "  |     |\n"
                "  |     | \n"
                "  |      \n"
                "  |      \n"
                "  |      \n"
                "__|__\n")
        print('틀렸습니다! 기회는 {}번 남았습니다.'.format(limit))  # 틀렸다고 말해주고 남은 기회를 알려줌
```

```
    elif count == 4 :   # count가 4이면
        time.sleep(0.5) # 0.5초동안 정지
        print("  _______ \n"    # 네번째 그림 출력
                "  ¦     ¦ \n"
                "  ¦     ¦\n"
                "  ¦     ¦ \n"
                "  ¦     O \n"
                "  ¦      \n"
                "  ¦      \n"
                "__¦__\n")
        print('틀렸습니다! 기회는 {}번 남았습니다.'.format(limit))  # 틀렸다고 말해주고 남은 기회를 알려줌

    elif count == 5 :   # count가 5이면 (기회 모두 사용)
        time.sleep(0.5) # 0.5초동안 정지
        print("  _______ \n"    # 마지막 그림 출력
                "  ¦     ¦ \n"
                "  ¦     ¦\n"
                "  ¦     ¦ \n"
                "  ¦     O \n"
                "  ¦    /¦\ \n"
                "  ¦    / \ \n"
                "__¦__\n")
        print('Game Over')  # Game Over이라고 알려주고
        print('Hangman 단어는 "{}"였습니다.'.format(word))  # 정답을 알려줌

hangman()   # hangman()함수 실행
```

· hangman()함수에서 선언했던 전역변수들을 모두 호출한다.

· count의 값에 따라 그림을 출력하고 남은 기회(limit)를 알려준다.

· count의 값이 5일 때 limit의 값은 0이므로 Game Over이라고 알려주고 정답을 알려준다.

· hangman_picture()함수는 사용자가 단어 속 알파벳을 맞추지 못했을 때에만 실행된다.

References

· https://data-flair.training/blogs/hangman-game-python-code/

· 컴퓨팅 사고 (김완섭)

· 으뜸 파이썬 (박동규, 강영민)