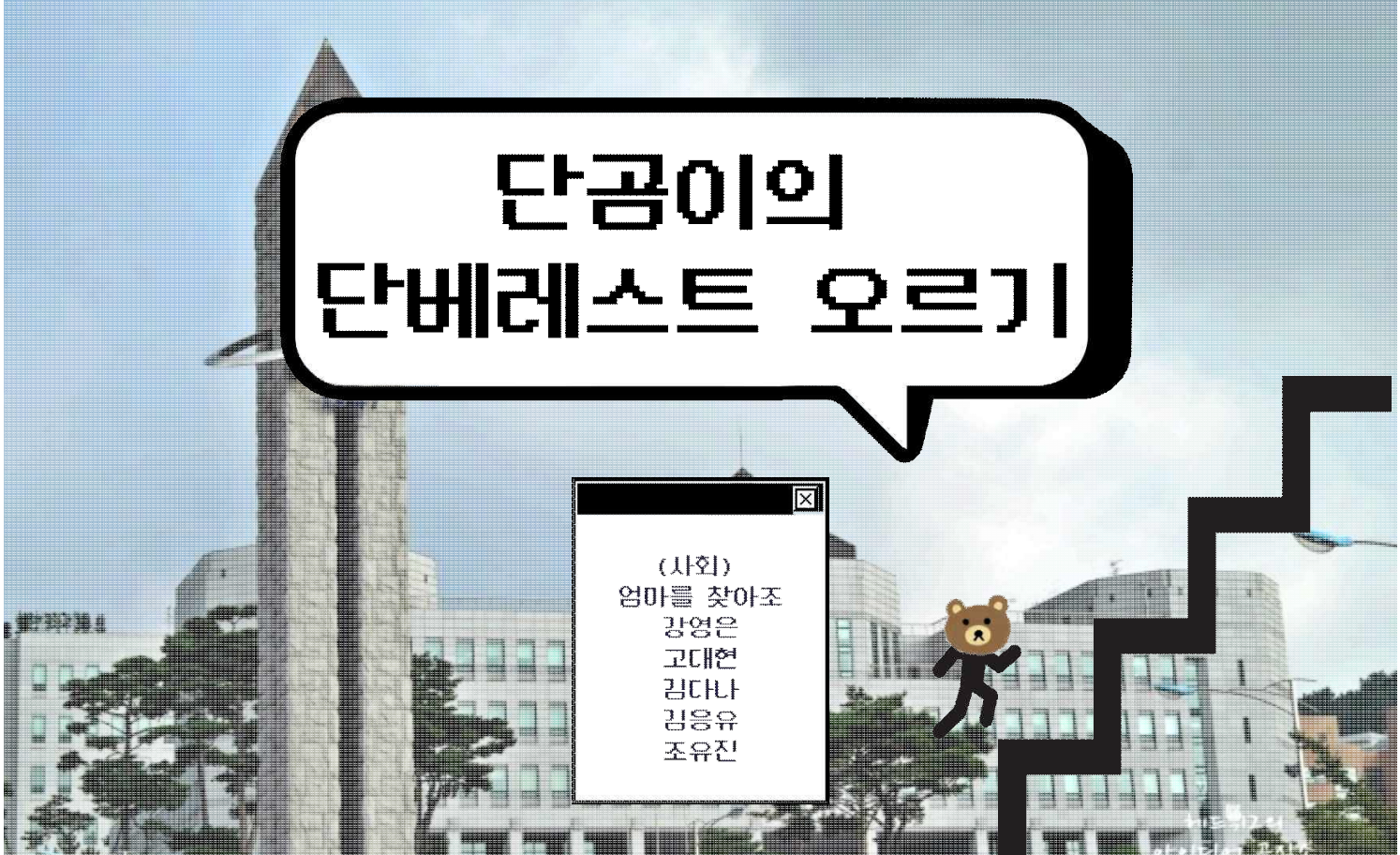

단곰이의
단베레스트 오르기
(사회)
엄마를 찾아조
강영은
고대현
김다나
김응유
조유진

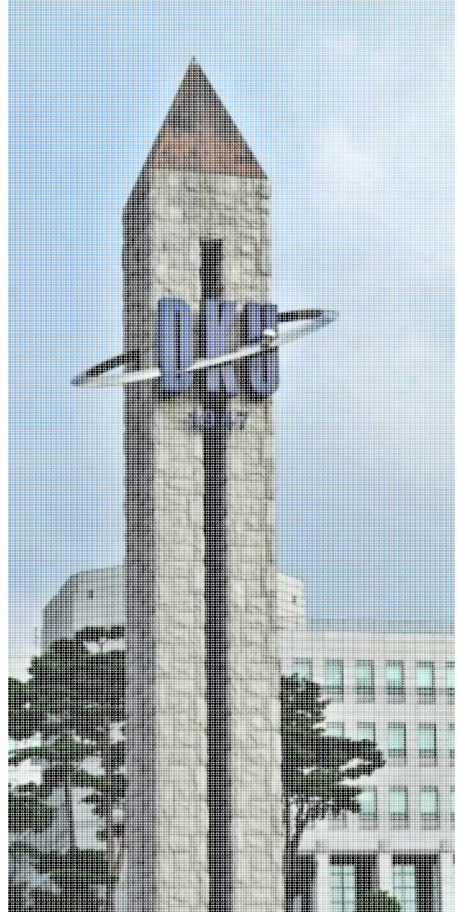
개요

01
게임 기획의도 및 게임 소개
- 스토리텔링

02
문제 정의 및 아이디어 구체화
- 게임 조건
- 구체화

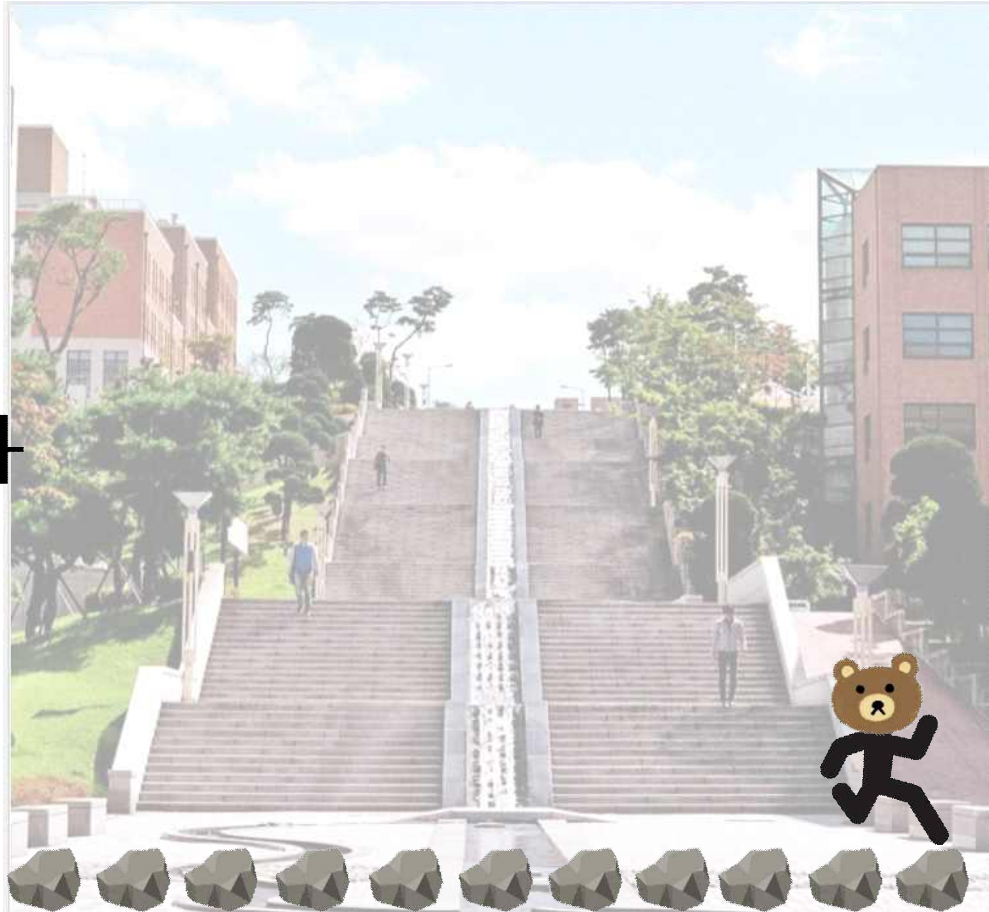
03
알고리즘 설계
- 마인드맵
- 순서도

04
코드 구현



01. 기획의도와
게임 소개

02. 문제 정의와
아이디어
구체화



<기획의도>

간단하면서도 재미요소가 있어서 **누구나 중독성 있게** 즐길 수 있는 게임을 제작하고 싶었습니다.

최근 드라마 '오징어 게임'이 단순한 게임을 소재로 활용하여 친숙함과 함께 세계적 흥행을 이뤄낸 만큼, 누구나 쉽게 참여할 수 있는 **'편한 게임'**을 만들고 싶었습니다.

<게임 소개>

윷놀이의 '**무작위의 숫자로 이동하고, 최종 목적지에 도착하면 승리한다.**'라는 게임 방식 차용

게임의 주 플레이어: 단국대학교 학생들

→ 우리 학교 학생들의 애교심을 고취시키고, 친숙하게 게임에 접근하도록 돕고 싶었기에 단국대학교의 건물들을 이어주고 있는 '학교 계단'을 오르는 스토리를 기획했습니다.

게임의 주 캐릭터: 단국대학교 상징 동물 '곰'

→ 귀여운 외모를 가진 곰, '단곰이'는 위험에 처해있는데, 이를 플레이어가 도와주어 목표를 달성한다는 스토리로 게임이 진행됩니다.

<기대 효과>

코로나 19로 신입생들이 2년 연속 캠퍼스를 경험하지 못하고 있고, 팀 구성원 대부분이 공감대를 형성하고 있었습니다.

따라서, 게임을 통해서 학교의 주요 건물들을 파악하고, '단베레스트'로 유명한 계단을 오르며 소속감과 애교심을 기를 수 있는 장을 마련하고 싶었습니다.

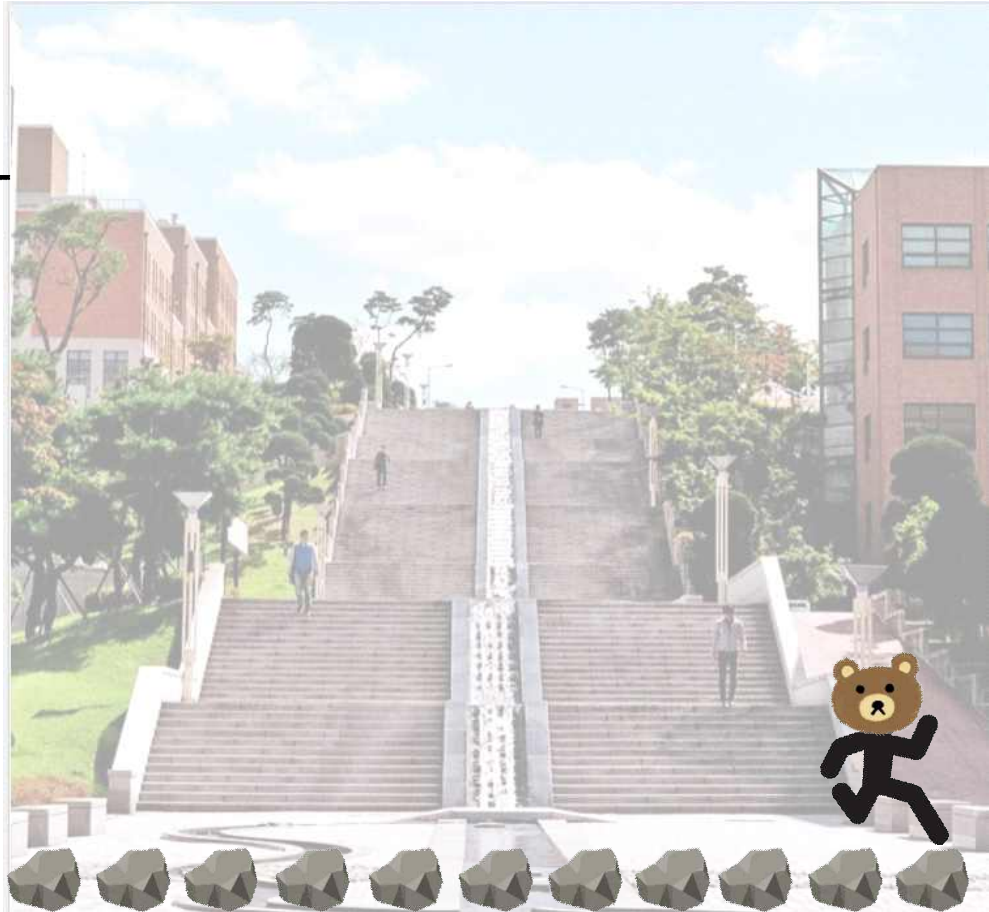
- 비대면 등교상황이 앞으로도 지속된다면, 신입생들이 이 게임을 통해 학교를 미리 경험할 수 있을 것입니다.
- 정상적인 상황으로 돌아가더라도, 합격생들이 입학을 기대하면서 학교를 경험하는 계기를 마련할 수 있다고 생각합니다.
- 졸업생, 휴학생도 학교에 대한 그리움을 게임으로 충족할 수 있을 것입니다.

storytelling

: 아기 단곰이가 엄마곰을 찾아 단국대학교 정문에서부터 곰상까지 이어지는 계단을 오르는 게임

단곰이는 어느날, 반드시 정확한 지점에 도착해야만 탈출할 수 있는 계단에 들어서게 되었다.
총 24개의 계단 중에는 단곰이를 막아서는 방해 계단들도 있다. 그 중 어떤 계단은 미니게임을 통해 더 많이 또는 더 적게 이동하게끔 하고, 어떤 계단은 아예 처음으로 돌아가게끔 한다.
단곰이는 무사히 이 계단을 올라 엄마곰을 찾을 수 있을까?

02. 문제 정의와 아이디어 구체화



02. 문제정의 및 아이디어 구체화

: 게임 조건

> 초기상태
게임 시작과 동시에 곰돌이는 1번 계단에 올라, 23개의 계단을 앞두고 있다.

> 목표상태
곰돌이는 정확히 24번 계단에 도착해야 한다.

>조건:
① 단곰이는 1번 계단에서 시작해 정확히 24번 계단에 도착해야 한다.
② 미니게임을 통해 뒤로 또는 앞으로 이동할 계단 수가 정해지고,
③ 이동할 칸이 0이거나 이동한 칸에 게임이 없을 때는 1~4 중 랜덤 숫자가 부여되는 <숫자뽑기>로 칸 이동이 가능하다.
④ 목표 달성 성공/실패 시 모두 게임을 종료하거나 재시도 할 수 있다.

02. 문제정의 및 아이디어 구체화

: 축소표현을 통한 게임아이디어 구체화

가위바위보

단곰 \ 게임	가위	바위	보
가위	비김. 0 이동	짐. -2 이동	이김. +2 이동
바위	이김. +2 이동	비김. 0 이동	짐. -2 이동
보	짐. -2 이동	이김. +2 이동	비김. 0 이동

홀짝

단곰 \ 게임	홀	짝
홀	이김. +1이동	짐. -1이동
짝	짐. -1이동	이김. +1이동

02. 문제정의 및 아이디어 구체화

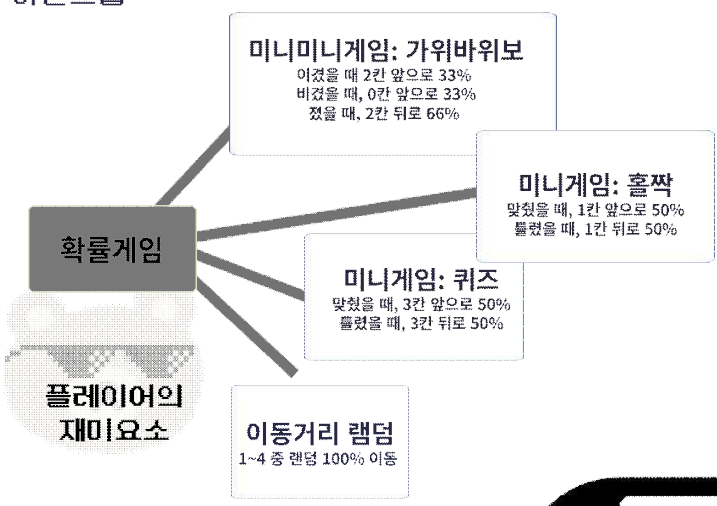
: 축소표현을 통한 게임아이디어 구체화

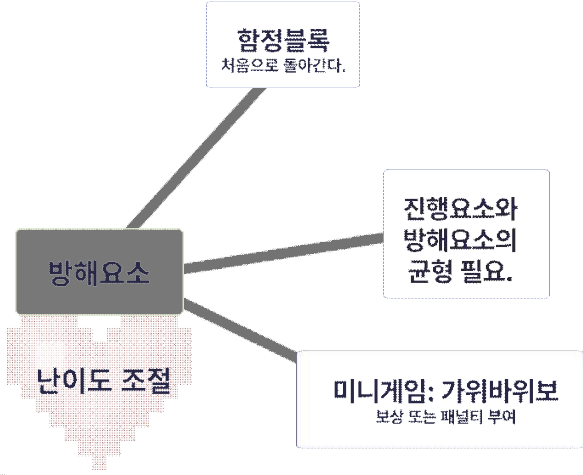
퀴즈 맞추기

단곰 \ 게임	1	2	3	4
1	이김.+3이동	짐.-3이동	짐.-3이동	짐.-3이동
2	짐.-3이동	이김.+3이동	짐.-3이동	짐.-3이동
3	짐.-3이동	짐.-3이동	이김.+3이동	짐.-3이동
4	짐.-3이동	짐.-3이동	짐.-3이동	이김.+3이동

03. 알고리즘 설계

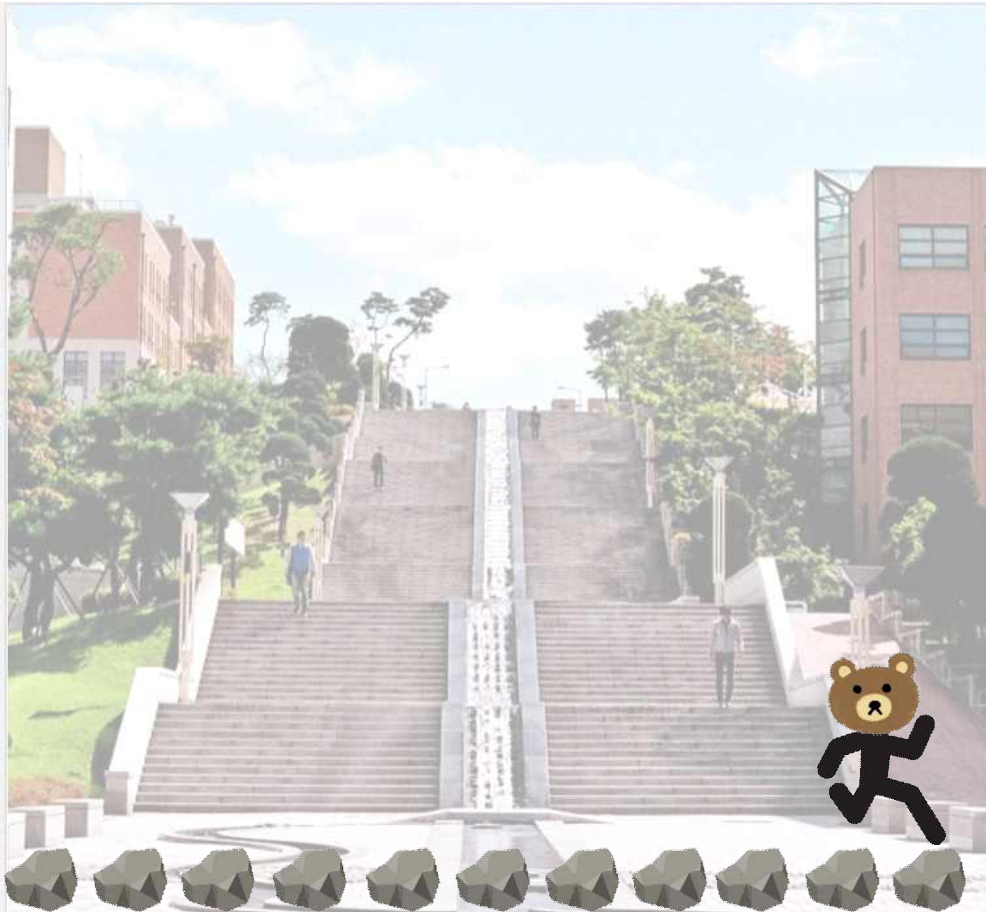
: 마인드맵





곰돌이의 징검다리 대탈출!

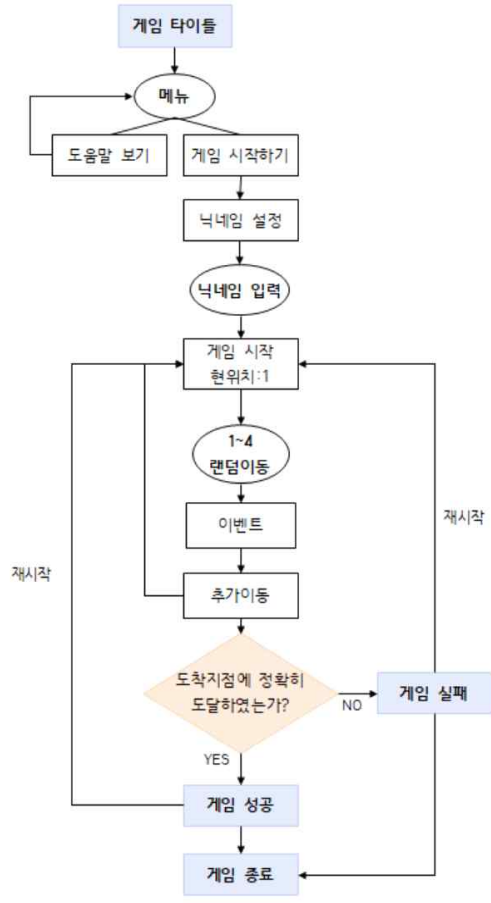
03. 알고리즘 설계



03. 알고리즘 설계

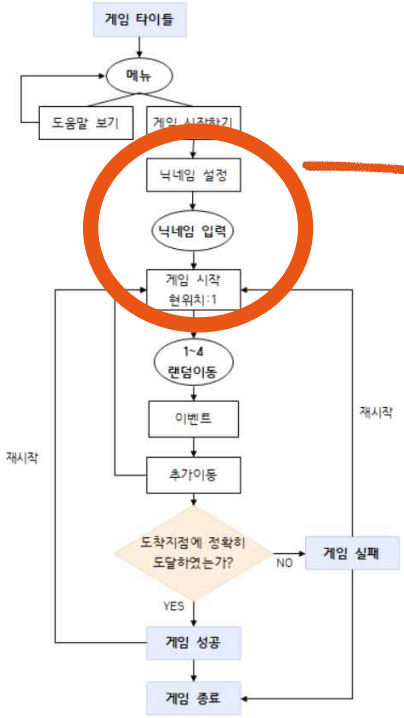
: 순서도

<순서도 - 게임 전체 흐름>

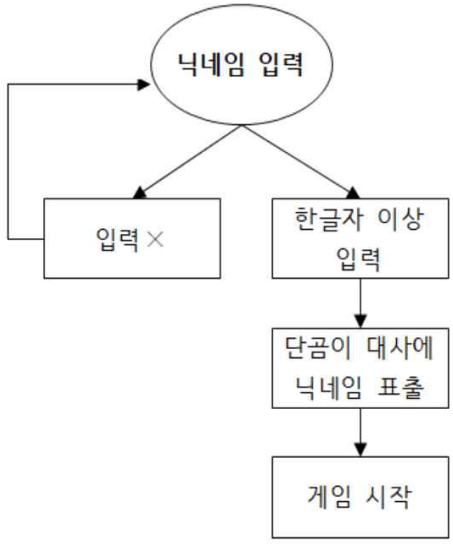


03. 알고리즘 설계

: 순서도

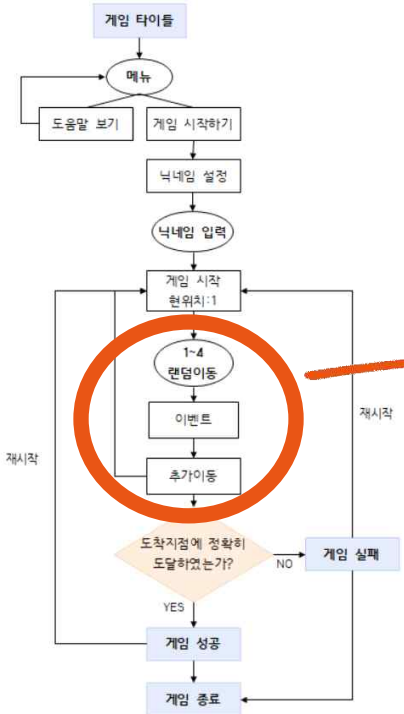


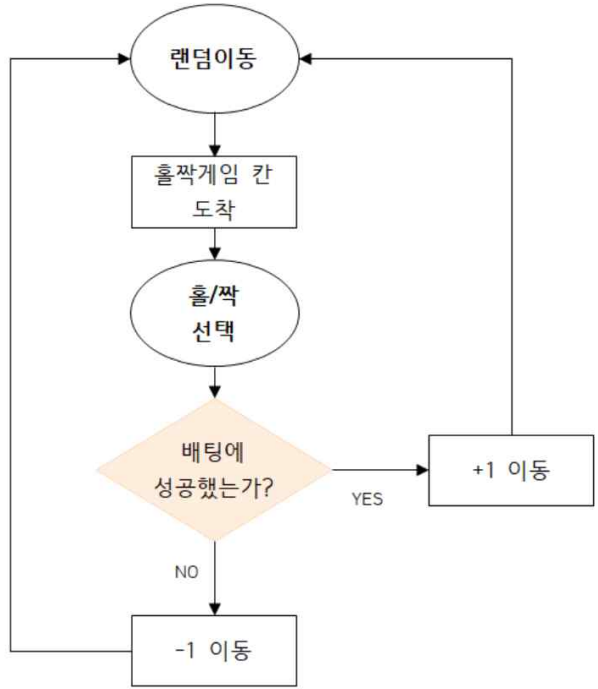
<순서도 - 닉네임 입력>



03. 알고리즘 설계

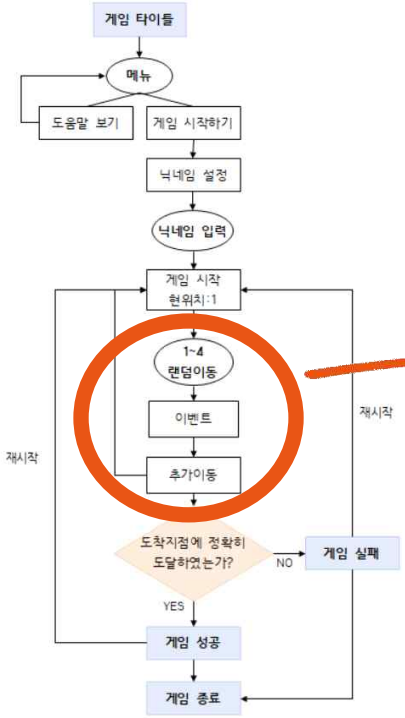
: 순서도



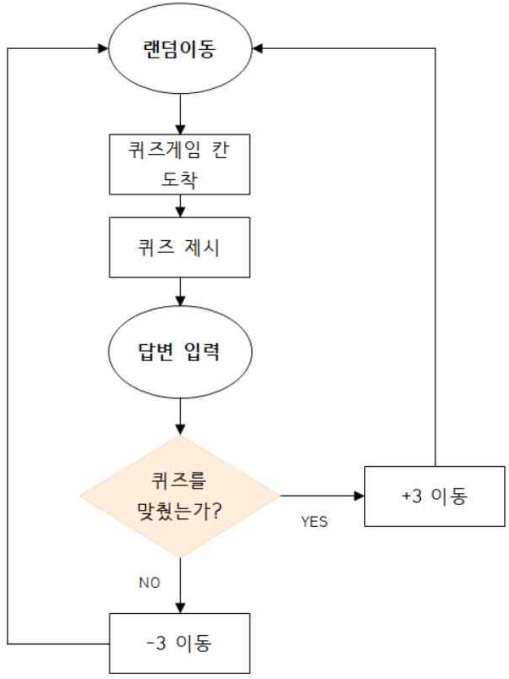


03. 알고리즘 설계

: 순서도

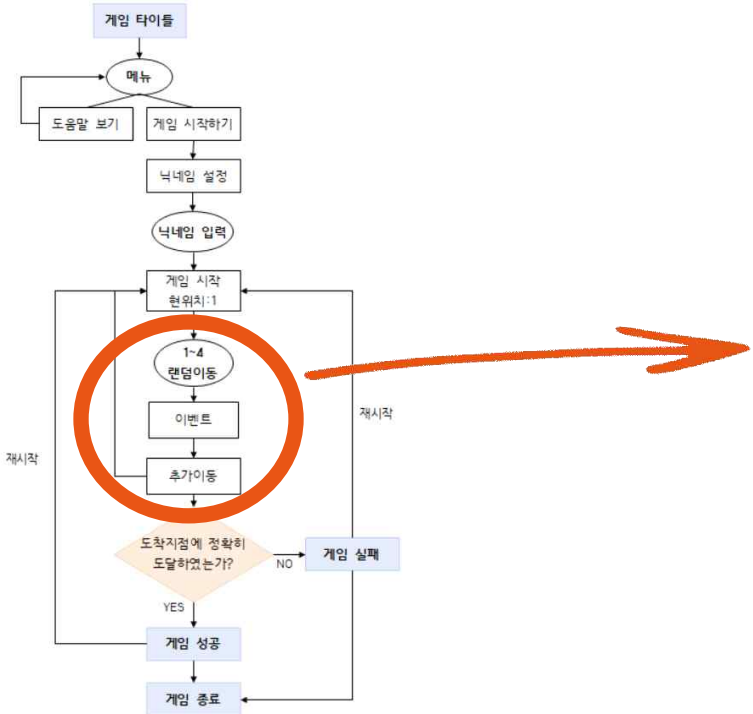


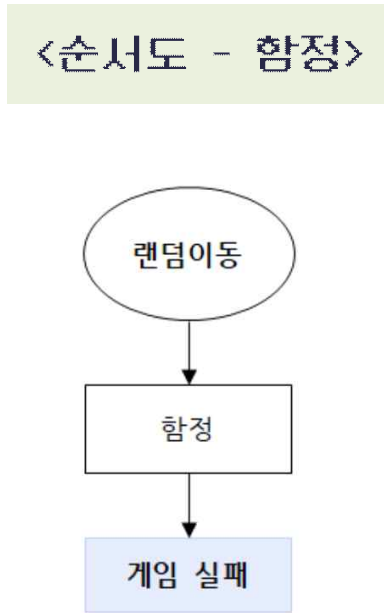
<순서도 - 퀴즈게임>



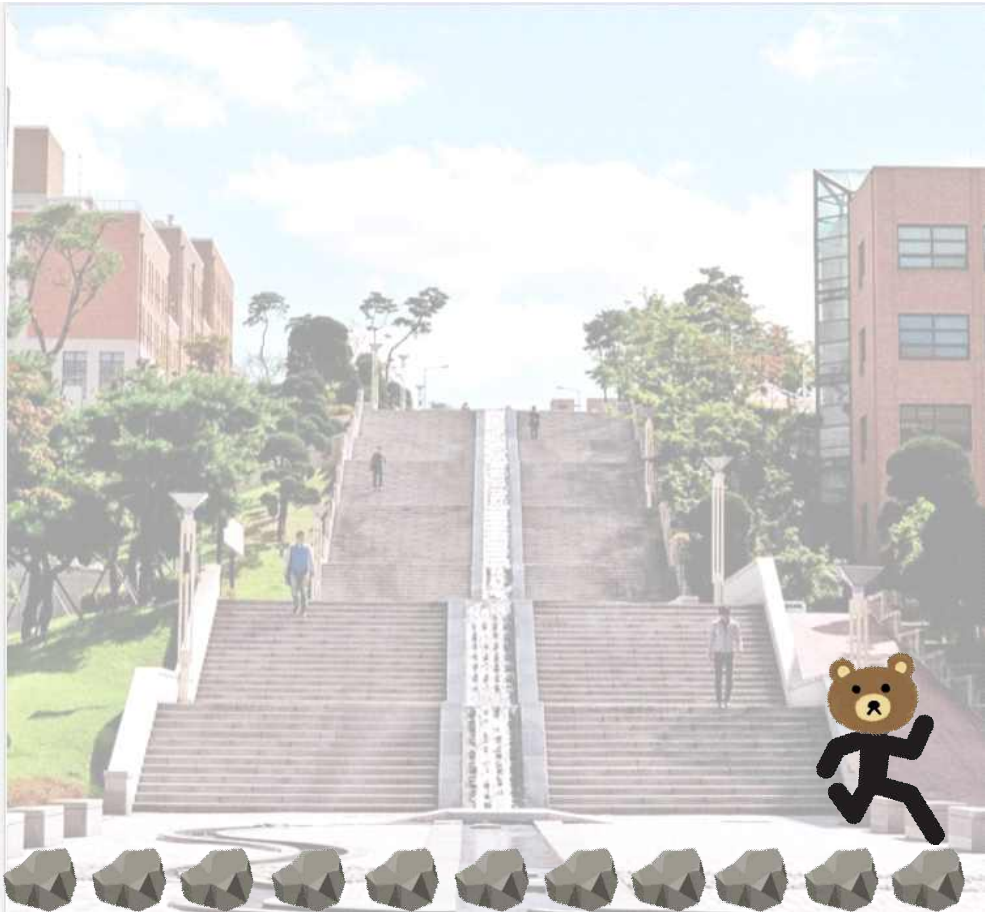
03. 알고리즘 설계

: 순서도





04. 코드 구현



04. 코드 구현

: 코드 입력 순서

#1. 초기화면
#2. 게임스토리 설명
#3. 게임방법 확인 및 닉네임 설정
#4. 게임 시작
#5. 이동하기
    #5-1. 8번째 칸의 <가위바위보 게임>
    #5-2. 11번째 칸의 <홀짝 게임>
    #5-3. 13번째 칸의 <함정>
    #5-4. 16번째 칸의 <퀴즈>
    #5-5. 20번째 칸의 <가위바위보> 코드
#6. 본게임 진행과정

#1. 초기 화면 코딩 구조 설명

>>> Print 함수

- 게임의 타이틀을 제시.

>>> input()/print 함수

- 사용자의 엔터키 입력 여부하에 게임을 시작할 수 있도록 설정

04. 코드 구현

#1. 초기 화면

```
#1.초기 화면
print ("                                    ")
print ("                                    ")
print ("     ·✧ *°♧   。    ₩n   ♧      · °*。·°₩n       ♧   ·。° *.  °✧ ")
print ("  ┃                                  ")
print ("  단                ┏━━━┓")
print ("   곰               ┗━━━┛  ")
print ("    이              ┗━━━┛  ")
print ("     의             ┗━━━┛  ")
print ("                    ┗━━━┛   ")
print ("        단          ┗━━━┛  ")
print ("         베         ┗━━━┛  ")
print ("          레        ┗━━━┛  ")
print ("           스       ┗━━━┛  ")
print ("            트      ┗━━━┛  ")
print ("                    ┗━━━┛   ")
print ("              오    ┗━━━┛ ")
print ("               르   ┗━━━┛ ")
print ("                기  ┗━━━₩n ┃ ")
print ("                G A M E")
print ("  ━━━━━━━━━━━━━━━━━━━₩n₩n")

print ("게임시작을 원하시면 엔터를 눌러주세요!")
input ()        #엔터를 입력하면 다음 코드가 진행됩니다.
print ("                      ₩n₩n")
```

#2. 게임스토리 코딩 구조 설명

>>> Print 함수

- 게임캐릭터인 단곰이가 스토리를 설명합니다.

04. 코드 구현

#2. 게임스토리 설명

```
#2.게임스토리 설명
print ("-------------------------------------------------")
print ("[단곰]-안녕! 나는 단곰이야!")
print('[단곰]-평화의 광장에 계시는 엄마를 찾으러 가야 하는데...₩n       곰상까지 가려면 이 단베레스트를 올라야 해!!₩n')
print('[단곰]-나를 도와줄 수 있겠니?₩n')
```

#3. 게임방법 확인 및 닉네임 설정 코딩 구조 설명

>>> 전역변수 global 설정
- 사용자가 설정한 name을 프로그램 가동 종료상황까지 사용하기 위함.

>>> 닉네임 설정 부분
- Input 함수로 사용자로부터 name을 입력받고 입력받은 글자 수에 따라 다른 결과 도출
- if문과 len(name) (1글자 이상 입력 = 게임 진행. 1글자 미만 입력 = 재질문)

>>> def print_menu 함수 정의
- print_menu = 안내문 출력과 함께 사용자로부터 입력 받은 숫자에 따라 다른 값을 도출하도록 설정.
- 사용자가 1번을 입력할 시 게임 방법 제시 후 print_menu 함수가 다시 출력됨.
- 사용자가 2번을 입력할 시 닉네임 짓는 화면으로 넘어갈 수 있도록 설정.

>>> Print_menu() 메뉴함수 호출
- 메뉴함수 결과에 따라 2번을 입력했을 때 nickname() 함수 호출 및 실행.

04. 코드 구현

#3. 게임방법 확인 및 닉네임 설정

```
#3. 게임방법 확인 및 닉네임 설정

    #닉네임 함수 정의

def nickname():

    global name   #닉네임으로 입력한 값을 함수 밖에서도 활용하기 위해 전역변수로 설정.
    name = input("₩n₩n----------""₩n[단공]- 너의 이름을 알고싶어!""₩n₩n단공이에게 닉네임을 알려주세요.")


    if len(name) < 1:       #닉네임란에 아무것도 입력하지 않았을 경우
        print("한 글자 이상 입력해주세요.")
        return False        #False로 재진행.

    else:
        print("₩n[단공]- 잘 부탁해!" + name,'₩n')    #닉네임란에 입력했을 경우, 단공이의 대사에 닉네임 노출됨.
        return True
```

#4. 게임 시작 코딩 구조 설명

>>> Print 함수

-게임 목표 제시. 플레이어가 다시 게임 진행 방식을 확인하게 함.
-#3에서 전역변수로 설정한 name이 활용됨. 플레이어의 몰입감 형성.
-계단 지도를 대략적으로 보여줌. 현재 위치 (1번 계단) 재확인.

#4. 게임 시작

```
#4. 게임 시작

print ('₩n₩n•──────────────────────────────────────────•° •❀•° •──────────────────────────────────────────•₩n')
print ('                    M  I  S  S  I  O  N                    ₩n')
print ('    ',name,', 단곰이가 단베레스트를 오르는 것을 도와주세요!')    #닉네임 함수에서 설정했던 전역변수 name이 활용됩니다.
print ('       계단은 총 24개. ₩n       8, 11, 16, 20번째 계단에서 이벤트가 발생합니다.')
print ('       13번째 계단은 함정이니 조심하세요!')
print (' ₩n₩n        <목표>: 곰상이 있는 24번째 계단에 정확히 도착하자!₩n')
print ('•──────────────────────────────────────────•° •❀•° •──────────────────────────────────────────•₩n₩n₩n₩n')


print ('₩n    ★ S T A I R S - M A P ★')
print ('₩n    [1] [2] [3] [4] [5] [6] [7] [범] [9] [10] [사] [12] [폭] [14] [15] [혜] [17] [18] [19] [도] [21] [22] [23] [24] [?]')


print (' ₩n₩n현재위치: [★] [2] [3] [4] [5] [6] [7] [범] [9] [10] [사] [12] [폭] [14] [15] [혜] [17] [18] [19] [도] [21] [22] [23] [24] [?]')
```

#5. 이동하기 코딩 구조 설명

>>> 반복문 while 설정
- 전체 반복문은 단곰이가 24칸에 도달할 때까지 진행됨.

>>> if 조건문 사용
- 단곰이가 24칸에 있으면 사용자에게 메시지 표시 후 게임 종료됨.

>>> time모듈과 def timer 함수 정의
- time모듈을 사용하여 15초 후 게임이 자동으로 종료됨.
- Timer 함수에 따라 15초가 흘러가는 것을 눈으로 볼 수 있음.

04. 코드 구현

#5. 이동하기 코드

```
#5. 이동하기 코드
import random


p = 1                       #현재위치 초기값인  1을  변수 p에 저장합니다.

while p <= 24:              #while문으로 24에 도달할때까지 반복합니다.
    print('-----------------------------₩n현재위치:', p, '번째 계단')   #이동할때마다 현재위치를 반복적으로 알려줍니다.

    if p == 24:             #정확히 24에 도달했을때의 조건문

        print (' ₩n현재위치: [1] [2] [3] [4] [5] [6] [7] [범] [9] [10] [사] [12] [목] [14] [15] [혜] [17] [18] [19] [도] [21] [22] [23] [★] [?]₩n
        print ("₩n [단곰]- 우와! 이곳이 「평화의 광장」 이구나! ")
        print(' [단곰]- 평화의 광장 중앙에 곰상이 있네!!')
        print(' [단곰]- 엄마를 찾았다!! ₩n')
        print('.    +*.o  ·*·:*:o*+o *o:·+')

        print('-------------------------------')
        print(" 단곰이는 성공적으로 곰상에 도착했고")
        print(" 엄마를 찾았습니다!")
        print('₩n [단곰]- 엄마랑 다시 만나게 해줘서 고마워 ㅎㅎ ')
        print('₩n₩n•-----------------------------------• S U C C E S S •-----------------------------------•')
        print('₩n ↳ 축하합니다.''₩n ↳ 게임이 15초 후 자동으로 종료됩니다.')  #게임이 성공적으로 종료됩니다.


        #시간차를 두고 종료하기 위해 time 모듈 생성
        import time

        start = 15      #15초 후 종료

        def timer(t):
            while t>0:
                print('▶', t,'초', end='₩r ')  #덮어쓰기
                t = t-1
                time.sleep(1)

        timer(start)
        break
```

#5-1. 미니게임 <가위바위보 게임> 코딩 구조 설명

>>> if 조건문 사용
-단곰이가 8칸에 있으면 범정관을 배경으로 가위바위보 게임이 진행됨.

>>> Print 함수
-현재 위치인 8번째 칸을 재명시해줌.

>>> result 함수
-가위바위보 결과에 따라 문구를 달리함.

>>> random 모듈 사용
-가위바위보 진행을 위함.

>>> 가위바위보 결과값을 새로운 p 변수로 설정.

>>> 가위바위보 소스 - def checkwin 정의, checkwin 함수 실행
-가위바위보에서 승리, 패배, 무승부에 따른 추가이동을 달리 하여 이동결과에 반영함.

04. 코드 구현

#5-1. 미니게임 <가위바위보 게임> 코드

```
#5-1. 미니게임 <가위바위보> 코드
    if p == 8:          #8번째 계단에 도착하면 범정관을 배경으로 가위바위보 게임이 진행됩니다.
        print('-----------------------------------')
        print("₩n 『범정관』에 도착했습니다.₩n 이벤트가 진행됩니다.")
        print('  [단곰]- 범정관은 대학 본부가 있는 곳이래~₩n')  #단곰이의 대사로 범정관에 대해 소개합니다.
        print(' 미니게임 <가위바위보>를 시작합니다.')



        import random

        # 가위, 바위, 보를 리스트에 올리기
        sel = ['가위', '바위', '보']
        # 가위바위보 결과 문구
        result = {0: ' 가위바위보에서 <승리>했습니다. ₩n 단곰이가 <두 칸 앞으로> 이동합니다.',
          1: ' 가위바위보에서 <패배>했습니다. ₩n 단곰이가 <두 칸 뒤로> 이동합니다.',
          2: ' 가위바위보에서 <비겼>습니다. ₩n 단곰이는 이동하지 않습니다.'}
```

04. 코드 구현

#5-1. 미니게임 <가위바위보 게임> 코드

```
# 가위바위보 소스
def checkWin(user, com):
    global p                #가위바위보 결과값을 저장할 p를 전역변수로 설정합니다.
    if not user in sel:
        print('잘못 입력하였습니다. 다시 입력하세요.') # 잘못 입력한 경우
        return False

    p = 8                   #현재 위치는 8임을 새로운 함수 내에서 변수로 정의.


    print(f'₩n[사용자 ( {user} vs {com} ) 게임]')# 가위바위보 경우의 수
    if user == com:                              # 가위바위보를 비길 경우
        state = 2
        print(result[state])
        move = p + 0                            #추가이동을 설명할 변수 move
        print('₩n----------------------------₩n현재위치:', move, '번째 계단')  #추가이동 후 위치
    elif user == '가위' and com == '바위':       # 가위바위보를 패배할 경우
        state = 1
        print(result[state])
        move = p - 2                            #추가이동을 설명할 변수 move
        print('₩n----------------------------₩n현재위치:', move, '번째 계단')  #추가이동 후 위치
    elif user == '바위' and com == '보':
        state = 1
        print(result[state])
        move = p - 2                            #추가이동을 설명할 변수 move
        print('₩n----------------------------₩n현재위치:', move, '번째 계단')   #추가이동 후 위치
    elif user == '보' and com == '가위':
        state = 1
        print(result[state])
        move = p - 2                           #추가이동을 설명할 변수 move
        print('₩n----------------------------₩n현재위치:', move, '번째 계단')  #추가이동 후 위치
    else:                                       # 가위바위보의 나머지 경우(이길 경우)
        state = 0
        print(result[state])
        move = p + 2                           #추가이동을 설명할 변수 move
        print('₩n----------------------------₩n현재위치:', move, '번째 계단')  #추가이동 후 위치

    p = move                                  #전역변수 p에 이동한 결과값이 저장되어 본게임 이동 결과에 반영됩니다.

    return True
```

04. 코드 구현

#5-1. 미니게임 <가위바위보 게임> 코드

```
while True:                                          # 가위바위보 과정 문구
    user = input("₩n 가위, 바위, 보 중 선택해주세요: ")     # 가위, 바위, 보 중 무엇을 낼 것인지 물어보기
    com = sel[random.randint(0, 2)]                  # 가위, 바위, 보 중 랜덤으로 설정
    if checkWin(user, com):
        break
```

#5-2. 미니게임 <홀짝 게임> 코딩 구조 설명

>>> if 조건문 사용
-단곰이가 11번째 계단에 도착하면 홀짝 게임이 진행됨.

>>> Print 함수
-현재 위치인 11번째 칸을 재명시해줌.

>>> result 함수
-홀짝 결과에 따라 문구를 달리함.

>>> def holzzack 함수 정의 및 실행
-11번째 칸에서 진행되는 홀짝게임을 모든 경우의 수를 고려하여 함수로 정의. 각각의 결과값은 result에 따라 출력.

>>> if - else 문 사용
-사용자가 결과값을 틀렸을 경우 한 칸 뒤로, 맞았을 경우 한 칸 앞으로 이동함.

04. 코드 구현

#5-2. 미니게임 <홀짝 게임> 코드

```
#5-2. 미니게임 <홀짝 맞추기> 코드

    if p == 11:          #11번째 계단에 도착하면 사회과학관을 배경으로 홀짝 게임이 진행됩니다.
        print (' ₩n현재위치: [1] [2] [3] [4] [5] [6] [7] [범] [9] [10] [★] [12] [폭] [14] [15] [해] [17] [18] [19] [도] [21] [22] [23] [24] [?]₩r

        print("₩n ┌──────────────────────────────────────────────────────┐ ")
        print(" │ 「사회과학관」에 도착했습니다. │")
        print(" │ 이벤트가 진행됩니다.          │")
        print(" └──────────────────────────────────────────────────────┘ ")


        print(' [단곰]- 사회과학관은 사회과학대학 소속 학생들이 공부하는 곳이야!')  #단곰이의 대사로 사회과학관에 대해 소개합니다.
        print(' [단곰]- 정치외교학과, 행정학과, 도시계획부동산학부가 주로 사용해!₩n')
        print('↳ 미니게임 <홀짝 맞추기>를 시작합니다.')
        print('₩n▸ 게임은 2~30중 하나의 숫자를 제시합니다.₩n▸ 게임이 고른 숫자가 <홀>일지, <짝>일지 맞춰주세요!')


        import random

        #홀짝 결과 문구
        result = {0: ' 홀짝 맞추기에서 <승리>했습니다. ₩n₩n단곰이가 <한 칸 앞으로> 이동합니다.',
                  1: ' 홀짝 맞추기에서 <패배>했습니다. ₩n₩n단곰이가 <한 칸 뒤로> 이동합니다.'}
```

04. 코드 구현

#5-2. 미니게임 <홀짝 게임> 코드

```
#홀짝 함수 설정
def holzzack():
    global p

    # 홀짝 결과 화면 보여주기
    N = random.randint(2,30)                    # 2에서 30까지 랜덤으로 숫자가 나옴.
    user = int(input("₩n  [홀(0), 짝(1)]₩n 숫자를 입력해주세요 :"))     # 유저가 홀인지 짝인지 선택(맞추기).
    print("₩n 정답 : {}".format(N), end=' / ')    # 정답은 몇인지, 정답 유무를 알려줌.

    p = 11     #현재 위치가 11임을 함수 내에 다시 명시함.

    if user:
        if N % 2:                                   #나머지 값에 따라 홀짝을 판명합니다.
            print("틀렸습니다.")
            state = 1                               #앞서 설정한 결과문구를 출력합니다.
            print(result[state])
            move = p - 1                            #추가이동을 설명할 변수 move
            print('₩n----------------------------------₩n현재위치:', move, '번째 계단')
        else:
            print("맞았습니다.")
            state = 0
            print(result[state])
            move = p + 1                            #추가이동을 설명할 변수 move
            print('₩n----------------------------------₩n현재위치:', move, '번째 계단')

    else:
        if N % 2:                                   #나머지 값에 따라 홀짝을 판명합니다.
            print("맞았습니다.")
            state = 0                               #앞서 설정한 결과문구를 출력합니다.
            print(result[state])
            move = p + 1                            #추가이동을 설명할 변수 move
            print('₩n----------------------------------₩n현재위치:', move, '번째 계단')
        else:
            print("틀렸습니다.")
            state = 1                               #앞서 설정한 결과문구를 출력합니다.
            print(result[state])
            move = p - 1                            #추가이동을 설명할 변수 move
            print('₩n----------------------------------₩n현재위치:', move, '번째 계단')

    p = move      #추가이동으로 move에 저장된 값이 p에 새로 저장됩니다.
```

04. 코드 구현

#5-2. 미니게임 <홀짝 게임> 코드

```
#홀짝 함수 호출
holzzack()
```

#5-3. 13번 째 칸의 <함정> 코딩 구조 설명

>>> Print 함수
-현재 위치인 13번째 칸을 재명시해줌.
>>> if → break 함수 설정.
-13번째 칸에 도달하면 break로 인하여 while 반복문 종료. (GAME OVER)
-print 함수로 장소 설명, 게임 진행 대사 제시.
>>> time모듈과 def timer 함수 정의
-time모듈을 사용하여 10초 후 게임이 자동으로 종료됨.
-Timer 함수에 따라 10초가 흘러가는 것을 눈으로 볼 수 있음.

04. 코드 구현

#5-3. 13번 째 칸의 <함정> 코드

```
#5-3. 13번째 칸에서 발동되는 <함정> 코드

    if p == 13:         #13번째 칸에 도착하면 폭포공원을 배경으로 함정이 발동됩니다.
        print (' Wn현재위치: [1] [2] [3] [4] [5] [6] [7] [범] [9] [10] [사] [12] [★] [14] [15] [해] [17] [18] [19] [도] [21] [22] [23] [24] [?]Wn

        print('-------------------------------')
        print("Wn 어랏? 단곰이가『폭포공원』에 도착했습니다.Wn 저런, 함정에 걸려 나오지 못하겠군요..")
        print(' [단곰]- 우와~ 폭포공원이 너무 예쁘다~~ ')
        print(' 단곰이는 폭포공원 정자 위에서 잠들고 말았습니다.')
        print('WnWn•————————————————————————• G A M E O V E R •————————————————————————•')
        print(' 안타깝게도 단곰이는 엄마를 찾지 못했습니다.''Wn 게임이 10초 후 자동으로 종료됩니다.')


        #시간차를 두고 종료하기 위해 time 모듈 생성
        import time

        start = 10      #10초 후 종료

        def timer(t):
            while t>0:
                print('▶', t,'초', end='Wr ')  #덮어쓰기
                t = t-1
                time.sleep(1)

        timer(start)
        break
```

#5-4. 16번 째 칸의 <퀴즈> 코딩 구조 설명

>>> Print 함수
-현재 위치인 16번째 칸을 재명시해줌.
>>>Word 변수에 퀴즈 선다 리스트 설정.
-플레이어는 이 중에서 퀴즈 정답을 고를 수 있음.
>>>Result 변수 설정.
-0 또는 1에 퀴즈의 결과에 따른 게임 설명 설정.

#5-4. 16번 째 칸의 <퀴즈> 코딩 구조 설명

>>> def quiz(user) 함수 정의
-global p
  미니게임에서 결정된 새로운 변수 p를 본게임에 반영하기 위해 전역변수로 설정.
-p = 16
  현재 위치는 16임을 새로운 함수 내에서 변수로 정의.
-if not user in word 사용
  Result에서 저장한 선다리스트에 없는 답을 입력한 경우 재입력 유도.
-if – else문 사용
  정답을 정확히 입력했을 경우 추가 전진 이동, 정답이 아닌 것을 입력했을 경우 추가 후진 이동.
  If – else문 내에서 result의 state를 설정하고, move 변수를 사용해서 플레이어에게 이동된 위치설명.
>>> While True문 사용
-print로 문제 및 선다 제시
-사용자가 입력한 정답을 input으로 확인
-quiz(user)함수 실행. 실행 결과에 따라 break로 while 반복문 종료.

04. 코드 구현

#5-4. 16번 째 칸의 <퀴즈> 코드

```
#5-4. 16번째 칸에서 진행되는 <퀴즈> 코드
    if p == 16:          #16번째 계단에 도착하면 해당관을 배경으로 퀴즈게임이 진행됩니다.
        print (' ₩n현재위치: [1] [2] [3] [4] [5] [6] [7] [범] [9] [10] [사] [12] [폭] [14] [15] [★] [17] [18] [19] [도] [21] [22] [23] [24]
        print("₩n ┌──────────────────────────────────────────┐ ")
        print(" │ 『해당관』에 도착했습니다.  │")
        print(" │ 이벤트가 진행됩니다.       │")
        print(" └──────────────────────────────────────────┘₩n")

        print(' [단곰]- 해당관은 학생들의 건물, 학생회관이야!')  #단곰이의 대사로 해당관에 대해 소개합니다.
        print(' [단곰]- 동아리도 하고, 학식도 먹고, 전공책도 살 수 있어~₩n')

        print(' ↳ 미니게임 <단국대학교 퀴즈>를 시작합니다.')


        # 퀴즈 선다 리스트 만들기
        word = [ '백범 김구', '범정 장형', '해당 조희재', '우남 이승만']

        # 퀴즈 결과 문구
        result = {0: '₩n 정답입니다. 단곰이가 <세 칸 앞으로> 이동합니다.',
                  1: '₩n 오답입니다. 단곰이가 <세 칸 뒤로> 이동합니다.',}
```

04. 코드 구현

#5-4. 16번 째 칸의 <퀴즈> 코드

```
# 퀴즈 함수 설정
def quiz(user):
    global p  #미니게임에서 결정된 새로운 변수 p를 본게임에 반영하기 위해 전역변수로 설정합니다.

    if not user in word:
        print('잘못 입력하였습니다. 다시 입력하세요.')
        return False                                #선다리스트에 없는 코드를 입력한 경우 다시 입력할 수 있습니다.

    if user == '범정 장형':                 #정답을 정확히 입력했을때의 경우
        state = 0                           #미리 설정한 result의 상태 0이 된다.
        print(result[state])                #result의 내용을 출력한다.
        move = p + 3                              #추가이동을 설명할 변수 move
        print('₩n------------------------------₩n현재위치:', move, '번째 계단')  #추가이동 후 위치


    else:                                  #정답 외의 것들을 입력했을 때의 경우
        state = 1                          #미리 설정한 result의 상태 1이 된다.
        print(result[state])               #result의 내용을 출력한다.
        move = p - 3                              #추가이동을 설명할 변수 move
        print('₩n------------------------------₩n현재위치:', move, '번째 계단')  #추가이동 후 위치


    p = move            #global p로 설정되어 본게임 이동 결과에 반영됩니다

    return True


while True:             #현재 위치가 16이 되었을때, 문제를 제출하고 정답을 받습니다.
    print("₩n○문제: 단국대학교의 설립자는 누구인가요?")
    print(word)
    user = input("○정답: ")
    if quiz(user):
        break           #While True문을 종료합니다.
```

#5-5. 20번 째 칸의 <가위바위보> 코딩 구조 설명

>>> print 함수
-도서관에 도착했다는 문구와 도서관 소개와 가위바위보 시작 문구 보여줌.
-현재 위치인 20번째 칸을 재명시해줌.
>>> result
-가위바위보 결과
>>> 가위바위보 소스
-global p
　가위바위보 결과값을 새로운 p 전역 변수로 설정해서 본게임에 적용함.
-if-else문
　승리, 패배, 비김에 따라 추가 이동함.

04. 코드 구현

#5-5. 20번 째 칸의 <가위바위보> 코드

```
#5-5. 20번째 칸에서 진행되는 <가위바위보> 코드

    if p == 20:    #20번째 계단에 도착하면 도서관을 배경으로 가위바위보 게임이 진행됩니다.
        print (' ₩n현재위치: [1] [2] [3] [4] [5] [6] [7] [범] [9] [10] [사] [12] [목] [14] [15] [해] [17] [18] [19] [★] [21] [22] [23] [24] [?]₩
        print("₩n ┌──────────────────────────────────────┐")
        print(" │ 『도서관』에 도착했습니다.  │")
        print(" │ 이벤트가 진행됩니다.       │")
        print(" └──────────────────────────────────────┘₩n")
        print(' [단곰]- 도서관이 정말 크지? 책도 정말 많을 것 같아~')
        print(' [단곰]- 멋진 도산라운지에서 친구들과 공부하고, 스터디룸도 빌릴 수 있대~')
        print(' [단곰]- 다음에 나랑 공부하러 가자!₩n')  #단곰이의 대사로 도서관에 대해 소개합니다.
        print(' ↳ 미니게임 <가위바위보>를 시작합니다.')

        import random

        # 가위, 바위, 보를 리스트에 올리기
        sel = ['가위', '바위', '보']
        # 가위바위보 결과 문구
        result = {0: '  ↳ 가위바위보에서 <승리>했습니다. ₩n  ↳ 단곰이가 <두 칸 앞으로> 이동합니다.',
          1: '  ↳ 가위바위보에서 <패배>했습니다. ₩n  ↳ 단곰이가 <두 칸 뒤로> 이동합니다.',
          2: '  ↳ 가위바위보에서 <비겼>습니다. ₩n  ↳ 단곰이는 이동하지 않습니다.'}
```

04. 코드 구현

#5-5. 20번 째 칸의 <가위바위보> 코드

```
# 가위바위보 소스
def checkWin(user, com):
    global p                                    #가위바위보 결과값을 새로운 p 전역변수로 설정해서 본게임에 적용합니다.
    if not user in sel:
        print('잘못 입력하였습니다. 다시 입력하세요.') # 잘못 입력한 경우
        return False

    p = 20    #현재 위치는 20임을 새로운 함수 내에서 변수로 정의.

    print(f'₩n[사용자 ( {user} vs {com} ) 게임]') # 가위바위보 경우의 수
    if user == com:                              # 가위바위보를 비길 경우
        state = 2
        print(result[state])
        move = p + 0                            #추가이동을 설명할 변수 move
        print('₩n-------------------------------₩n현재위치:', move, '번째 계단')  #추가이동 후 위치
    elif user == '가위' and com == '바위':       # 가위바위보를 패배할 경우
        state = 1
        print(result[state])
        move = p - 2                            #추가이동을 설명할 변수 move
        print('₩n-------------------------------₩n현재위치:', move, '번째 계단')  #추가이동 후 위치
    elif user == '바위' and com == '보':
        state = 1
        print(result[state])
        move = p - 2                            #추가이동을 설명할 변수 move
        print('₩n-------------------------------₩n현재위치:', move, '번째 계단')   #추가이동 후 위치
    elif user == '보' and com == '가위':
        state = 1
        print(result[state])
        move = p - 2                           #추가이동을 설명할 변수 move
        print('₩n-------------------------------₩n현재위치:', move, '번째 계단')  #추가이동 후 위치
    else:                                       # 가위바위보의 나머지 경우(이길 경우)
        state = 0
        print(result[state])
        move = p + 2                           #추가이동을 설명할 변수 move
        print('₩n현재위치:', move, '번째 계단')  #추가이동 후 위치

    p = move                                #global p로 설정되어 본게임 이동 결과에 반영됩니다.

    return True
```

04. 코드 구현

#5-5. 20번 째 칸의 <가위바위보> 코드

```
        while True:                                     # 가위바위보 과정 문구
            user = input("₩n 가위, 바위, 보 중 선택해주세요: ")        # 가위, 바위, 보 중 무엇을 낼 것인지 물어보기
            com = sel[random.randint(0, 2)]             # 가위, 바위, 보 중 랜덤으로 설정
            if checkWin(user, com):
                break                                   #While True문을 종료합니다.
```

#6. 본게임 진행 과정 코딩 구조 설명

>>> input()
- 사용자가 입력한 값에 따라 코드 진행

>>> random.randint
- 앞에서 호출한 random모듈로 1~4중 랜덤의 숫자를 뽑아 n에 저장함.

>>> if문
- 플레이어의 위치가 24를 초과할 경우 게임 오버 메세지 출력.
- print 함수로 현재 24초과 명시.

>>> time모듈과 def timer 함수 정의
time모듈을 사용하여 10초 후 게임이 자동으로 종료됨.
Timer 함수에 따라 10초가 흘러가는 것을 눈으로 볼 수 있음.

#6. 본게임 진행 과정

```
#6. 본게임 진행 과정

    print('₩n▶(엔터)를 눌러 숫자카드를 뽑습니다.')
    input()                                  #사용자가 엔터를 누르면 다음 코드가 진행됩니다.

    n = random.randint(1, 4)                 #변수 n에 1~4 중 랜덤의 숫자가 지정됩니다.
    print('단곰이는', '<', n, '>', "칸 이동합니다.₩n₩n") #랜덤 숫자를 사용자에게 알려줍니다.
    p = p + n                                #5단계에서 p(현재위치)는 1로 설정했었고, 랜덤의 숫자 n이 이에 더해지는 값이 새로운 p로 지정됩니다

    if p > 24:                               #게임의 결과에 p(현재위치)가 24를 초과하면 게임이 오버됩니다.
        print (' ₩n현재위치: [1] [2] [3] [4] [5] [6] [7] [범] [9] [10] [사] [12] [폭] [14] [15] [혜] [17] [18] [19] [도] [21] [22] [23] [24] [★]₩

        print('-------------------------------')
        print('┌─────────┐')
        print('┐┐┐┐┐┐  ')
        print('┐┐┐┐┐┐  ')
        print('┐┐┐┐┐┐  ┌─────────┐')
        print('┐┐┐┐┐┐  │         │')
        print('┐┐┐┐┐┐  │  □ □ □  │')
        print('┐┐┐┐┐┐  │  ┌┐┌┐┌┐  │')

        print("₩n 어랏? 단곰이가 『보정동 카페거리』에 도착했습니다.₩n 저런, 엄마가 있는 곳상에 도착하지 못했군요.")
        print(' [단곰]- 여긴 어디지..? 엄마는 어디계신거지? ')
        print(' 단곰이는 길을 잃었습니다.')
        print('₩n₩n•————————————————————————• G A M E O V E R •————————————————————————•')
        print(' 안타깝게도 단곰이는 엄마를 찾지 못했습니다.''₩n 게임이 10초 후 자동으로 종료됩니다.')
```

#6. 본게임 진행 과정

```
#시간차를 두고 종료하기 위해 time 모듈 생성
import time

start = 10      #10초 후 종료

def timer(t):
    while t>0:
        print('▶', t,'초', end='₩r ')  #덮어쓰기
        t = t-1
        time.sleep(1)

timer(start)
break
```

감사합니다:)