Final Report
- 수도 맞추기 게임

컴과고 홍득조 교수님

202312746 이가은
202312839 최정식
202312814 정지연

0. 목차

1. 개요

1.1 윤곽

본 project는 수도 게임 맞추기 게임이다. 해외여행의 컨셉을 담아 게임 내 비행기를 이동시켜 나라에 도착하여 해당 나라의 수도를 맞추는 게임이다.

1.2 디자인 개요

본 project를 설계하기 위해 필요한 함수를 정의하였다. 게임 내 수도를 물어보고 답을 입력받기 위한 def Qiuz, 게임이 돌아갈 이벤트 루프, 퀴즈의 오답 여부를 설정하였다.
또한 게임 내에서 이동하는 캐릭터가 될 비행기를 만들고
비행기와 충돌하여 이벤트를 발생시킬 무인도 island 해적선 priateship 다양한 나라마다 각각의 pin의 함수를 만들었다.

1.3 개발 환경

파이썬 아이들을 이용하였다.

2. 팀원 역할 분담

2.1) 202312746 이가은
- 수도게임 맞추기를 제안하였다,
- 게임의 전체적인 틀을 작성하였다.
- 게임의 캐릭터와 이미지에 관한 모든 부분을 제안하고 코딩을 작성하였다.
- 게임 화면 내에 점수와 텍스트에 관련된 오류를 디버깅하였다.
- 레포트를 작성하였다.
- 팀원들의 만남을 주도하였다.
- 각자 짠 코딩을 합쳤다.

2.2 202312839 최정식
- 게임에 등장하는 퀴즈의 일부 (코드에서 일본~멕시코) 제안함
- 게임 엔딩에 등장하는 결과 화면 제안함
- 게임에 등장하는 퀴즈의 작동 부분 (함수, 조건문)을 만듦
- 게임의 인트로, 엔딩 창을 만듦
- 각자 짠 코딩을 합치고 마무리함
- 코드에 쓰인 개념을 정리함

2.3 202312814 정지연
- 실시간으로 점수 화면이 상단에 뜰 수 있도록 코드 작성
- 점수와 관련된 코드 작성

- 게임이 끝나면 몇 번을 맞추고 틀렸는지 계산하는 코드 작성
- 총 점수가 음수가 되면 실패 창이 뜨도록 200점이 되면 성공 창이 뜨도록 코드 작성

2.4 함께
팀원 다같이 게임의 주제를 정하고 게임을 만들기 위해 어떠한 과정이 필요한지 게임에 대한 알고리즘을 작성하였다. 퀴즈 게임에 넣을 나라를 정하고 서로의 코드를 보완하며 다같이 코드의 오류를 수정하였다. 또한 각자 코드를 일부분씩 담당하여 작성하였다.
레포트에 들어갈 팀원 역할 분담, 자신이 짠 코드 설명, ai이용, 소감을 각자 작성하였다.

3. 아이디어 논의 과정

3.1) 주제 선정
팀원들과 처음 만나 먼저 게임의 주제를 정하였다. 캐릭터가 날고 피하고 죽는 게임말고 참신한 게임이 없을까 고민을하면서 온갖 게임을 제시하는 아이스 브레이킹을 진행하였다. 리듬게임, 자판기게임, 풍선 터뜨리기..등 다양한 주제가 나왔고 그중 수도 맞추기 게임으로 결정하였다.

3.2) 알고리즘
(1) 기본 게임 틀 설정
(1.1) 수도를 맞추는 게임
게임 내에 있는 비행기를 좌우상하 키보드를 눌러 이동하여 나라에 위치한 핀과 만나면 수도 퀴즈 창이 뜨게 된다. 사용자가 수도 퀴즈 창에 답을 입력하여 정답일 경우 점수를 얻고 오답일 경우 점수를 잃게 된다. 점수는 화면에 실시간으로 반영되며 그 외 무인도와 해적선을 만들어 다양한 이벤트를 만들었다. 게임 내에서 점수가 특정 숫자 이상일 때 이하일 때 게임은 종료되며, 게임이 종료된 후 게임을 맞춘 횟수, 승률, 성공 실패 여부가 표시된다.

(1.2) 수도 정답을 맞추는 경우
- 점수 획득 / 나라마다 점수 랜덤
(1.3) 수도 정답을 틀리는 경우
- 점수 잃음

(2) 캐릭터 설정
(2.1) 비행기, 무인도, 해적선의 캐릭터를 설정
(2.2) 각 이미지를 삽입

(3) 비행기를 움직이게 하기
(3.1) 키보드를 이용하여 움직이게 하기

(4) 각 나라에 핀을 설치하기
(4.1) 수도에 맞출 나라를 선정

먼저 수도 퀴즈에 넣을 나라를 선정하였다. 나라는 러시아. 한국, 인도, 프랑스, 미국, 브라질, 아르헨티나, 호주, 이집트, 캐나다, 중국, 마다가스카르, 일본, 몽골, 튀르키예, 영국, 아이슬란드, 그린란드, 알제리, 멕시코 로 정했다.

(4.2) 나라 위치 표현

지도에 위에서 정한 나라에 핀을 설치하기 위해 스크린 내에 좌표를 구하고 삽입함

(5) 각 캐릭터마다 충돌 함수 구현

- 충돌한 상태로 엔터 키를 눌러야 한다
- 최초방문 여부를 확인

(5.1) 비행기와 나라 핀이 충돌한 경우

- 퀴즈 맞추기 창 뜸

(5.2) 비행기와 무인도이 충돌한 경우

- 점수 얻기 / 100점 획득

(5.3) 비행기와 해적선이 충돌한 경우

- 점수 잃기 / 100점 잃음

(6) 게임 종료

(6.1) 500점 이상시 게임 종료 <성공>

(6.2) -200점 이상시 게임 종료 <실패>

3.3) 게임에 이용된 개념들

- pygame : 게임 개발용 모듈로 스프라이트 쪽을 담당
- tkinter : 그래픽 인터페이스 모듈로 인트로, 퀴즈 창, 엔딩을 담당
- def : 여러 줄의 코드를 원할 때마다 바로바로 사용할 수 있도록 하나의 함수로 정의하는데 사용
- global : 하나의 함수에서만 사용 가능한 지역변수를 어디서든 사용 가능한 전역변수로 만들어줌
- while <조건문> : 조건문이 True면 작동하는 반복문
- if, elif, else : 특정 조건을 만족하는지 또는 만족하지 않을 때의 분기를 확인

4. 최종구현물

py 파일로 제출하였습니다.

5. 코드 설명

1) 파이 게임 틀

```
import pygame

pygame.init()

# 스크린 설정
screen_width = 480
screen_height = 640
screen = pygame.display.set_mode((screen_width, screen_height))

# 게임 제목 설정
pygame.display.set_caption("Capital Quiz Game")

# FPS 설정
clock = pygame.time.Clock()

# 이벤트 루프
running = True
while running:
    dt = clock.tick(30)

    for event in pygame.event.get():
        if event.type == pygame.QUIT :
            running = False

    pygame.display.update()

# pygame 종료
pygame.quit()
```

위 게임의 기본적인 틀은 다음과 같다. 파이게임이 진행되기 위해 절대적으로 필요한 요소들이다.

(1.1) 파이게임 시작
- 파이게임을 삽입하고 pygame.init()을 사용하여 파이게임을 초기화 하였다.

(1,2) 스크린 설정
screen_width, screen_height 변수를 사용해 화면 크기를 설정하였습니다.
pygame.display.set_mode로 화면을 생성하였다.
pygame.display.set_caption으로 게임 창의 제목을 설정하였습니다.
pygame.time.Clock으로 게임의 FPS를 설정하였습니다.

(1.3) 이벤트 루프
이벤트 루프를 설정하였습니다.
- running은 게임 루프를 제어하기 위한 변수입니다.
- True를 넣어 게임이 진행중인지 확인합니다.
- while runnig: 일 때 게임이 계속 돌아가게 하였습니다.
- running이 `False`로 설정되면 게임 루프가 종료됩니다.
for event in pygame.event.get(): 의 이벤트 루프를 사용해 프로그램이 중단되지 않고 중간에 사용자가 어떤 입력을 하였는지 어떤 동작이 들어오는지 확인하기 위해 작성하였습니다.
if event.type == pygame.QUIT: running = False

여러 이벤트들 중 QUIT이 들어왔을 때 창을 닫는 용도로 만들었습니다.

(1.4) 파이게임 종료
pygame 종료 처리를 하기 위해 pygamme.quit()을 이용하였습니다.

2) 배경

```
# 폰트 정의
font = pygame.font.Font(None, 40) # 폰트 객체 생성 (폰트, 크기)

# 배경 이미지 불러오기
background
= pygame.image.load("C:\\Users\\USER\\python\\pygame_basic\\background.jpeg")
```

(1) 폰트 정의
폰트 변수를 생성하였다. pygame.font.Font를 이용하여 폰트와 크기를 설정하였다.

(2) 배경 이미지 불러오기
배경 변수를 background로 설정하고 이미지를 불러왔다.
pygame.image.load(“절대경로”) 를 이용하여 배경의 이미지를 불러왔다.

3) 비행기 캐릭터 설정

```
# 이미지 삽입
character
= pygame.image.load("C:\\Users\\USER\\python\\pygame_basic\\character.png")

# 이미지 배경 제거
character.set_colorkey((255, 255, 255))

# 이미지 크기 설정
character_size = character.get_rect().size
character_width = character_size[0]
character_height = character_size[1]

# 캐릭터 좌표 설정
character_x_pos = (screen_width / 2) - (character_width/2)
character_y_pos = (screen_height / 2) - (character_height/2)

# 이동할 좌표
to_x = 0
to_y = 0
```

비행기에 대한 코드이다. 변수이름은 character로 설정하였다. 위에서 설명한 바와 같이 배경이미지를 삽입하였다.

(1) 이미지 배경 제거
set_colorkey를 이용하여 삽입한 이미지의 특정 색깔을 지웠다. (255,255,255)는 rgb 10진수 색상이며 이를 이용하여 해당 이미지의 흰색(배경)을 없앴다.

(2) 이미지 크기 설정
character_width와 변수를 만들어서 첫 번째값을 가로로 설정했다.
character_height에 두 번째값을 세로로 설정하였다.
character.get_rect().size를 이용하여 핀의 사각형 사이즈를 구했다. 나중에 충돌 이벤트를 설정할 때 쓰일 것이다.

(3) 캐릭터 좌표 설정
x_pos, y_pos 변수를 정의하였다. 캐릭터를 중앙에 위치하게 하고 싶어서 화면의 가로와 세로를 2로 나누어 중심 x 좌표, y 좌표를 구했다. 캐릭터 사이즈를 2로 나누어 빼 캐릭터가 화면 중앙에 위치하게 하였다. (ai이용)
그리고 캐릭터를 움직이게 하기 위하여 캐릭터의 좌표를 to_x, to_y로 초기화 하였다.

4) 해적선, 무인도 설정

```
# 캐릭터 해적선
pirateship                                                                    =
pygame.image.load("C:\\Users\\USER\\python\\pygame_basic\\pirateship.png")
pirateship.set_colorkey((255, 255, 255))
pirateship_size = pirateship.get_rect().size # 이미지의 크기를 구해줌
pirateship_width = pirateship_size[0] # 캐릭터의 가로 크기
pirateship_height = pirateship_size[1] # 캐릭터의 세로 크기
pirateship_x_pos = 1100 #좌표 설정
pirateship_y_pos = 200

# 캐릭터 무인도
island
= pygame.image.load("C:\\Users\\USER\\python\\pygame_basic\\island.png")
island.set_colorkey((255, 255, 255))
island_size = island.get_rect().size # 이미지의 크기를 구해줌
island_width = island_size[0] # 캐릭터의 가로 크기
island_height = island_size[1] # 캐릭터의 세로 크기
island_x_pos = 700
island_y_pos = 400
```

해적선과 무인도 객체를 설정하였다. 중복되는 코드설명은 생략하였다. 해적선과 무인도는 위치하길 원하는 좌표를 구하여 스크린에 표현하였다.

5) 나라마다 핀 작성

```
# 한국 pin
korea_pin
= pygame.image.load("C:\\Users\\USER\\python\\pygame_basic\\pin.png")

korea_pin.set_colorkey((255, 255, 255))

# 이미지 크기 구함
korea_pin_size = korea_pin.get_rect().size

# 핀 크기 설정
korea_pin_width = korea_pin_size[0]
korea_pin_height = korea_pin_size[1]

# 스크린 내 핀 위치 좌표 설정
korea_pin_x_pos = 510
korea_pin_y_pos = 330

...
```

나라 pin 에 대한 코드는 이와 같다. 중복되는 코드설명은 생략하였다.
변수는 나라이름_pin으로 설정하여 각 나라 이름으로 변수만 바꾸어 설정하였다.
또한 나라마다 좌표 위치를 구하여 스크린 내에 핀을 설치하였다.

6) 게임 변수 초기화

```
# 게임 상태
score = 0
correct_count = 0
incorrect_count = 0

# 상호작용 조건부 초기화
island_visited = False
russ_solved = False
...

# answer 값 초기화
answer = ""
```

- 게임 내에서 활용될 점수, 맞은 횟수, 틀린 횟수를 0으로 초기화 하였다.

- `score`는 현재 점수를 저장하고, `correct_count`는 맞힌 문제 수, `incorrect_count`는 틀린 문제 수를 저장한다.
- 나라별로 퀴즈를 시도했는가를 나타내는 조건부 bool을 False로 정의
- answer를 빈값으로 초기화 하였다.

7) 퀴즈 창

```
# 퀴즈 창 띄우기 함수
def open_quiz_russia():
    global answer
    root = tkinter.Tk()
    root.withdraw()
    answer=simpledialog.askstring("","러시아의 수도는: ")
    root.destroy()

def open_island():
    Message(parent=None, title="야호!", message="보물을 획득하였습니다\n(점수 100 획득)").show()

def open_pirateship():
    Message(parent=None, title="저런!", message="해적에게 점수를 빼앗겼습니다\n(점수 100 감소)").show()
```

- 나라별로 tkinter 퀴즈 창을 띄우는 함수를 정의
- 게임에 등장하는 퀴즈의 작동 부분 (함수, 조건문)
- 퀴즈의 난이도별로 주는 점수를 다르게 함
- 흔히 알고 있는 문제는 50점, 잘 모르는 문제는 100점, 정답이 긴 문제는 150점
- 무인도와 해적선과 충돌하면 점수에 관련된 창을 띄우는 함수 정의

8) 이벤트 루프

이벤트 루프에 들어가는 함수가 많아 끊어서 설명하겠다.

8-1) 이벤트 루프

```
# 이벤트 루프
is_intro = True
running = True # 게임이 진행중인가?
while running:

    dt = clock.tick(30) # 게임화면의 초당 프레임 수를 설정
    for event in pygame.event.get(): # 어떤 이벤트가 발생하였는가?
        if event.type == pygame.QUIT : # 창이 닫히는 이벤트가 발생하였는가?
            running = False # 게임이 진행중이 아님
```

'1) 게임틀 부분'과 똑같은 부분이다.

8-2) 게임의 인트로 창

```
if is_intro:
            Message(parent=None, title="게임 시작", message="==CAPITAL QUIZ GAME에 오신 것을 환영합니다!==\n이 게임은 비행기를 조종해 원하는 나라의 수도를 맞춰 수도왕이 되는 게임입니다.\n\n==플레이방식==\n표시가 된 나라에 다가간 다음 Enter 키를 누르면 문제가 나오기 시작합니다.\n문제를 맞추면 난이도에 따른 점수를 얻고, 문제를 틀린다면 50점을 잃게 됩니다.\n만약 점수가 500점 이상이 되거나 -200점 이하가 된다면 성공 또는 실패로 게임은 끝납니다.\n무인도와 해적선은 상호작용하면 당신의 점수에 어떠한 영향을 줄 겁니다.\n기억하세요! 게임 안에 있는 사물은 해적선을 제외하고는 2번 이상 방문할 수 없습니다!\n\n==조작==\n화살표 키: 비행기 조종\nEnter 키: 상호작용\n\n그러면 수도왕이 되기 위한 여행을 떠나봅시다! Let's go!").show()
            is_intro = False
```

시작할 때 인트로 창을 띄우고 확인 버튼을 누르면 게임이 시작되게 하였다.

8-3) 키보드 이벤트

```
 if event.type == pygame.KEYDOWN: # 키가 눌러졌는지 확인
            if event.key == pygame.K_LEFT: # 캐릭터를 왼쪽으로
                to_x -= 5
            elif event.key == pygame.K_RIGHT: # 캐릭터를 오른쪽으로
                to_x += 5
            elif event.key == pygame.K_UP: # 캐릭터를 위로
                to_y -= 5
            elif event.key == pygame.K_DOWN: # 캐릭터를 아래로
                to_y += 5
if event.type == pygame.KEYUP: # 방향키를 떼면 멈춤
            if event.key == pygame.K_LEFT or event.key == pygame.K_RIGHT:
                to_x = 0
            elif event.key == pygame.K_UP or event.key == pygame.K_DOWN:
                to_y = 0
```

앞에서 만들어 놓은 캐릭터를 키보드로 움직이게 하였다.
키보드를 누르면 움직이고 키보드를 떼면 멈추는 2가지 경우로 나누었다.

pygame.KEYDOWN으로 키보드가 눌렸을 때를 확인한다.
if문을 써서 방향키보드 상,하,좌,우 의 경우를 만들어 각각 방향으로 5만큼 이동하게 하였다.

pygame.KEYUP으로 방향키를 떼었을 때를 확인한다. 이 함수로 캐릭터의 움직임을 멈추게 하였다. 그 위치 그대로 있기위해 0값을 집어넣었다.

8-4) 게임 종료 조건

```
 # 게임 종료 조건 확인
        if score >= 500:
            running = False
        if score <= -200:
            running = False
```

8-5) 캐릭터 위치 디테일

```
    # 캐릭터 위치 업데이트
    character_x_pos += to_x
    character_y_pos += to_y

    # 가로 경계값 처리
    if character_x_pos < 0:
        character_x_pos = 0
    elif character_x_pos > screen_width - character_width:
#오른쪽에서 그림이 그려지기 때문에 캐릭터 가로 길이 만큼 빼주기
        character_x_pos = screen_width - character_width

    # 세로 경계값 처리
    if character_y_pos < 0:
        character_y_pos = 0
    elif character_y_pos > screen_height - character_height:
#오른쪽에서 그림이 그려지기 때문에 캐릭터 가로 길이 만큼 빼주기
        character_y_pos = screen_height - character_height
```

(1) 캐릭터 위치 업데이트
캐릭터 위치가 계속 바뀌는 것을 업데이트 하기 위해 character_x_pos를 정의하였다.

(2) 경계값 처리
캐릭터가 화면 밖으로 나가는 것을 방지하기 위해 함수를 설정하였다.
캐릭터의 좌표가 화면의 왼쪽 (0,0)을 넘어갔을 때와 오른쪽 (스크린 가로)을 넘어갔을 때의 경우를 설정하였다.

8-6) 충돌 처리 rect 정보 업데이트

```
# 충돌 처리를 위한 rect 정보 업데이트
    character_rect = character.get_rect()
    character_rect.left = character_x_pos
    character_rect.top = character_y_pos

    pirateship_rect = pirateship.get_rect()
    pirateship_rect.left = pirateship_x_pos
    pirateship_rect.top = pirateship_y_pos

    island_rect = island.get_rect()
    island_rect.left = island_x_pos
    island_rect.top = island_y_pos

    russia_pin_rect = russia_pin.get_rect()
    russia_pin_rect.left = russia_pin_x_pos
    russia_pin_rect.top = russia_pin_y_pos
```

get.rect를 이용하여 가로와 세로 길이를 가져온다. 그리고 바뀐 좌표를 가져와 캐릭터 직사각형의 왼쪽과 위에 부분에 업로드를 한다.

8-7) 충돌 체크

```
# 충돌 체크
    if character_rect.colliderect(pirateship_rect):
        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_RETURN:
                open_pirateship()
                score -= 100



    if character_rect.colliderect(island_rect):
        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_RETURN and island_visited == False:
                open_island()
                score += 100
                island_visited = True
```

rect.colliderect를 이용하여 캐릭터가 부딪혔을 때 상황을 가정한다.
캐릭터와 해적선이 부딪혔을 때 키보드가 눌러져있다면 open_pirateship()를 실행하고 점수를 100점을 잃게한다. 무인도도 이와 같다.

8-8) 게임에 등장하는 퀴즈의 작동 부분 (함수, 조건문)

```
        # 퀴즈 체크
        if character_rect.colliderect(korea_pin_rect):
        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_RETURN and kore_solved == False:
                open_quiz_korea()
                if answer=="서울":
                    score +=50
                    correct_count += 1
                if answer!="서울":
                    score -= 50
                    incorrect_count += 1

                kore_solved = True
```

8-7과 같이 나라와 캐릭터가 부딪혔을 때의 상황도 만들어주었다.

- 나라에 꽂힌 pin과 플레이어가 충돌했을 때 엔터 키를 누르면 그 나라에 해당하는 퀴즈 함수를 시작
- answer를 전역변수로 하여 함수를 끝내고 입력한 답을 인식
- 함수가 끝나면 퀴즈 시도 조건부를 True로 설정하여 같은 문제를 다시 시도할 수 없게 만듬
- 퀴즈의 난이도별로 주는 점수를 다르게 함
- 흔히 알고 있는 문제는 50점, 잘 모르는 문제는 100점, 정답이 긴 문제는 150점

부딪히면 open_quiz_korea()를 실행한다. 실행한 후 답이 정답일 경우 점수추가

8-9) 화면 업데이트

```
    score_text = font.render(f"Score: {score}", True, (0, 0, 0)) # 화면에 점수 표시

    screen.blit(background, (0, 0)) # 배경 그리기
    screen.blit(score_text, (10, 10)) # 글자 그리기
    screen.blit(pirateship, (pirateship_x_pos, pirateship_y_pos)) # 적 그리기
    screen.blit(island, (island_x_pos, island_y_pos)) # 섬 그리기
    screen.blit(russia_pin, (russia_pin_x_pos, russia_pin_y_pos)) # 나라 핀 그리기
        ...
    screen.blit(character, (character_x_pos, character_y_pos)) # 캐릭터 그리기

    pygame.display.update() # 게임 화면을 다시 그리기(반드시 필요)
```

screen.blit을 이용해서 화면에 만들어둔 캐릭터, 글자, 배경 등을 업데이트를 해준다. . font.render를 이용해 현재 점수를 나타내는 텍스트를 생성한다.

9) 엔딩 창

```
# pygame 종료
if correct_count+incorrect_count != 0:
    correct_percent = int(100 * correct_count / (correct_count + incorrect_count))
else:
    correct_percent = 0
Message(parent=None, title="게임이 종료되었습니다!", message=f"==게임 결과==\n\n최종 점수: {score}\n정답 수: {correct_count}개\n오답 수: {incorrect_count}개\n정답률: {correct_percent}%").show()

pygame.quit()
```

게임이 특정 조건(성공/실패)에 들어서면 정답률을 계산하고 엔딩 창에 정답률을 포함한 결과를 띄운 다음 게임은 종료된다..

6. ai 이용

7.1) 202312746 이가은

- 캐릭터 비행기의 위치를 스크린 중앙으로 오는데에 위치를 계산하는데 쓰였다.
- 초기 인트로 부분이 실행이 되지 않아 디버깅에 사용

7.1) 202312814 정지연

처음 게임을 짜기에 앞서 파이 게임이 어떤 것인지 알기 위해 그리고 기본적인 개념과 함수를 알기 위해서 챗 GPT를 활용했습니다. 제가 짜야하는 게임 코드가 점수 계산과 관련된 부분이었기에 미리 정하고 생각할 것들이 많았습니다. 첫 번째는 수도의 정답을 맞추거나 틀리게 되면 실시간으로 점수 화면이 상담에 출력되도록 하고 싶었습니다. 이를 실행시키기 위해 도구의 도움을 받았습니다. 두 번째는 게임에서 점수를 얻고 잃는 것을 바로 계산해주는 코드가 필요했습니다. 정답을 맞추면 +10 틀리면 –10 무인도와 접하게 된다면 +100 해적선과 접하게 된다면 –100점으로 점수를 바로 계산해주는 코드를 짜기 위해 도움을 청했습니다. 세 번째는 게임이 종료되는 시점을 점수와 관련하여 정하여 도합 점수가 음수가 될 때 실패 창이 뜨게 하고 200점이 되었을 때 성공 창이 뜨게 하기 위해 질문을 했습니다. 마지막으로 게임이 끝나게 되면 몇 번을 맞추고 틀렸는지를 계산하여 창에 나타나도록 하기 위해 도구를 활용하였습니다. 더하여 무인도에 접했을 때 점수를 얻을 수 있는 건 게임 중 단 한번을 만들기 위해 도구를 활용했습니다. 그리고 결국 이루지는 못 했지만 배가 육지가 아닌 바다로만 계속해서 움직이고 있게 하는 옵션을 추가하고 싶어 도움을 청했습니다.

7.3) 202312839 최정식

- pygame 중에 tkinter의 창을 띄우는 함수 예시를 생성
- 퀴즈 함수의 초기 버전
- 개발 중 발생한 버그를 디버깅하는 데 도움이 되는 예시를 생성

7. 느낀점

7.1) 202312746 이가은

처음에 과제를 보고 실화인가 생각이 들었고 친구도 없고 실력도 없는데 조별과제를 할 생각을 하니 막막하였다.
원래도 소심한 편이라 팀원들을 이끌어가면서 힘든적이 많았지만 지나고 나니 뿌듯하였다.
처음에는 오류도 많고 어디서부터 코드를 짜야할지 막막하였고 오류도 많이 발생하였다. 그렇지만 오류를 찾고 게임을 다 만들고 난 뒤 성취감은 말로 표현할 수 없이 짜릿하였다.

7.2) 202312814 정지연

처음 이 과제를 부여받았을 때는 '내가 게임을 만들 수 있을지', '어떻게 게임을 만들지' 막막하고 더 나아가 과제 제출을 할 수 있을지 걱정이 되었다. 하지만 일단 나에게 주어진 거고 팀원들에게 해가 되면 안 되겠다는 생각에 '최선을 다 해보자'라는 생각을 가지고 과제를 하기 시작했다. 각자 어떤 게임을 만들면 좋을지 사전 조사를 하고 팀원들과 만나기로 한 첫날

이었다. 팀원들과 만나 자신이 찾아본 것들에 대해 이야기하고 또 다른 아이디어도 내고 하니 무언가를 같이 하는 데서 오는 즐거움을 느낄 수 있었다. 이후에도 같이 회의도 하고 서로 협력하니 점점 게임이 만들어지고 완성되어갔다. 처음에는 막막했던 것이 누군가와 같이 힘을 합치니 즐겁고 하고 싶은 것으로 바뀌었다는 게 신기했다. 이번 과제는 꽤 복잡했고 혼자 하기에는 쉽지 않은 작업이기 때문에 팀원들과 효율적인 협업과 소통이 중요하다는 것을 느꼈다. 또한 테스트와 디버깅을 철저히 수행해야 한다는 것을 깨달았다. 여러 코드를 입력해 보고 팀원들과 구상했던 다양한 상황과 조건들을 시뮬레이션하고 버그를 찾아 수정하는 과정이 게임의 완성도를 높여준다고 생각했기에 여러 번의 테스트와 디버깅은 필수였다. 마지막으로 게임의 규칙이나 그림과 같은 게임의 디테일한 부분을 신경 쓰는 것도 중요하다는 것을 느꼈다. 이번 과제는 많이 생각하고 여러 번 수정하고 실행해야 했기에 힘들고 머리 아픈 부분도 있었지만 결국 팀원들과 힘을 합쳐 우리만의 게임을 만들었다는 결과를 내니 뿌듯하고 보람차다.

7.3) 202312839 최정식

처음에는 목표가 갑작스러워 어렵고 막연했지만 조원들간의 상의 과정을 거치고 조원들이 만들어준 코드 위에 알고 있는 정보들을 탑 쌓듯이 합치면서 하나의 작동하는 게임을 만드는 그 과정에서 상황에 맞는 좋은 팀플레이와 배운 내용을 적재적소에 활용한다는 것이 무엇인지 알게 되었다.