일일 업무 사항 정리

작성자	제품팀 이민성 인턴
업무 일시	2022.09.12~2022.09.16

세부 사항

1. 업무 내역 요약 정리

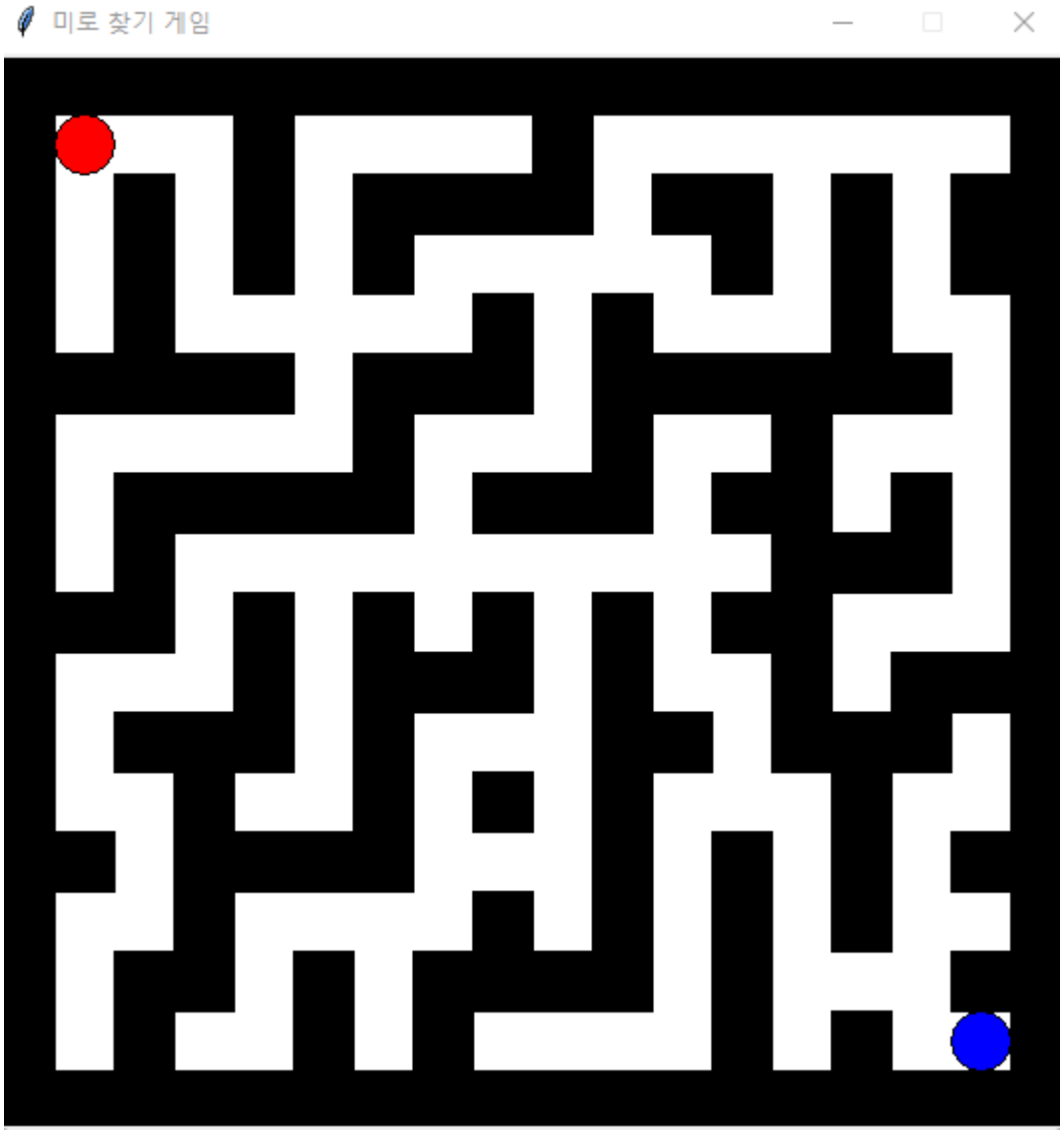
목표 내역	Done & Plan
1. 쉘 스크립트 언어 이해 및 활용 2. 미로게임 만들기 3. 오라클 데이터베이스 설치과정 복기	for문과 while문 등을 이용해 미로 찾기 게임을 만들려고 시도했습니다. 하지만 쉘 언어로 할 방법이 생각나지 않아 같은 스크립트 언어인 파이썬으로 만들어 보았습니다. 파이썬을 잘 다루지는 못하지만 인터넷을 참조하며 만들어 보았습니다. 후에 쉘 언어로 바꿔서 해보려고 했지만 tkinter등 다른 점이 있어서 어려워서 실패하였습니다. 파이썬 파일을 첨부하였습니다.

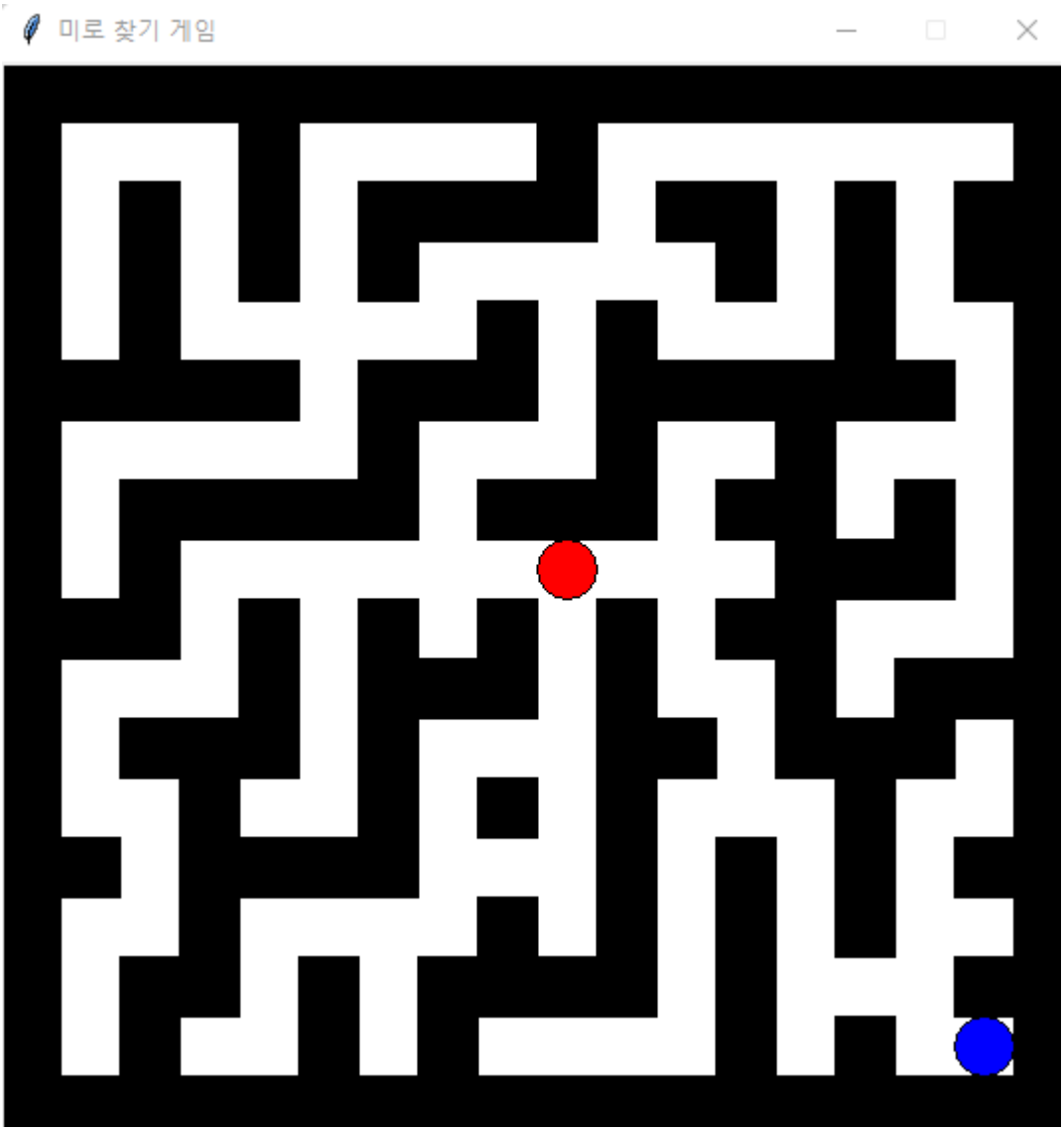
2. 내용 세부 (업무 세부 내역 정리 및 기타 사항 정리)

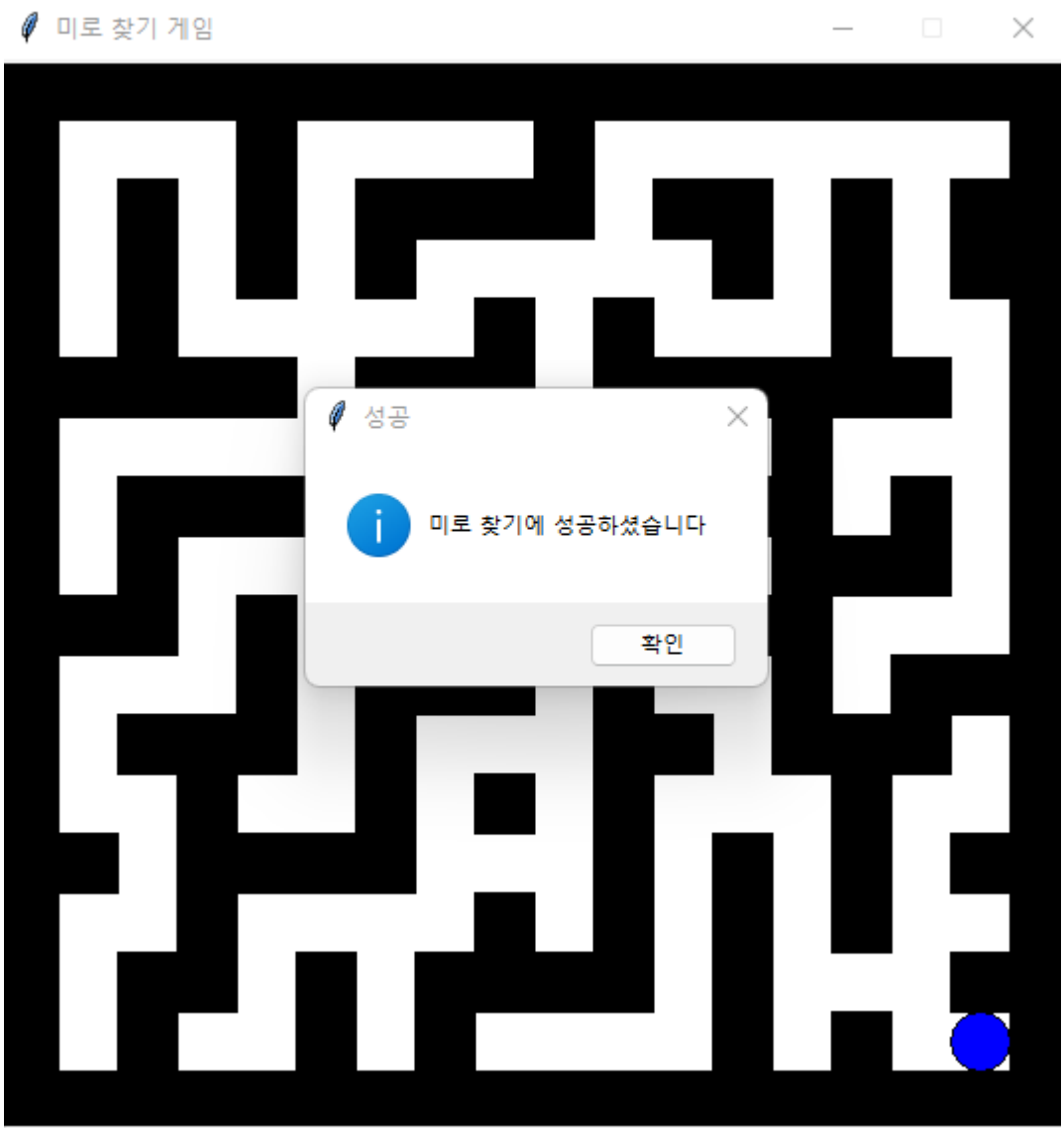
```
from tkinter import *
from tkinter import messagebox # 창을 띄우기 위해 tkinter 을 import 해준다

class Player(): # 플레이어가 움직일 수 있게 해준다
    def __init__(self, canvas, x, y):
        self.canvas = canvas
        self.id = canvas.create_oval(x * 30, y * 30, x * 30 + 30, y * 30 + 30, fill =
"red")
        self.x, self.y = x, y
        self.nx, self.ny = x, y # 추후 이동할 수 있는 위치인지 판별하기 위해 nx, ny 변수를
만들었다

    def move(self, direction): # 플레이어의 움직임을 처리할 메소드
        if direction == 'w':
            self.nx, self.ny = self.x, self.y - 1
        elif direction == 'a':
            self.nx, self.ny = self.x - 1 , self.y
        elif direction == 's':
            self.nx, self.ny = self.x, self.y + 1
        elif direction == 'd':
            self.nx, self.ny = self.x + 1, self.y

        if not self.is_collide(): # 만약 벽이 아닐 경우 위치를 갱신하며 이동시킨다
            self.canvas.move(self.id, (self.nx - self.x) * 30, (self.ny - self.y) * 30)
            self.x, self.y = self.nx, self.ny

        if map[self.y][self.x] == 3: # 목표지점에 도달하면 게임 클리어를 알리는 창 하나를
띄워준다
            messagebox.showinfo(title="성공", message="미로 찾기에 성공하셨습니다")

    def is_collide(self): # 이동한 위치가 혹시 벽인지 아닌지 판별하기 위해 is_collide 메소드를
만들어준다
        if map[self.ny][self.nx] == 1:
            return True
        else:
            return False

def keyEvent(event): # 입력받은 키를 player 의 move 메소드에 전달하는 이벤트를 작성하고
    player.move(repr(event.char).strip("'"))
```

```
root = Tk()
root.title("미로 찾기 게임")
root.resizable(False, False) # 창의 제목, 사이즈 조절 가능 여부도 설정해준다

width, height = 540, 540
x,y = (root.winfo_screenwidth() - width) / 2, (root.winfo_screenheight() - height) / 2 #
창의 크기를 정하고 가운데 지점을 알아내서
root.geometry("%dx%d+%d+%d" % (width, height, x, y)) # geometry 메소드로 창을 중앙에 배치해
준다

canvas = Canvas(root, width=width, height=height, bg="white")
canvas.bind("<Key>", keyEvent) # 키 입력을 받을 시 호출해주면
canvas.focus_set()
canvas.pack() # 이제 여기에 게임 화면을 그릴 canvas를 붙여준다

# 1 : 벽, 2 : 플레이어 시작 지점, 3 : 골인 지점
map = [
    [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 2, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1],
    [1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1],
    [1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1],
    [1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1],
    [1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1],
    [1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1],
    [1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1],
    [1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1],
    [1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1],
    [1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1],
    [1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1],
    [1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1],
    [1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1],
    [1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1],
    [1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1],
    [1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 3, 1],
    [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
]

for y in range(len(map[0])): # 이를 토대로 반복문으로 미로를 그려준다
    for x in range(len(map[y])):
        if map[y][x] == 1:
            canvas.create_rectangle(x * 30, y * 30, x * 30 + 30, y * 30 + 30,
fill="black" )
```

```
        elif map[y][x] == 2:
            player = Player(canvas, x, y) # player 클래스를 이용해 플레이어를 맵에 배치해주고
        elif map[y][x] == 3:
            canvas.create_oval(x * 30, y * 30, x * 30 + 30, y * 30 + 30, fill="blue" )
            # 각 오브젝트의 크기는 30 x 30 으로 벽의 경우 검정 네모, 플레이어는 빨강 원,
골인지점을 파랑 원을 그려주고

root.mainloop()
```




미로 찾기 게임


미로 찾기 게임
성공
미로 찾기에 성공하셨습니다
확인