2025년 1학기

**프로그래밍과 문제해결**

**Assignment #2**

담당교수: 윤은영

학번: 20250988

학과: 무은재학부

이름: 홍준우

POVIS ID: jwhong

**명예서약(Honor code)**

“나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다”

**Problem: 다마고치 프로그램**

1. 문제의 개요

본 프로그램을 간략히 설명하면 다음과 같다.

* 사용자와 컴퓨터가 상호작용하며 빙고게임을 플레이한다
* 사용자는 파일로부터 빙고판을 생성하거나, 랜덤으로 생성할 수 있다.
* 빙고판에 대한 정보는 2차원 리스트로 저장된다.
* 프로그램의 로그는 파일로서 저장된다.

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

스크린샷, 폰트, 사각형, 라인이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

* 입력부 : 사용자의 입력에 따라 보드를 생성하고, 문자 선택을 받는다.
* 처리부 : 사용자의 입력에 따라 빙고 상태를 확인하고, 컴퓨터 선택을 도출한다.
* 출력부 : 화면에 보드 상태를 출력하고 로그를 저장한다.

2. 알고리즘

해당 프로그램의 작성을 위한 알고리즘을 Pesudo 코드 형태로 나타내면 다음과 같다.

|  |
| --- |
| **Pesudo code for Tamagotchi** |
| Initialize:  checked = []  LOG\_PATH = "result.txt"  Function create\_board\_from\_file(file\_path):  Open file at file\_path  Read and strip all lines  size = first line as integer  player\_board = lines[1 to size]  computer\_board = lines[size+1 to 2\*size]  Return player\_board, computer\_board, size  Function create\_random\_board(size):  board\_alphabet = shuffled list of size^2 capital letters  player\_board = sliced 2D array from board\_alphabet  Shuffle board\_alphabet again  computer\_board = sliced 2D array from board\_alphabet  Return player\_board, computer\_board, size  Function boards\_to\_string(player\_board, computer\_board, size, hide\_comp=True):  Initialize s = ""  For i in 0 to size-1:  Append horizontal lines for both boards  Append player row values  Append computer row values (hide if needed)  Append bottom border  Return s  Function write\_log(text):  Append text + newline to LOG\_PATH  Function mark\_board(board, choice, size):  Add choice to checked  For each cell in board:  If cell == choice → replace with '#', return True  Return False  Function check\_bingo(board, size):  For each row and column:  If all cells == '#' → return True  If both diagonals are '#' → return True  Else return False  Function valid\_character(c, size):  Convert c to uppercase  If c is length 1, in allowed alphabet range, and not in checked → return True  Else → return False  Function get\_player\_choice(board, size):  LOOP:  Prompt user for input  If valid\_character → return input  Else print "Invalid choice"  Function get\_computer\_choice(board, size):  choices = all unmarked characters in board  Return random choice from choices  Function main():  checked = []  Clear LOG\_PATH file  LOOP:  Ask mode (1: Random, 2: File)  If 1:  Ask size (3~5) until valid  Call create\_random\_board(size)  Else:  Ask file\_path  Call create\_board\_from\_file(file\_path)  Break when valid  Print and log initial boards  WHILE True:  # Player Turn  player\_choice = get\_player\_choice()  Call mark\_board() for both boards  Print and log boards  If check\_bingo(player\_board or computer\_board) → break  # Computer Turn  Wait 0.5s  computer\_choice = get\_computer\_choice()  Call mark\_board() for both boards  Print and log boards  If check\_bingo(player\_board or computer\_board) → break  If both bingo → result = "Draw!"  Else if player\_bingo → result = "You win!"  Else → result = "Computer wins!"  Print and log result  Call main() |

3. 프로그램 구조 및 설명

a) 자료구조

* player\_board : 플레이어의 빙고판 (2차원 리스트)
* computer\_board : 컴퓨터의 빙고판 (2차원 리스트)
* checked : 현재까지 선택된 알파벳을 저장하는 리스트
* size : 보드의 크기 (3, 4, 5 중 하나)
* LOG\_PATH : 게임 진행 내용을 기록하는 로그 파일 경로 (result.txt)

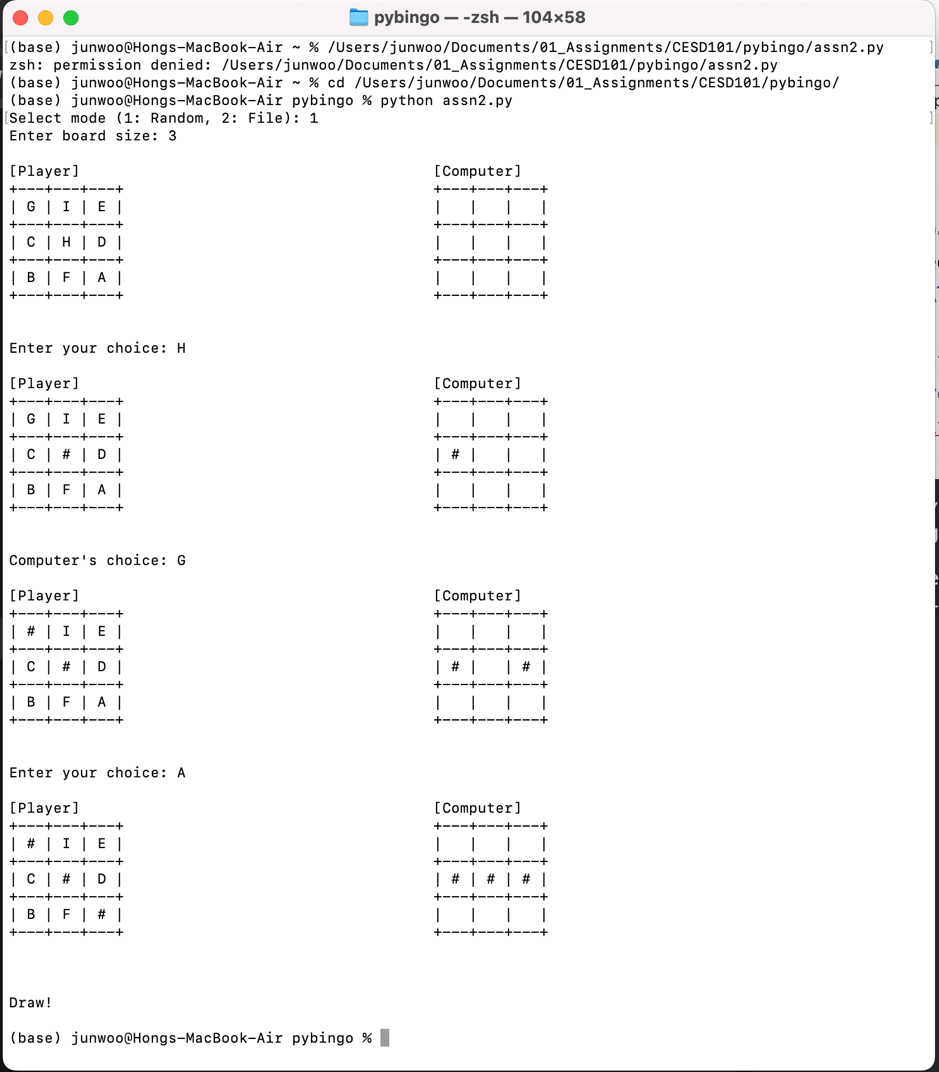
b) 주요 함수

* create\_board\_from\_file(file\_path) : 파일에서 보드를 불러와 player\_board, computer\_board, size를 초기화
* create\_random\_board(size) : 무작위로 섞인 알파벳으로 player\_board와 computer\_board를 초기화
* boards\_to\_string(player\_board, computer\_board, size, hide\_comp=True) : 현재 보드 상태를 문자열로 반환하며, hide\_comp=True 일 경우 컴퓨터의 보드는 숨김
* wirte\_log(text) : 게임 진행상황을 result.txt에 기록
* valid\_character(c, size) : 입력된 알파벳이 유효한지 확인
* get\_player\_choice(board, size) : 사용자로부터 유효한 알파벳을 입력받음
* get\_computer\_choice(board, size) : 현재까지 선택되지 않는 알파벳 중 하나를 무작위로 선택
* mark\_board(board, choice, size) : 입력된 알파벳이 보드에 있다면 ‘#’으로 마크
* check\_bingo(board, size) : 가로, 세로,, 대각선 중 하나라도 ‘#’로 마크되어 있으면 빙고 성립
* main() : 프로그램 전체 흐름을 담당하는 메인 루프

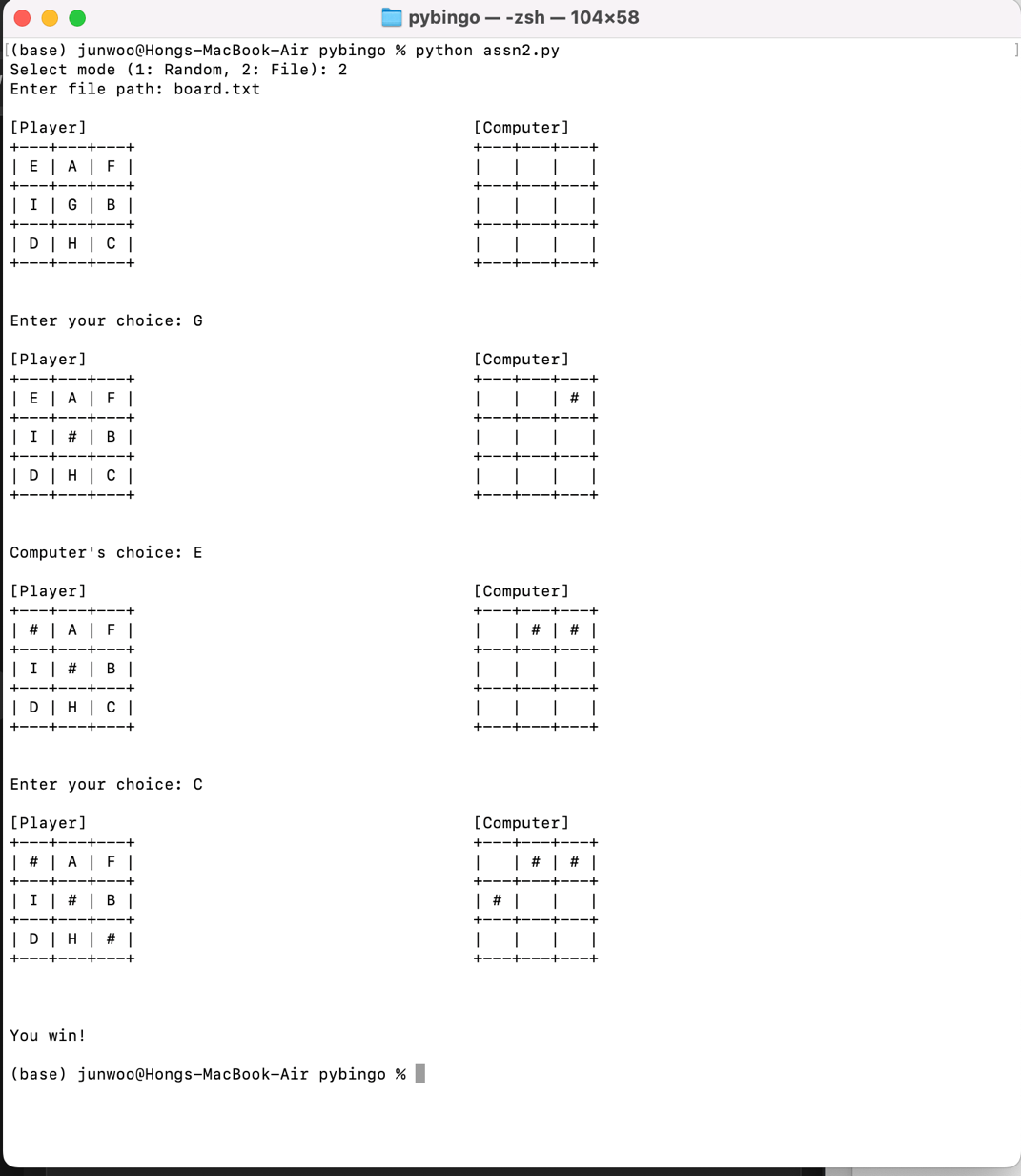
4. 프로그램 실행방법 및 예제

terminal에서 assn2.py가 있는 디렉토리로 이동, “python assn2.py” 커맨드를 실행한다.

a) 렌덤 모드 선택 후 플레이



b) 파일 모드 선택 후 플레이



c) result.txt 예시

텍스트, 영수증, 번호, 문서이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.

5. 토론

* 전역변수의 checked 의 사용으로 중복선택 여부를 관리하였지만, 이로 인해 함수 내에서 global 키워드를 사용해야하는 불편함이 발생하였다.
* 게임 진행상황을 파일로 기록하기 위해 wirte\_log() 함수를 따로 정의하여 코드의 중복을 줄이고 일관성을 유지할 수 있었다.
* 사용자 입력과 컴퓨터 선택로직을 분리하여 처리함으로써 각 역할의 책임이 명확해졌으며, 추후 UI나 로직을 개별적으로 개선하기 쉬운 구조가 되었다.

6. 결론

* 본 과제를 통해 함수 분할과 파일 입출력, 그리고 조건문과 반복문을 활용하여 하나의 완성된 프로그램을 구현하는 능력을 향상시킬 수 있었다.
* 또한 사용자 입력 검증, 상태 업데이트, 로그 기록 등의 기능을 순차적으로 구현하면서, 프로그램 흐름을 체계적으로 구성하는 훈련이 되었다.

7. 개선방향

* 게임 전체를 클래스 기반으로 재구성한다면 상태 관리와 기능 확장(예: 점수 기록, 다회차 게임 기능 등)에 더욱 유리할 것으로 기대된다.
* 현재는 각 보드 출력 시 동일한 코드를 반복 사용하고 있으므로, 출력 형식을 통일하는 전용 함수나 템플릿을 활용하면 코드 가독성이 높아질 수 있다.
* 선택 가능한 문자의 목록을 실시간으로 보여주는 기능을 추가하면 사용자의 편의성이 향상될 수 있다.
* 게임 종료 시 각 턴의 선택과정을 시각적으로 다시 출력해주는 “리플레이” 기능을 추가하면 게임의 몰입도를 높일 수 있다.