# PROJEKT „MASTERMIND”

*wykonany przez: Marfenko Mykhailo*

# *Treść*

**1. Cel i opis projektu** ..... 1

**2. Structura oraz klasy programu** ..... 1

2.1 Plik game.py ..... 1

2.2 Plik mastermind.py ..... 1

2.3 Plik button.py ..... 2

2.4 Plik test_mastermind_game.py ..... 2

2.5 Plik data.json ..... 2

**3. Instrukcja obsługi** ..... 2

3.1 Uruchamianie programu ..... 2

3.2 Format plików konfiguracyjnych ..... 2

3.3 Interfejs gry ..... 3

**4. Podsumowanie** ..... 3

# 1. Cel i opis projektu

Celem projektu jest poszerzenie wiedzy z zakresu programowania oraz zaliczenie przedmiotu. Do zrealizowania dostałem grę "Mastermind". Postanowiłem robić ten program używając biblioteki pygame jako narzędzie do interfejsu gry ponieważ ta biblioteka jest najbardziej używana dla takiego rodzaju gier na Pythonie oraz istnieje bardzo szczegółowa i czytelna dokumentacja do tej biblioteki. Zaimplementowałem możliwość grania zarówno jak między dwoma graczami, tak i z dwa rodzajami sztucznej inteligencji.

Zasady gry są bardzo proste. Jeden z graczy układa kod składający się z różnokolorowych elementów, następnie ukrywa go przed drugim graczem. Jego zadaniem będzie z kolei odgadnięcie tego kodu w skończonej liczbie prób.

# 2. Structura oraz klasy programu

Program składa się z 5 plików i folderu Assets. O każdym w kolejności:

## 2.1 Plik game.py

To jest główny plik zawierający główną pętlę gry. Z tego pliku uruchamia się grę interpreterem python'a. Można podzielić ten plik na 3 kluczowe części:

1) Najpierw deklaruję się wszystkie zmienne globalne oraz ładowanie obrazków, z którymi będziemy dalej pracować.

2) Dalej następuje szereg funkcji, które tworzą główną funkcjonalność programu, a także tworzone są przyciski, które odnoszą się (refer to) do powyższych funkcji.

3) A na końcu znajduje się główna pętla gry, w której dzieje się magia! Ona zajmuje się obsługą zdarzeń oraz wszystkich elementów w trakcie działania programu (np. przesuwaniem obiektów, rysowaniem obiektów na ekranie, itd.) W tym miejscu w razie potrzeby wywoływane są funkcje odświeżania ekranu (screen update). Ekran jest warunkowo podzielony na 2 części, w których ekran jest aktualizowany. W przypadku zmiany stanu przycisku (active – disabled) aktualizowana jest tylko lewa strona ekranu. Jeśli coś się zmieni na plansze, aktualizowany jest cały ekran. Z tego powodu znacznie poprawiła wydajność mojej gry. *aktualizowany ekran = updating screen

## 2.2 Plik mastermind.py

Ten plik zawiera w sobie główną klasę gry "MastermindGame". Ta klasa ma następującą funkcjonalność: 1) Generowanie kodu. 2) Sprawdzenie odpowiedzi gracza i zwrócenie odpowiedniego wyniku. 3) Obsługiwanie sztucznej inteligencji.

Instancja tej klasy pojawia się każdego razu, kiedy użytkownik zaczyna grę.

Obsługiwanie sztucznej inteligencji w tej klasie dzieje się to następująco: Najpierw generowane są wszystkie możliwe kombinacje odpowiedzi, które są przechowywane w zmiennej „possibilities”. Po otrzymaniu wyniku za odpowiedź sztuczna inteligencja tworzy nowe możliwości odpowiedzi na podstawie wcześniejszych odpowiedzi i wyników. W ten sposób zawsze generowana jedna z pozostałych odpowiedzi, zmniejszając liczbę możliwych odpowiedzi.

### 2.3 Plik button.py

Ten plik zawiera klasę "Button". Ta klasą jest odpowiedzialną za wszystkie przyciski w grze. Tutaj śledzone są zdarzenia (events) kliknięcia i zdarzenie, gdy wskaźnik myszy znajduje się nad przyciskiem. Również tutaj jest zaimplementowana rendering przycisku.

### 2.4 Plik test_mastermind_game.py

Tu są wszystkie testy programu, które odpowiadają za sprawdzenie działania sztucznej inteligencji oraz systemu oceniania odpowiedzi.

### 2.5 Plik data.json

Plik ten zawiera informacje o zapisanej grze. Gra jest zapisywana po naciśnięciu przycisku. Można także załadować sesję gry, klikając przycisk.

### 2.6 Folder Assets

Ten folder zawiera wszystkie elementy interfejsu gry, takie jak przyciski, plansza i inne elementy gry.

*Szczegółowa dokumentacja opisująca działanie każdej funkcji znajduje się w kodzie.

# 3. Instrukcja obsługi

### 3.1 Uruchamianie programu

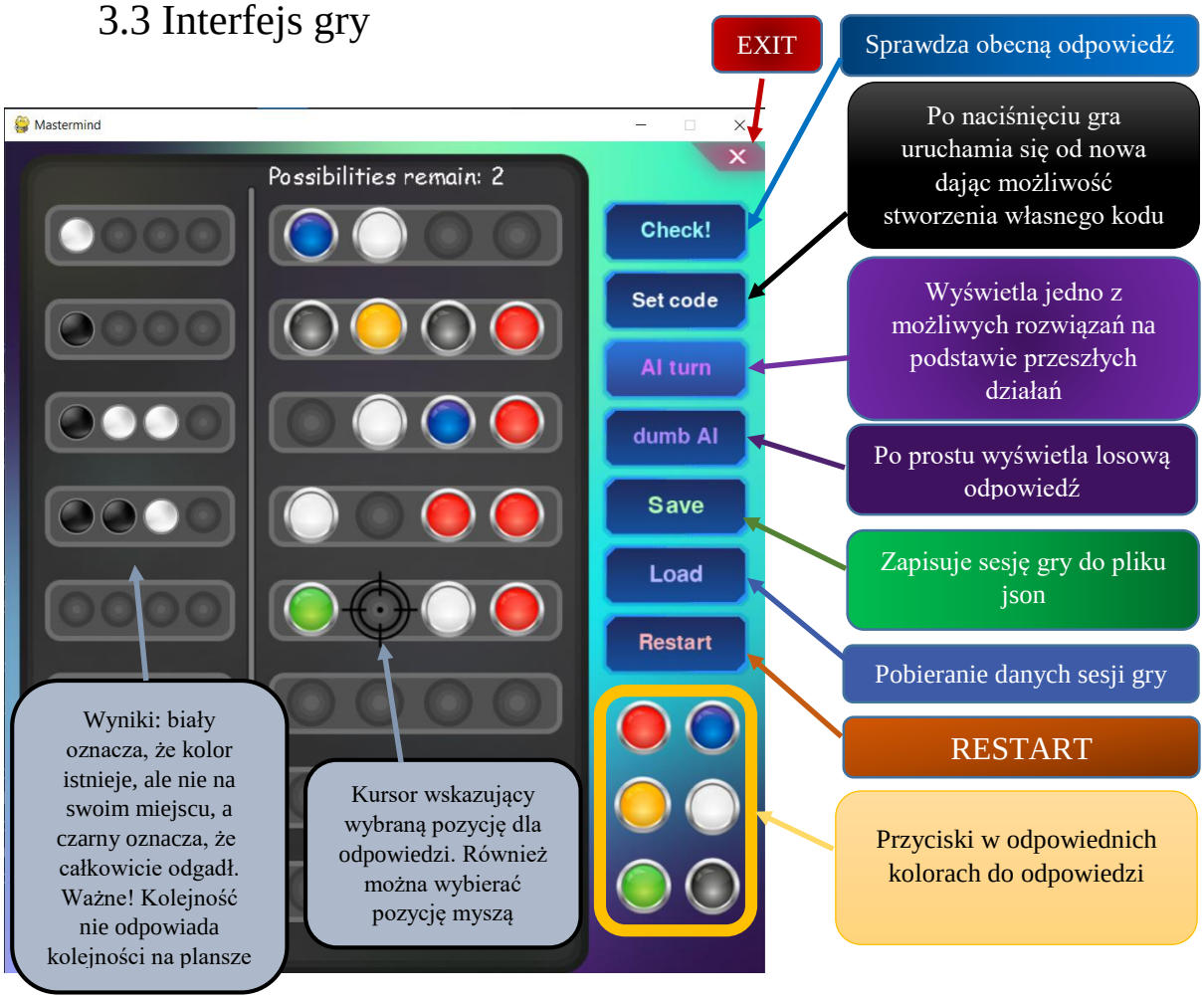
Aby zacząć zabawę należy uruchomić plik main.py interpreterem python’a.

Wymagane biblioteki: pygame, easygui.

### 3.2 Format plików konfiguracyjnych

Plik data.json ma 6 parametrów, które zawierają informacje o zapisanej sesji gry: "selected_column" i "selected_row" oznaczają, która pozycja na plansze jest wybrana. W "player_grid" zapisane wszystkie odpowiedzi gracza (lub komputera) na plansze. Odpowiednio w "score_grid" są zapisane wszystkie wyniki. W "possibilities" zapisana informacja o możliwych kombinacjach kodu które mają sens basując na przeszłych działaniach. A w "code" jest zapisany kod.

### 3.3 Interfejs gry




# 4. Podsumowanie

W ten projekt zaimplementowałem wszystkie funkcjonalności, które chciałem i jestem bardzo zadowolony z rezultatu oraz z procesu robienia programu. Po pierwsze znacznie poszerzyłem swoją wiedzę o języku programowania Python i nauczyłem się wielu nowych możliwości języka. Po drugie na podstawowym poziomie zrozumiałem, jak robi się i jak działają gry takiego rodzaju.

Ogólnie chcę powiedzieć, że ten projekt był bardzo interesujący w robieniu, ponieważ sama gra jest bardzo ekscytująca i edukacyjna. Szczególnie zafascynowała mnie implementacja sztucznej inteligencji. To było trudne, ale w końcu moje wysiłki były uzasadnione.

Moim zdaniem, zdobyłem duże doświadczenie, robiąc ten projekt i każdemu, kto ucze się język programowania, radzę zacząć od takich niewielkich projektów.

*Przepraszam za moje błędy w niektórych mięscach, nie jestem polakiem