Opis i wymagania do projektu zaliczeniowego nr 1 - mini-gra w języku Python

Celem projektu jest napisanie prostej gry w języku Python (z wykorzystaniem biblioteki `pygame`) z elementami algorytmiki.

- **Gotowy program należy zaprezentować osobiście na konsultacjach oraz przesłać na platformę Moodle.**
- Wymagana jest bezwzględna znajomość prezentowanego kodu źródłowego tzn. trzeba umieć odpowiedzieć na pytanie: "Co to jest i do czego służy?".
- Program powinien działać poprawnie wykonując zamierzone przez programistę zadania.
- Program musi wykorzystywać paradygmat programowania obiektowego.
  W tym:
  - należy tworzyć własne klasy;
  - wykorzystywać dziedziczenie (chociażby po klasach wbudowanych).
- Ponadto należy zadbać o:
  - poprawne rozmieszczenie kodu w pliku źródłowym (lub plikach);
  - czytelność kodu i konsekwentnie stosować styl nazewniczy;
  - ogólną estetykę kodu i działanie programu.
- Możliwe są dwa typy projektu nr 1:
  - **Projekt typ I** - gra napisana w języku Python $3.x$ z wykorzystaniem biblioteki `pygame` ilustrująca sortowanie bąbelkowe. Po uruchomianiu programu gracz powinien ujrzeć ciąg przypadkowo ułożonych liczb lub liter[1], które należy posortować w kolejności rosnącej (lub odwrotnie) wykorzystując algorytm sortowania bąbelkowgo. Gracz powinien mieć możliwość przestawiania kolejności wskazanej pary sąsiednich elementów (tych które są w jakiś sposób znaczone np. ujętych w ramkę). Jeżeli gracz dokona przestawienia elementów w nieodpowiedniej kolejności, gra powinna zaczynać się od nowa (od początkowego ustawienia elementów). Gra powinna kończyć się gdy

[1] Mogą to być również inne obiekty, które można sortować.

wszystkie elementy będą uporządkowane zgodnie z krokami wynikającymi z algorytmu sortowania bąbelkowgo.

Program może zawierać następujące klasy:

* `Letter` (lub `Number`) - klasa reprezentująca obiekt litery (oparta na klasie `pygame.sprite.Sprite`), z takimi metodami jak `__init__()`, `run()`, `draw()` i `update()`;
* `Frame` - klasa reprezentująca obiekt ramki;
* `Button` - klasa reprezentująca obiekt przycisku;
* `Game` - główna klasa odpowiadająca za całą rozgrywkę z takimi metodami jak `__init__()`, `draw()`, `update()`, metodą obsługującą zdarzenia oraz metodą wyznaczającą prawidłową kolejność ruchów (sortowania liter);
* `Text` - klasa odpowiadając za obiekty tekstu wyświetlane na ekranie.

Pisząc program można wykorzystać grafiki umieszczone na platformie Moodle. Poniżej przykładowy screen gry:

– **Projekt typ II** - gra napisana w języku Python $3.x$ z wykorzystaniem biblioteki `pygame` ilustrująca znajdowanie drogi w labiryncie wykorzystując przeszukiwanie w głąb. Gra ma polegać na przejściu labiryntu (wylosowanego spośród kilku dostępnych) wykorzystując technikę przeszukiwania grafu głąb. Gracz wchodzi w głąb labirynty, który początkowo jest zakryty zawsze kierując się zasadą, że w pierwszej kolejności na wschód, następnie na południe, zachód i w ostateczności na północ (gdy już nie ma innej możliwości). Gracz zawsze powinien widzieć pole na którym stoi i osiem pól sąsiednich oraz wybierać drogę zgodnie z określonym priorytetem. Gdy gracz zabrnie

w ślepą uliczkę to powinien wrócić się do miejsca, z którego wychodzą niezbadane jeszcze drogi i wybrać kolejną drogę zgodnie z określonym priorytetem. Jeżeli gracz uda się w kierunku, który nie wynika z zastosowanego algorytmu to gra powinna rozpoczynać się od początku, a labirynt powinien być znów zakryty.

Program może zawierać następujące klasy:

* `Plate` - klasa reprezentująca obiekt pola labiryntu - ściana, droga lub zakryte pole (oparta na klasie `pygame.sprite.Sprite`), z takimi metodami jak `__init__()` oraz `draw()`;
* `Player` - klasa, która reprezentuje obiekt gracza szukającego wyjścia z labiryntu (oparta o klasę `pygame.sprite.Sprite`);
* `Game` - główna klasa odpowiadająca za całą rozgrywkę z takimi metodami jak `__init__()`, `draw()`, `update()`, metodą obsługującą zdarzenia oraz metodami pomocniczymi;
* `Text` - klasa odpowiadając za obiekty tekstu wyświetlane na ekranie.

Pisząc program można wykorzystać grafiki umieszczone na platformie Moodle. Poniżej przykładowy screen gry: