|  |  |
| --- | --- |
|  | **AKADEMIA GÓRNICZO-HUTNICZA** |

Dokumentacja do projektu

**Zuzomaze**

z przedmiotu

**Języki programowania obiektowego**

Elektronika i Telekomunikacja rok 3

*Zuzanna Dąbrowa*

piątek 12:15

prowadzący: Rafał Frączek

08.01.2022

# Opis projektu

Celem projektu jest utworzenie narzędzia do losowego generowania labiryntów, oraz wyszukiwania najkrótszej ścieżki w nowopowstałym labiryncie. W celu utworzenia projektu posłużono się jednym z najpopularniejszych algorytmów iteracyjnych do generowania labiryntu o nazwie ‘Randomized depth-first search ‘, oraz algorytmem Breadth-first search w celu odnalezienia rozwiązania wygenerowanego labiryntu. Po uruchomieniu projektu można wygenerować obrazek z wizualizacją labiryntu oraz jego rozwiązania.

# Project description

The main goal of this project was implemetation of maze-generating and solving tool using two popular algorithsm – Ramdomized depth-first search (iterative version), and BFS. In result the program generates image with randomly generated maze and its solution.

# Instrukcja użytkownika

W tym punkcie należy umieścić instrukcję użytkowania programu. Może być to na przykład opis poszczególnych menu w programie. W przypadku gry należy opisać zasady gry. Opcjonalnie można wstawić zrzuty ekranu. Jeśli uruchomienie programu wymaga wykonania jakiś niestandardowych lub dodatkowych czynności (na przykład uruchomienie serwera baz danych itp.) to należy zamieścić tę informację.

# Kompilacja

Python jest językiem interpretowanym, zatem potrzebny jest dowolny interpreter Python 3 (najlepiej dla wersji 3.6 lub wyższej).

Aby uruchomić program należy przejść do katalogu z projektem, a następnie wywołać go poleceniem

python main.py

W rezultacie w ścieżce projektu zostanie zapisany wygenerowany obrazek.

# Pliki źródłowe

Projekt składa się z następujących plików źródłowych:

* *Cell.py* – deklaracja oraz implementacja klasy Cell,
* *Maze.py* – deklaracja oraz implementacja klasy Maze,
* *Consts.py* – deklaracja stałych używanych w całym projekcie.
* *Main*.py – główny plik programu zawierający dodatkowe funkcje generujące wynikowy obrazek. Służy do uruchomienia programu.

# Zależności

W projekcie wykorzystano następujące dodatkowe biblioteki (które powinny być wbudowane razem z najnowszą wersją Pythona, jednak podaję je w razie ewentualnych problemów):

* Python Imaging Library (PIL) Pillow – biblioteka służąca do przetwarzania obrazów. Strona internetowa: https://pypi.org/project/Pillow.

# Opis klas

W tym punkcie należy umieścić opis wszystkich stworzonych w projekcie klas. Należy podać do czego służy dana klasa oraz informację o jej publicznych metodach. Opcjonalnie można załączyć fragmenty kodu źródłowego. Na przykład:

W projekcie utworzono następujące klasy:

* Cell – reprezentuje pojedyńczą komórkę w labiryncie.
  + Klasa nie posiada metod, jedynie pola które ją charakteryzują.
* Maze – klasa zawierająca publiczne metody i zmienne.
  + Display\_cells(self) – Wyświetla informacje o każdej komórce w labiryncie,
  + Check\_neighbours(self,current) – Funkcja pomocnicza, która jest częścią iteracyjnego algorytmu reverse backtracking. Znajduje sąsiadów aktualnie podanej komórki (current) i zwraca ich listę.
  + Generate(self) – funkcja generująca labirynt. Usuwa losowo ściany poszczególnym komórkom tworząc w ten sposób labirynt.
  + Get\_cells(self) – zwraca listę komórek w labiryncie
  + Randomize\_endpoints(self) – zwraca id (w zasadzie indeksy) komórek, które mają być oznaczone jako początek oraz koniec labiryntu.
  + Find\_bfs\_nefighbours (self, current) – sprawdza i zwraca listę sąsiadów aktualnej komórki, do których możemy przejść (czyli sprawdzamy, czy sąsiedzi aktualnej komórki, oraz komórka posiadają ścianę z którejś ze stron)
  + Find\_path(self) – implementacja algorytmu BFS. Zwraca prawdę, jeśli udało się odnaleźć drogę do wyjścia. Dodatkowo oznacza komórki, które prowadzą do wyjściowej trasy. W przeciwnym wypadku zwraca fałsz.

# Zasoby

* brak

# Dalszy rozwój i ulepszenia

Podstawowym zamysłem projektu było wygenerowanie labiryntu oraz utworzenie gry 3D ( z użyciem silnika Ursina) z perspektywy pierwszej osoby, jednak ze względu na ograniczony czas podstawowe założenia projektu uległy zmianie, nie oznacza to jednak że nie będzie on dalej rozwijany.

Kolejnym pomysłem rozwoju projektu, jest implementacja kolejnych algorytmów losowo generujących labirynty i porównanie ich ze sobą.

# Inne

Wykorzystane informacje i przydatne źródła:

<https://en.wikipedia.org/wiki/Breadth-first_search>

<https://en.wikipedia.org/wiki/Maze_generation_algorithm>

<https://helion.pl/ksiazki/algorytmy-ilustrowany-przewodnik-aditya-bhargava,algoip.htm#format/e>

<https://pypi.org/project/Pillow/>

https://en.wikipedia.org/wiki/Python\_Imaging\_Library