

Projekt – zapalki

Cel i opis projektu

Temat projektu to gra w zapalki (spotykana bardzo często właśnie na pudełkach zapalek). Zasady są bardzo proste, całość gry polega na przesunięciu jednej lub kilku zapalek, aby równanie przedstawione z nich było poprawne. Program posiada interfejs tekstowy, każda zapalka w równaniu ma przyporządkowany numer, który umożliwia jej przesunięcie. Kolorem niebieskim oznaczone są zapalki, które możemy przesunąć, zaś białym „puste miejsca”.

Użytkownik ma do wyboru trzy wbudowane poziomy trudności: łatwy, średni oraz trudny. Poziomy łatwy charakteryzują się tym, że mamy do przesunięcia jedną zapalkę i musimy utworzyć z podanej cyfry inną. Poziomy średni działają na podobnej zasadzie, również mamy do przesunięcia jedną zapalkę, ale musimy zrobić to w taki sposób aby całe równanie było poprawne. W poziomach trudnych do przesunięcia są dwie zapalki. Do programu została dodana jeszcze jedna ciekawa opcja, mianowicie użytkownik może tworzyć swoje zagadki i zapisywać je w pliku json. Tworzenie poziomu polega na wprowadzeniu przez użytkownika dowolnego równania, po czym wypisze się ono w oknie konsoli. Użytkownik może wybrać jedną zapalkę, którą chce przesunąć oraz miejsce dla niej. Jeśli wszystko zostanie wprowadzone poprawnie zostaniemy zapytani czy aby na pewno chcemy dodać tę zagadkę do gry. Po akceptacji zostanie ona dodana do pliku z zagadkami gracza oraz będziemy mogli w nią zagrać wybierając opcję „Play levels created by the user” znajdującą się w MENU głównym programu pod numerem 2.

Budowa projektu

Projekt został podzielony na trzy katalogi: data, game oraz tests. Oddzielnie został umieszczony moduł main.py służący do uruchomienia programu. W katalogu „data” znajdują się dwa pliki json. W levels.json są przechowywane poziomy wbudowane, w które możemy zagrać wybierając odpowiednią opcję w naszej grze, są one zweryfikowane i odpowiedzi do każdego z nich faktycznie mają sens. Inaczej to wygląda w przypadku pliku user_levels.json, właśnie tam są przechowywane poziomy użytkownika gdy przy wywołaniu nie poda on ścieżki do pliku ze swoimi poziomami lub nie uda się z niego odczytać listy poziomów. Zawartość tego pliku jest modyfikowana w trakcie gry, już na samym początku gracz zostaje zapytany czy chce wyczyścić listę poziomów znajdującą się w nim. Dzięki temu rozwiązaniu mamy wybór: możemy zacząć grę z pustą listą poziomów stworzonych przez użytkownika i dopisywać do niej poziomy w trakcie rozgrywki lub zacząć grę z dodatkowymi poziomami stworzonymi przez nas poprzednio. W repozytorium ten plik zawiera domyślnie pustą listę poziomów. W katalogu „game” znajdują się najistotniejsze moduły, które stanowią podstawowe funkcjonalności programu. Są to:

- errors.py – znajdują się tam wyjątki obsługiwane przez program.
- interface.py – najważniejszy moduł, znajduję się tam klasa Interface, która steruje funkcjami programu na podstawie tego co poda użytkownik. Na wejściu przyjmując ścieżkę do pliku z poziomami gracza oraz listy z poziomami. Odpowiednie metody są wywoływane rekurencyjnie.

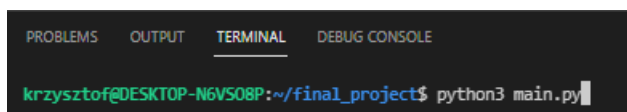
- `interface_functions.py` – znajdują się tam funkcje „wspierające” interfejs, są odpowiedzialne za poszczególne działania w programie.
- `levels_classes.py` – znajdują się tam klasy pozwalające na utworzenie poziomów, główna klasa to `Level`, a dziedziczą po niej odpowiednio `EasyAndMediumLevel` oraz `HardLevel`. Każda przyjmuje parametry szczególne dla każdego poziomu, takie jak: polecenie, równanie, odpowiedzi oraz liczbę zapalek do przesunięcia.
- `levels_io.py` – znajdują się tam funkcje odpowiedzialne za czytanie poziomów z plików.
- `making_ascii_equation.py` – moduł zawiera funkcje, które są odpowiedzialne za utworzenie równania w postaci ASCII ART, oraz ważną funkcją, która sprawdza czy takie działanie jest w ogóle możliwe.

W „tests” zostały umieszczone testy jednostkowe weryfikujące działanie poszczególnych funkcji.

Instrukcja użytkownika

- Program należy uruchamiać z poziomu, katalogu głównego
- Istnieją dwie możliwości uruchomienia programu:

Sposób I – bez podania ścieżki do pliku na wejściu



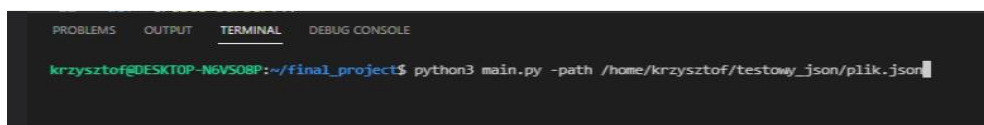
```

PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
krzysztof@DESKTOP-N6VS08P:~/final_project$ python3 main.py

```

Przy takim wywołaniu poziom stworzone przez nas będą domyślnie zapisywane oraz odczytywane z pliku `user_levels.json` znajdującego się w katalogu `data`. Przy każdym wywołaniu możemy wyczyścić listę z poziomami, zostaniemy o to zapytani na samym początku.

Sposób drugi II – z podaniem ścieżki do pliku na wejściu poprzez `-path`



```

PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
krzysztof@DESKTOP-N6VS08P:~/final_project$ python3 main.py -path /home/krzysztof/testowy_json/plik.json

```

Przy takim wywołaniu poziom użytkownika będą zapisywane oraz odczytywane z pliku podanego na wejściu. Musi on zawierać poprawne dane, z których odczytem nasz program sobie poradzi. Powinien on zawierać obiekt json z kluczem „`created_by user`” a jako wartość powinna zostać podana lista poziomów (może być pusta). Każdy poziom również powinien być obiektem zawierającym określone klucze. Niżej zamieszczam kilka przykładów plików, z których nasz program jest w stanie odczytać dane.

- Poruszanie się po programie

Interfejs programu jest bardzo prosty w MENU głównym mamy do wyboru opcje oznaczone numerami 1-4 i jesteśmy proszeni o wybranie jednej z nich, możemy tutaj zdecydować czy chcemy zagrać w poziom wbudowane, stworzone przez użytkownika lub stworzyć własny poziom. W każdym menu jest stosowany tożsamy system komunikacji z użytkownikiem. Gdy użytkownik wybierze opcję spoza dostępnych (np.

podane są opcje, od 1 do 4, a użytkownik wybierze 5), w większości przypadków zostanie on przekierowany do MENU głównego, skąd może opuścić program wybierając opcje 4. QUIT.

```
{ } plikjson x
{ } plikjson > ...
1  {
2    "created_by_user": [
3      {
4        "equation": "1+2=3",
5        "task": "Move one match to make the equation correct",
6        "first_move": "1",
7        "second_move": "2"
8      }
9    ]
10  }
```

```
{ } plikjson x
{ } plikjson > ...
1  {
2    "created_by_user": [
3      {
4      }
5    ]
6  }
```

```
{ } plikjson x
{ } plikjson > ...
1  {
2    "created_by_user": [
3      {
4        "equation": "1+2=3",
5        "task": "Move one match to make the equation correct",
6        "first_move": "1",
7        "second_move": "2"
8      },
9      {
10       "equation": "2+2=4",
11       "task": "Move one match to make the equation correct",
12       "first_move": "2",
13       "second_move": "3"
14     }
15   ]
16 }
```

Podsumowanie

Projekt, pomimo z pozoru prostego tematu, stanowił dość duże wyzwanie i był bardzo czasochłonny, problematyczne było zbudowanie podstawowych funkcji przekształcających równanie podane jako pythonowy string na tzw. ASCII ART oraz jego reprezentacja jako kolorowe cyfry w postaci podobnej jak na wyświetlaczu 7-mio segmentowym. Ostatecznie udało się osiągnąć zamierzony efekt. Problemem występującym w całej grze jest jednak weryfikacja poziomów tworzonych przez gracza. Nie udało mi się stworzyć algorytmu, który sprawdzi, czy równania stworzone przez użytkownika faktycznie mają sens i są poprawne matematycznie. Wynikało to z jego złożoności oraz stopnia trudności. Mimo wszystko, myślę, że dodawanie swoich poziomów do gry to bardzo ciekawa funkcjonalność mojego programu i tworzenie skomplikowanych zagadek może dostarczyć wiele satysfakcji użytkownikom.