Projekty z przedmiotu Wstęp do programowania

Opracowała: Laura Grzonka, na podstawie materiałów Grzegorza Madejskiego

PLAN DZIAŁANIA

Na przygotowanie projektu będziecie mieć kilka tygodni :) Plan działania wygląda tak:

1. Wybieracie temat projektu 🚀
2. W repozytorium poświęconym zajęciom z WdP na GitLabie tworzycie nowy branch o nazwie *Projekt*. W nim tworzycie plik readme.md z tytułem i krótkim opisem projektu, a także instrukcją obsługi gry/aplikacji 📄
3. Pracujecie nad projektem 💻
4. Prezentujecie swoje gry/aplikacje 🎉

CEL

Celem projektu jest stworzenie:

- prostej gry komputerowej lub ewentualnie
- innej prostej aplikacji czy
- strony internetowej.

W dwóch ostatnich przypadkach proszę najpierw o kontakt ze mną.

Zasady:

- **Projekty tworzycie samodzielnie.**
- Proszę nie robić najbardziej oklepanych gier, takich jak kółko i krzyżyk czy wąż (*Snake*) w najprostszej wersji. Jeśli chcecie nad takimi pracować, to zmodyfikujcie i rozwińcie zasady, tak by gra była oryginalna i ciekawsza. Najprostsze, nierozwinięte wersje takich gier mogą zostać ocenione negatywnie.
- Poza tym wybór gry jest dowolny: platformówki, RPG, wyścigi, gry strategiczne – do wyboru, do koloru 🎯♟️🎮
- Gra powinna być **interaktywna** – tzn. gracz steruje w jakiś sposób grą (np. myszką lub naciskając klawisze na klawiaturze).
- W uzasadnionych przypadkach można stworzyć grę nieinteraktywną – wówczas powinna ona zawierać inny trudniejszy element, np. przebiegu gry uczy się algorytm uczenia maszynowego. W takich wypadkach proszę najpierw o skonsultowanie pomysłu ze mną.
- Gra, oprócz trybu jednoosobowego, może, ale nie musi, uwzględniać tryb wieloosobowy – tzn. kilku graczy rywalizuje, ale wygrywa jeden. Oczywiście wybór trybów bardzo zależy od samej gry (kółko i krzyżyk jest z natury wieloosobowa, a sudoku – jednoosobowa).
- Gra może wykorzystywać pliki do zapisywania ustawień, zagadek, haseł lub najlepszych wyników. Najlepsze wyniki mogą być zapisane w postaci słownika lub tabelki csv.
- Gra może mierzyć czas, który jest uwzględniony w najlepszych wynikach. W najlepszych wynikach mogą też być przetrzymywane punkty.
- Gra może, ale nie musi, mieć kolejne poziomy, do których się przechodzi po osiągnięciu konkretnej liczby punktów.

- Gra może być tekstowa/konsolowa. Jeśli ktoś jest bardziej ambitny, może pójść w kierunku gry z interfejsem graficznym, lub wykorzystującej obiekty i klasy. W takim przypadku pomocne może być np. skorzystanie z biblioteki pygame. Nie jest to jednak wymagane i wykracza poza zakres materiału przedmiotu.
- Jeśli ktoś zamiast gry, chce stworzyć aplikację użytkową, to jest to możliwe, pod warunkiem, że aplikacja będzie miała odpowiednio wiele ciekawych funkcji. Proszę najpierw skonsultować ze mną swój pomysł.

WYMAGANIA TECHNICZNE

- Piszemy w języku Python. Można – zachęcam i polecam! – korzystać z bibliotek pythonowych.
- Kod gry powinien być przejrzysty: zmienne i funkcje powinny mieć sensowne nazwy. Projekt należy w sensowny sposób podzielić na funkcje, z których każda powinna być opatrzona komentarzem wyjaśniającym jaka jest jej rola. (Patrz: PEP8).
- Do gry proszę dołączyć krótką instrukcję obsługi: jak uruchomić? jakie są zasady? jak grać? Powinna się ona znajdować w pliku readme na GitLabie.

A CO Z TYM AI?

W trakcie realizacji projektu możecie korzystać z samouczków i pomocy AI. Warunki:

- **Warunkiem zaliczenia jest wykonanie większości pracy samodzielnie.**
- W każdym wypadku proszę wyraźnie wskazać, jaką część pracy wykonaliście samodzielnie, jaką skopiowaliście z samouczków, a jaką opracowaliście z pomocą chatbota. Proszę nie opowiadać o tym w trakcie prezentacji na koniec semestru, lecz zawrzeć taką informację na slajdach/w readme/w dodatkowym pliku w repozytorium. Wskażcie, które elementy były własne, oryginalne i wymagały większej pracy/wiedzy/umiejętności programistycznych.
- Każde źródło proszę zacytować w bibliografii (ponownie: na slajdach/w readme/w dodatkowym pliku w repozytorium) wymieniając:
    - W przypadku źródeł internetowych:
        - Tytuł artykułu/strony, autora_kę, jeżeli jest podany_a, adres URL i **koniecznie** datę dostępu, czyli ostatnią datę, kiedy korzystaliście z danej strony.
    - W przypadku AI:
        - Nazwę i **numer** modelu, nazwę firmy, która go stworzyła, datę dostępu. Dotyczy to również grafik wygenerowanych za pomocą modeli AI.
    - W przypadku bibliotek i frameworków:
        - Nazwę i wersję biblioteki i wszelkie inne niezbędne informacje, które zazwyczaj można znaleźć na oficjalnej stronie internetowej (tak jak np. na stronie Matplotlib). Nie jest to potrzebne w przypadku biblioteki standardowej.

Brak lub błędne cytowanie źródeł wpłynie na ocenę projektu.

Proszę pamiętać, że choć sztuczna inteligencja rozwija się obecnie w niesamowitym tempie, daleko jej jeszcze do bycia bezbłędną. Wbrew pozorom, AI to nie jest wyrocznia ;) Zawsze weryfikujcie informacje, w tym kod, który otrzymujecie. Pamiętajcie, że ostatecznie to **Wasza odpowiedzialność**, w jaki sposób korzystacie ze źródeł – a także **Wasza ocena** :)

TEMAT GRY

Każdy może wybrać grę jaką chce stworzyć. Jeśli macie jakieś ciekawe pomysły, proszę o zgłoszenie mi tego.

Przykładowe źródła inspiracji:

- **Kurnik.pl - strona z grami**. Można się przyjrzeć tym grom i zrobić coś podobnego (może być uproszczona wersja). Przykłady: gra w kości, gomoku. Gry w karty są chyba zbyt problematyczne, ale może prostą grę też dałoby radę zrobić.
- **Teleturnieje.** W telewizji jest sporo teleturniejów, które można zasymulować w Pythonie np. "Koło fortuny", "Postaw na milion", "Milionerzy", "Familiada". Być może nawet można przygotować prymitywną wersję "Jaka to melodia", bo w Pythonie są paczki do generowania dźwięków (można parę nut piosenki wygenerować).
- **Łamigłówki z gazet**. W gazetach można znaleźć krzyżówki, sudoku, wykreślanki, obrazki logiczne (nonogramy).
- **Gry animowane z planszą**. Te gry najtrudniej zasymulować w środowisku tekstowym na konsoli i być może lepszym rozwiązaniem będzie skorzystanie ze środowiska graficznego (np. dzięki paczce pygame). Gry jakie przychodzą do głowy to "Wężyk", "Pacman" (lub zwykłe szukanie drogi w labiryncie - im szybciej tym lepiej), proste strzelanki.

Przykładowy rozszerzony opis dla gry "Koło fortuny", jaki można zawrzeć w readme.md:

Gra symuluje teleturniej "Koło fortuny" w konsoli pythonowej. Przewidziana jest dla 2-4 graczy. Liczbę graczy wraz z ich nickami podajemy na starcie programu. Gracze naprzemiennie zgadują litery w haśle. Gracz w swojej kolejce:

1. zawsze kręci kołem (losuje stawkę pieniędzy) - losowana jest stawka z listy;
2. podaje spółgłoskę (każda odkryta spółgłoska dodaje stawkę z koła do stanu konta gracza) lub kupuje samogłoskę (jednokrotnie odjęta jest stawka z koła, opcja dostępna, gdy mamy odpowiednią ilość pieniędzy) - musi podać którą opcję wybiera;
3. opcjonalnie: może podać hasło.

Gracz powtarza kolejkę tak długo, jak poprawnie zgaduje litery. W przypadku skuchy (albo wylosowania bankruta czy stopu na kole), inicjatywę przejmuje kolejny gracz.

Gra trwa ustaloną liczbę rund/haseł do odgadnięcia (np. 3). Po każdej rundzie ilość pieniędzy jest zapisywana, przez co nie może być wyzerowana poprzez wylosowanie bankruta. Zwycięża gracz z największą ilością pieniędzy po 3 rundach. Pula zgromadzonych pieniędzy to wynik punktowy, który może być dopisany do listy najlepszych 10 wyników w osobnym pliku tekstowym.

Listę haseł znajduje się w pliku tekstowym csv, wraz kategorią (np. historia, powiedzenia, geografia, itp.) Hasło jest losowane spośród podanych i zapisane w wygwiazdkowanej formie np. ***** ***, dodatkowo podana jest kategoria hasła.

OCENA PROJEKTU

W ocenie projektu będę brała pod uwagę takie kryteria, jak:

- Jakość i poziom realizacji projektu: czy gra/aplikacja dobrze chodzi, czy jest dobrze przemyślana, zawiera wszystkie niezbędne i istotne elementy i funkcjonalności (a jeżeli jakieś dodatkowe, to jeszcze lepiej) – im wyższe, tym lepiej;
- Wasz wkład w projekt: czas, energia, własny kod programistyczny, ciekawe pomysły/rozwiązania – im większy, tym lepiej 
- Poziom zrozumienia kodu, programu i sposobu jego funkcjonowania;
- Oryginalność projektu (w mniejszym stopniu);
- Prezentacja projektu: czy była jasna, logicznie i przejrzyście uporządkowana, ciekawa, czy zmieściliście się w czasie;
- Jakość dokumentacji: czy wszystkie funkcjonalności i sposób użytkowania gry/aplikacji zostały opisane, czy zacytowano źródła, itp.

PREZENTACJA

- Na prezentację będziecie mieli **po 5 minut na osobę**. To mało, ale grupy są duże, więc to jedyne wyjście :)
- Projekt będziecie przedstawiać na forum grupy. Po prezentacji będzie czas na jedno-dwa pytania.
- Można przygotować prezentację w PowerPoincie lub podobnym programie, ale można też zostać przy pokazaniu kodu oraz samej gry/aplikacji.

LINKI

Czysty i przejrzysty kod:

- https://peps.python.org/pep-0008/
- https://realpython.com/python-pep8/
- https://kamil.kwapisz.pl/gramatyka-dla-programisty/