Projekt ze wstępu do programowania

19 grudnia 2024

1 Terminy i prezentacja

Do 2.01.2025 należy wybrać temat projektu (tytuł projektu i krótki opis/plan), założyć brancha (nową gałąź) o nazwie "projekt" na swoim repozytorium ze "wstępu do programowania" i zedytować plik readme.md na branchu, tak by zawierał opis gry/aplikacji.

Projekty należy ukończyć do dnia 29.01.2025, 23:59. Na zajęciach dnia 30.01.2025, projekty zostaną ocenione. Można także oddać projekt pod ocenę na zajęciach tydzień wcześniej.

2 Cel

Celem projektu jest stworzenie prostej gry komputerowej. Generalnie wybór gry jest dowolny. Proszę jednak nie robić gier, które są mocno znane i dostępne w Internecie ("Kółko i krzyżyk" w podstawowej wersji, "Wężyk" w typowej wersji). Jeśli wybierzemy grę tego typu, to modyfikujemy jej zasady by była inna i bardziej oryginalna. Jest kilka rzeczy, które są mile widziane w grze i poprawią punktację za projekt:

- Gra powinna być interaktywna - tzn. gracz steruje w jakiś sposób grą (np. naciskając klawisze na klawiaturze).
- Gra, oprócz trybu jednoosobowego, może 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 powinna 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 być tekstowa/konsolowa. Jeśli ktoś jest bardziej ambitny, może pójść w kierunku gry z interfejsem graficznym, lub wykorzystującej obiekty/klasy (np. z wykorzystaniem pygame), nie jest to jednak wymagane i wykracza poza zakres materiału przedmiotu. W przypadku gier tworzonych za pomocą pygame, proszę wskazać, które

elementy były skopiowane z samouczków, a które były własne, oryginalne i wymagały większej pracy programistycznej.

- Kod gry powinien wykorzystywać jak najwięcej tego, o czym była mowa na zajęciach, w szczególności słowniki, wyjątki, operacje na plikach.
- w czasie pisania kodu powinno się korzystać z testów jednostkowych (temat testów będzie jeszcze na zajęciach.

Jeśli ktoś zamiast gry, chce stworzyć aplikację użytkową, to jest to możliwe, pod warunkiem, że aplikacja będzie miała odpowiednio wiele ciekawych funkcji i pod warunkiem, że poinformujesz o tym prowadzącego zajęcia.

3 Wymagania techniczne

- Piszemy w języku Python. Można korzystać z bibliotek pythonowych.
- Kod gry powinien być przejrzysty: zmienne i funkcje powinny mieć sensowne nazwy. Projekt należy w sensowny sposób podzielić na funkcje. Każda funkcja powinna być opatrzona komentarzem wyjaśniającym jaka jest jej rola.
- Do gry trzeba dołączyć, krótką instrukcję obsługi: jak uruchomić? jakie są zasady? jak grać? Np. wewnątrz pliku readme (na gitlabie).

4 Gitlab

Projekt należy przetrzymywać na portalu gitlab.com w naszym repozytorium w osobnym branchu projekt. Plik pythonowy z grą, wszystkie pliki tekstowe związane z grą oraz plik z instrukcją obsługi trzeba umieścić na repo w terminie podanym powyżej. Projekt na gitlabie powinniśmy systematycznie rozwijać wprowadzając nowe commity opatrzone sensownymi komentarzami. Systematyczność też będzie brana pod uwagę przy ocenianiu.

5 Temat gry

Każdy może wybrać grę jaką chce stworzyć. Gdzie można czerpać inspiracje:

- Kurnik.pl - strona z grami. Można się przyjrzeć tym grom i zrobić coś podobnego (może być uproszczona wersja). Przykłady: gra w kości, gomoku.
- Teleturnieje. W telewizji jest sporo teleturniejów, które można zasymulować w Pythonie np. “Koło fortuny”, “Postaw na milion”, “Milionerzy”, “Familiada”. Być może nawet można zrobić 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 popularny teleturniej "Koło fortuny" w konsoli pythonowej. Przewidziana jest dla 2-4 graczy (liczbę graczy z nickami podajemy na starcie programu). Gracze naprzemiennie zgadują litery w haśle. Gracz w swojej kolejce: zawsze kręci kołem (losuje stawkę pieniędzy) - losowana jest stawka z listy podaje spółgłoskę (każda odkryta 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 opcjonalnie: może podać hasło Gracz powtarza kolejkę tak długo, jak 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 pieniądze są zapisane i nie mogą być wyzerowane przez wylosowanie bankruta. Zwycięża gracz z największa liczbą 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.