Podstawy programowania
Projekt zaliczeniowy

1 Zasady oceniania

Do wykonania jest jeden wybrany projekt. Ocena za zadanie zależy od tego w jakim stopniu zostały spełnione wymienione wymagania. Plik źródłowy programu proszę przesłać na platformie **Moodle**. **Plik musi zawierać numer albumu**. Projekt może być rozbity na kilka plików jeżeli jest to potrzebne do jego działania.

UWAGA ODNOŚNIE OCENIANIA: Funkcjonalności oznaczone gwiazdką (*) są minimalnym wymogiem na uzyskanie oceny pozytywnej.

UWAGA: Termin oddania zadania jest ustawiony w systemie moodle. W przypadku nie oddania zadania w terminie, nie ma możliwości późniejszego przesłania. Po terminie ustawionym na Moodle zadanie rozliczane będzie w trybie indywidualnym na zajęciach lub po umówieniu się z prowadzącym.

UWAGA: W przypadku wysłania zadania w formie niezgodnej z opisem w instrukcji prowadzący zastrzega prawo do wystawienia oceny negatywnej za taką pracę.

2 Baza studentów

Do napisania jest program imitujący bazę danych studentów. Program musi posiadać niżej wymienione funkcjonalności:

- **Zarządzanie bazą***: Umożliwienie dodawania i usuwania studentów, gdzie przechowujemy imię, nazwisko i numer albumu.
- **Formatowanie wyświetlania***: Możliwość wyświetlenia danych studentów w postaci tabeli (użyć dowolnego sposobu formatowania albo znaleźć bibliotekę która to robi).
- **Wykorzystanie pliku**: Wczytanie bazy ze wskazanego pliku, bądź zapisanie do wskazanego pliku, sugerowany format to `.csv`
- **Historia**: Zapisywanie oraz możliwość wyświetlenia ostatnich wykonanych operacji (przechowywane powinno być minimum 10 ostatnich operacji, może być wyświetlana kolejność, bądź czas).
- **Możliwość oceniania**: Dodaj możliwość oceniania studenta w konkretnym przedmiocie. Możliwość dodania pojedynczej oceny, jeżeli student nie został oceniony w przypadku danego przedmiotu to można wyświetlić "X".

3 Konwerter walut

Opracowanie prostego, opartego na konsoli przelicznika walut, który pozwala użytkownikom na konwersję pomiędzy różnymi walutami w oparciu o bieżące kursy wymiany. Powinien zawierać następujące funkcjonalności:

- **Kursy walut***: Dołącz predefiniowaną listę kursów wymiany walut. Zezwalaj użytkownikom na aktualizację kursów wymiany w razie potrzeby.
- **Opcje konwersji***: Umożliwienie użytkownikom wyboru waluty źródłowej i docelowej do konwersji. Zaimplementuj opcje konwersji bezpośredniej i odwrotnej.

- **Walidacja danych wprowadzanych przez użytkownika**: Weryfikacja danych wprowadzanych przez użytkownika w celu zapewnienia dokładnego wyboru waluty i wprowadzenia kwoty. Wdrożenie obsługi błędów dla przypadków takich jak nieprawidłowe kody walut lub kwoty.
- **Historia**: Przechowuj historię ostatnich przeliczeń walut i wyświetlaj ją na żądanie.
- **Zapisywanie i wczytywanie kursów wymiany**: Pozwól użytkownikom na zapisywanie i wczytywanie niestandardowych kursów wymiany ze wskazanego pliku.

4 Kółko i krzyżyk

Stwórz konsolową implementację klasycznej gry kółko i krzyżyk. Gracze na zmianę umieszczają swoje znaczniki na siatce 3x3, dążąc do uzyskania zwycięskiej kombinacji. Program musi posiadać niżej wymienione funkcjonalności:

- **Wprowadzanie danych przez gracza***: Umożliwia graczom wprowadzanie ruchów poprzez określenie wiersza i kolumny. Weryfikuj dane wejściowe, aby upewnić się, że mieszczą się w prawidłowym zakresie.
- **Wizualne aktualizacje planszy***: Akutalizuj stan wyświetlanej w konsoli planszy w celu odzwierciedlenia aktualnego stanu gry po każdym ruchu.
- **Opcja ponownej gry**: Po zakończeniu gry gracze mogą zagrać ponownie.
- **Warunki zwycięstwa/przegranej/remisu**: Określanie warunków wygranej, przegranej lub remisu na podstawie stanu planszy. Wyświetl wynik na koniec gry.
- **Historia rozgrywki**: Zezwól graczom na wprowadzenie swoich imion / nick'ów i w przypadku ponownej gry wyświetlaj wynik między graczami i zapisuj wynik do pliku. *(historia w pliku ma być utrzymywana cały czas, tj. jeżeli program zostanie wyłączony i włączony ponownie, a rozgrywkę będzie rozgrywała ta sama para osób, to wynik dotychczasowych rozgrywek również powinien się wyświetlać)*

5 Tekstowa gra przygodowa

Stwórz interaktywną, tekstową grę przygodową, w której gracze przechodzą przez serię scenariuszy i podejmują decyzje wpływające na wynik historii (historię można wygenerować przy wykorzystaniu np chat-gpt, bądź napisać samodzielnie). Gra powinna zawierać następujące funkcjonalności:

- **Pobieranie danych wejściowych od gracza***: Użyj danych wejściowych użytkownika, aby umożliwić graczom podejmowanie decyzji w krytycznych momentach fabuły. Zaimplementuj system walidacji i obsługi różnych wyborów gracza. Uwzględnij wiele rozgałęziających się scenariuszy i możliwych zakończeń.
- **Zdarzenia losowe***: Wprowadź losowe wydarzenia lub spotkania, aby dodać nieprzewidywalności do gry (można wykorzystać bibliotekę na przykład `random`).
- **Interakcja postaci**: Przedstaw postacie niezależne (NPC), z którymi gracze mogą wchodzić w interakcje. Uwzględnij dialogi i punkty decyzyjne, które wpływają na relacje z postaciami niezależnymi.
- **System ekwipunku**: Wdrożenie podstawowego systemu ekwipunku, w którym gracze mogą zbierać i wykorzystywać przedmioty w trakcie gry.
- **Zapisz i wczytaj grę**: Umożliwienie graczom zapisywania postępów i wznawiania gry później. Wdrożenie funkcji wczytywania gry w celu odzyskania zapisanych postępów.
- **Grafika (OPCJONALNIE)**: spróbuj wykorzystać grafikę ASCII w celu poprawy wrażeń wizualnych.