Scenariusz lekcji w szkole ponadpodstawowej na kwalifikacji INF. 04 z przedmiotu: Projektowanie, programowanie i testowanie aplikacji

Temat: Pythonowe gry - koduj, twórz i współpracuj!

Cele lekcji:

1. Uczniowie potrafią zaprojektować i zaimplementować prostą grę w Pythonie.
2. Uczniowie rozwijają umiejętności współpracy w grupie poprzez podział ról i odpowiedzialności.
3. Uczniowie rozwijają umiejętności programistyczne w zakresie używania pętli, instrukcji warunkowych i funkcji.

Czas trwania: 90 minut (2 lekcje 45-minutowe)

Materiały i narzędzia:

- Komputery z zainstalowanym Pythonem.
- Projektor lub tablica interaktywna.
- Przygotowany kod bazowy (szkielet gry).
- Arkusze z opisem ról w zespole i checklistą zadań.
- Karty pracy z instrukcjami.
- Przykładowe rozwiązania do gier.

. Wprowadzenie (15 minut)

1. **Powitanie i wprowadzenie tematu lekcji:**
    - Nauczyciel wyjaśnia, że celem lekcji jest stworzenie prostych gier w Pythonie (np. "Zgadnij liczbę", "Rzut kostką", "Kamień, papier, nożyce", „Koło fortuny”), z uwzględnieniem współpracy w grupach.
    - Krótkie przypomnienie podstaw Python: funkcja random, pętle, instrukcje warunkowe, wczytywanie danych od użytkownika.

2. **Przykładowa demonstracja gry:**
    - Nauczyciel pokazuje działający przykład gry "Zgadnij liczbę": komputer losuje liczbę, a gracz próbuje ją odgadnąć na podstawie wskazówek "za dużo" lub "za mało".
    - Wskazuje możliwości rozbudowy gry: np. dodanie licznika prób.
    - Cel gry**:** Komputer losuje liczbę od 1 do 50, a gracz zgaduje liczbę, otrzymując wskazówki "za dużo" lub "za mało".

```python
import random

def zgadnij_liczbe():
    liczba = random.randint(1, 50)
    proby = 0

    print("Zgadnij liczbę od 1 do 50")

    while True:
        proby += 1
        odpowiedz = int(input("Twoja odpowiedź: "))

        if odpowiedz < liczba:
            print("Za mało!")
        elif odpowiedz > liczba:
            print("Za dużo!")
        else:
            print(f"Brawo! Zgadłeś w {proby} próbach.")
            break

zgadnij_liczbe()
```

3. **Podział na grupy:**
    - Uczniowie dzielą się na grupy 3-4 osobowe.
    - Każda grupa wybiera lub losuje swoją grę do realizacji: "Rzut kostką", "Kamień, papier, nożyce", „Koło fortuny”.

2. Przydział ról w grupach (10 minut)

Każda grupa przydziela swoim członkom następujące role:

1. **Lider projektu:**
    - Koordynuje pracę zespołu, dba o realizację zadań zgodnie z harmonogramem.
    - Pilnuje czasu i rozdziela zadania między członków grupy.
2. **Główny programista:**
    - Pisze główną część kodu gry.
    - Konsultuje się z zespołem w kwestiach związanych z logiką gry.
3. **Tester:**
    - Na bieżąco sprawdza poprawność działania programu.
    - Wykrywa i zgłasza błędy do naprawy.
    - Proponuje ulepszenia w grze.
4. **Dokumentalista:**
    - Sporządza dokumentację projektu (np. opis gry, decyzje zespołu).
    - Przygotowuje materiały do prezentacji projektu.
    - Dodaje komentarze w kodzie, aby był czytelny.

3. Realizacja projektu (45 minut)

1. **Praca nad grą:**
    - Lider projektu rozpoczyna pracę, przypominając o podziale zadań.
    - Główny programista tworzy kod bazowy w oparciu o otrzymane wskazówki.
    - Tester analizuje poszczególne fragmenty kodu na bieżąco, zgłaszając uwagi.
    - Dokumentalista opisuje działanie programu, wprowadza komentarze do kodu i zbiera materiały do prezentacji.
2. **Wsparcie nauczyciela:**
    - Nauczyciel obserwuje pracę grup, wspiera w rozwiązywaniu problemów technicznych i międzyludzkich.
3. **Karty pracy:**
    - Każda grupa otrzymuje kartę pracy z zadaniami i wskazówkami do stworzenia wybranej gry.

Karta pracy: "Rzut kostką"

Cel gry: Komputer losuje wynik rzutu kostką (od 1 do 6), a gracz zgaduje wynik.

Instrukcje dla ról:

- **Lider projektu:**
  - Zaplanuj dodanie systemu punktacji (np. liczba poprawnych zgadnięć).
  - Ustal harmonogram testowania gry.
- **Główny programista:**
  - Napisz funkcję losującą wynik rzutu.
  - Dodaj mechanizm zgadywania wyniku przez gracza.
- **Tester:**
  - Sprawdź, czy wynik losowania jest zawsze w zakresie od 1 do 6.
  - Testuj grę kilkukrotnie, aby upewnić się, że punkty są poprawnie naliczane.
- **Dokumentalista:**
  - Przygotuj opis zasad gry do przedstawienia innym grupom.
  - Zaznacz w dokumentacji miejsca, gdzie kod może zostać rozbudowany.

```python
import random

def rzut_kostka():
    print("Rzut kostką - zgadnij wynik!")

    kostka = random.randint(1, 6)
    odpowiedz = int(input("Twoja odpowiedź (1-6): "))

    if odpowiedz == kostka:
        print("Brawo! Zgadłeś wynik.")
    else:
        print(f"Niestety, nie zgadłeś. Wynik to: {kostka}")

rzut_kostka()
```

Karta pracy: "Kamień, papier, nożyce"

Cel gry: Gracz wybiera jedną z opcji (kamień, papier, nożyce), komputer losuje swoją opcję, a gra wyłania zwycięzcę.

Instrukcje dla ról:

- **Lider projektu:**
  - Ustal kolejność implementacji: losowanie, porównywanie wyników, liczenie punktów.
  - Dopilnuj, aby każdy w zespole znał zasady gry.
- **Główny programista:**
  - Zaimplementuj funkcję losującą wybór komputera.
  - Dodaj warunki decydujące o wyniku gry (wygrana, przegrana, remis).
- **Tester:**
  - Sprawdź wszystkie możliwe kombinacje wyborów gracza i komputera.
  - Upewnij się, że wyniki są zgodne z zasadami gry.
- **Dokumentalista:**
  - Przygotuj opis zasad gry i przykładowe wyniki.
  - Dodaj komentarze, które wyjaśniają, jak gra decyduje o zwycięzcy.

```python
import random

def kamien_papier_nozyce():
    opcje = ["kamień", "papier", "nożyce"]
    komputer = random.choice(opcje)

    print("Wybierz: kamień, papier lub nożyce")
    gracz = input("Twój wybór: ").lower()

    if gracz == komputer:
        print(f"Remis! Obaj wybraliście {gracz}.")
    elif (gracz == "kamień" and komputer == "nożyce") or \
         (gracz == "papier" and komputer == "kamień") or \
         (gracz == "nożyce" and komputer == "papier"):
        print(f"Wygrałeś! {gracz} pokonuje {komputer}.")
    else:
        print(f"Przegrałeś! {komputer} pokonuje {gracz}.")

kamien_papier_nozyce()
```

Karta pracy: "Koło fortuny"

Cel gry: Komputer losuje słowo, a gracz próbuje odgadnąć poszczególne litery. Celem jest odgadnięcie całego słowa.

- **Lider projektu:**
  - Organizuje podział pracy w grupie i ustala etapy realizacji projektu.
  - Pilnuje, aby wszyscy członkowie grupy aktywnie uczestniczyli w realizacji projektu.
  - Koordynuje wybór słów do gry i opracowanie zasad losowania liter.
- **Główny programista:**
  - Tworzy kod gry, w tym logikę losowania słów i obsługę wprowadzania liter przez gracza.
  - Implementuje system śledzenia liczby prób oraz odkrywania liter.
  - Rozwiązuje problemy techniczne, takie jak wyświetlanie ukrytego słowa czy kończenie gry.
- **Tester:**
  - Sprawdza, czy gra działa poprawnie na różnych etapach (np. losowanie słów, odgadywanie liter).
  - Testuje różne przypadki, takie jak podanie litery nieobecnej w słowie lub wpisanie całego słowa zamiast pojedynczej litery.
  - Zgłasza błędy i proponuje usprawnienia, np. dodanie wskazówek.
- **Dokumentalista**:
  - Dodaje komentarze do kodu, aby wyjaśnić logikę działania gry.
  - Tworzy dokument opisujący grę, jej zasady, oraz instrukcję użytkowania.

```python
import random

def kolo_fortuny():
    slowa = ["python", "komputer", "programowanie", "edukacja"]
    slowo = random.choice(slowa)
    ukryte = ["_" for _ in slowo]
    proby = 6

    print("Witaj w grze Koło Fortuny!")

    while proby > 0 and "_" in ukryte:
        print("".join(ukryte))
        print(f"Pozostałe próby: {proby}")
        litera = input("Podaj literę: ").lower()

        if litera in slowo:
            for i in range(len(slowo)):
                if slowo[i] == litera:
                    ukryte[i] = litera
        else:
            proby -= 1
            print("Zła litera!")

    if "_" not in ukryte:
        print(f"Brawo! Odgadłeś słowo: {slowo}")
    else:
        print(f"Przegrałeś! Słowo to: {slowo}")

kolo_fortuny()
```

4. Podsumowanie pracy w grupach (20 minut)

- **Prezentacja wyników:**
  - Każda grupa przedstawia swoją grę, omawia jej działanie i prezentuje kluczowe elementy kodu.
  - Lider projektu podsumowuje proces pracy grupy: co poszło dobrze, co było wyzwaniem.

- **Feedback krzyżowy:**
  - Każda grupa ocenia inną grupę, skupiając się na współpracy i innowacyjności. Mogą użyć kryteriów:
    - Czy gra działa poprawnie?
    - Jakie pomysły wyróżniają grę?
    - Jak współpracowała grupa?

- **Dyskusja o współpracy:**
  - Nauczyciel zadaje pytania:
    - Co pomogło Wam w pracy zespołowej?
    - Jakie role były najważniejsze dla sukcesu projektu?
    - Co byście zmienili w swoim sposobie pracy?

- **Działania integracyjne:**
  - **Wymiana ról:** Każda grupa powtórnie analizuje swój projekt, ale zamienia role – dokumentalista wchodzi w rolę lidera, a programista staje się testerem. To umożliwia lepsze zrozumienie pracy innych.
  - **Burza mózgów:** Grupy wspólnie wymyślają, jak rozwinąć swoje gry (np. nowe funkcje, poprawa interfejsu). Najlepsze pomysły mogą być zaimplementowane na kolejnych lekcjach.

- **Nagrody za współpracę:**
  - Każdy zespół otrzymuje wyróżnienia w różnych kategoriach, np. "Najbardziej innowacyjny pomysł", "Najlepsza współpraca", "Najbardziej rozbudowany kod".

- **Indywidualna refleksja:**
  - Uczniowie wypełniają krótki formularz:
    - Co było najciekawszym elementem pracy w grupie?
    - Czego się nauczyłem/am o współpracy?
    - Co zrobiłbym/zrobiłabym inaczej przy kolejnym projekcie?