Mini-Jeu : Aventure en forêt

Scénario du jeu : Le joueur contrôle un héros qui explore une forêt mystérieuse. Dans la forêt, il peut rencontrer des monstres, combattre ou fuir, et collecter des objets qui peuvent l'aider à survivre.

Plan :

1. **Classes principales :**
    - `Personnage` (héros et monstres)
    - `Objet` (potion, épée, bouclier, etc.)
    - `Jeu` (mécanique principale du jeu)
2. **Interactions entre classes :**
    - Le héros peut :
        - Combattre des monstres
        - Utiliser des objets pour améliorer ses capacités
    - Les monstres peuvent attaquer le héros
    - Les objets peuvent être ramassés et utilisés en combat

1. Création de la classe `Personnage`

La classe `Personnage` va servir de modèle pour notre héros et nos monstres. Elle comprendra des attributs tels que le nom, les points de vie, la force, et des méthodes comme `attaquer()`.

```
class Personnage:
    def __init__(self, nom, points_de_vie, force):
        self.nom = nom
        self.points_de_vie = points_de_vie
        self.force = force

    def attaquer(self, autre_personnage):
        autre_personnage.points_de_vie -= self.force
        print(f"{self.nom} attaque {autre_personnage.nom} et inflige
{self.force} dégâts.")

    def est_vivant(self):
        return self.points_de_vie > 0

    def afficher_etat(self):
        if self.est_vivant():
            print(f"{self.nom} a {self.points_de_vie} points de vie restants.")
        else:
            print(f"{self.nom} est mort.")
```

2. Création de la classe `Objet`

Les objets que le héros peut trouver dans la forêt, comme des potions et des armes, seront représentés par la classe `Objet`.

```
class Objet:
```

```
    def __init__(self, nom, effet):
        self.nom = nom
        self.effet = effet  # Effet est une fonction qui modifie les stats du
personnage

    def utiliser(self, personnage):
        print(f"{personnage.nom} utilise {self.nom}.")
        self.effet(personnage)
```

Exemples d'objets :

- **Potion de soin :** Restaure des points de vie
- **Épée :** Augmente la force du personnage

```
# Création d'une potion qui restaure 20 points de vie
potion_soin = Objet("Potion de soin", lambda personnage: setattr(personnage,
'points_de_vie', personnage.points_de_vie + 20))

# Création d'une épée qui augmente la force de 10
epee = Objet("Épée", lambda personnage: setattr(personnage, 'force',
personnage.force + 10))
```

3. Création de la classe `Jeu`

La classe `Jeu` va gérer la boucle principale du jeu, les rencontres avec des monstres, et l'utilisation d'objets.

```
import random

class Jeu:
    def __init__(self, heros):
        self.heros = heros
        self.monstres = [
            Personnage("Gobelin", 30, 5),
            Personnage("Troll", 50, 10),
            Personnage("Dragon", 100, 20)
        ]
        self.objets = [potion_soin, epee]

    def rencontre_monstre(self):
        monstre = random.choice(self.monstres)
        print(f"Un {monstre.nom} apparaît !")
        return monstre

    def trouver_objet(self):
        objet = random.choice(self.objets)
        print(f"{self.heros.nom} trouve un objet : {objet.nom}")
        return objet

    def jouer(self):
        while self.heros.est_vivant():
            action = input("\nQue voulez-vous faire ? (combattre/fuir/chercher
un objet) : ").lower()

            if action == "combattre":
                monstre = self.rencontre_monstre()
                while monstre.est_vivant() and self.heros.est_vivant():
                    self.heros.attaquer(monstre)
                    if monstre.est_vivant():
```

```
                    monstre.attaquer(self.heros)
                self.heros.afficher_etat()
                monstre.afficher_etat()

            elif action == "chercher un objet":
                objet = self.trouver_objet()
                utiliser = input(f"Voulez-vous utiliser l'objet {objet.nom} ?
(oui/non) : ").lower()
                if utiliser == "oui":
                    objet.utiliser(self.heros)
                self.heros.afficher_etat()

            elif action == "fuir":
                print(f"{self.heros.nom} décide de fuir le combat.")

            if not self.heros.est_vivant():
                print("Le jeu est terminé. Votre héros est mort.")
                break
```

4. Mise en place du héros et lancement du jeu

Nous allons maintenant créer un héros et démarrer la boucle principale du jeu.

```
# Création du héros
heros = Personnage("Héros", 100, 15)

# Création et démarrage du jeu
jeu = Jeu(heros)
jeu.jouer()
```

Exemple de déroulement du jeu :

```
Que voulez-vous faire ? (combattre/fuir/chercher un objet) : combattre
Un Gobelin apparaît !
Héros attaque Gobelin et inflige 15 dégâts.
Gobelin attaque Héros et inflige 5 dégâts.
Héros a 95 points de vie restants.
Gobelin a 15 points de vie restants.
Héros attaque Gobelin et inflige 15 dégâts.
Gobelin est mort.
Héros a 95 points de vie restants.

Que voulez-vous faire ? (combattre/fuir/chercher un objet) : chercher un objet
Héros trouve un objet : Potion de soin
Voulez-vous utiliser l'objet Potion de soin ? (oui/non) : oui
Héros utilise Potion de soin.
Héros a 115 points de vie restants.
```

Améliorations possibles :

- **Ajout d'une classe** `Monstre` qui hérite de `Personnage`, permettant de créer des monstres spécifiques avec des comportements uniques.
- **Ajout de différents types d'objets** : des potions de mana, des armures pour augmenter la défense, etc.

- **Création d'un système de niveau** pour le héros et les monstres, avec un gain d'expérience après chaque combat.
- **Amélioration de l'intelligence artificielle des monstres**, par exemple en leur permettant de fuir ou d'utiliser des objets.
- **Ajout d'une carte** permettant au héros de se déplacer et de rencontrer des monstres aléatoires à chaque déplacement.

Avec ce mini-jeu, tu peux approfondir la gestion des interactions en POO, tout en t'amusant à explorer la forêt, combattre des monstres et ramasser des objets ! N'hésite pas à le modifier et l'améliorer selon tes envies !