COURS 1

Les dictionnaires Python

Durée : 20 minutes

1.1 1. Qu'est-ce qu'un dictionnaire ?

Un **dictionnaire** est une structure de données qui permet de stocker des informations sous forme de **paires clé-valeur**.

Analogie : Un dictionnaire papier - On cherche un **mot** (la clé) pour obtenir sa **définition** (la valeur) - On ne dit pas "donne-moi le 1547ème mot" mais "donne-moi la définition de 'escape' "

1.2 2. Syntaxe d'un dictionnaire

```
# Création d'un dictionnaire
personne = {
    "nom": "Alice",
    "age": 25,
    "ville": "Paris"
}
```

Structure :

```
nom_dictionnaire = {
    "clé1": valeur1,
    "clé2": valeur2,
    "clé3": valeur3
}
```

- Les **clés** sont entre guillemets (chaînes de caractères)
- Les **valeurs** peuvent être : nombres, chaînes, booléens, listes, images, rectangles Pygame...
- Les paires clé-valeur sont séparées par des virgules

1.3 3. Opérations de base

1.3.1 Accéder à une valeur

```
joueur = {"nom": "Alice", "vie": 100, "niveau": 1}

print(joueur["nom"])      # Affiche : Alice
print(joueur["vie"])      # Affiche : 100
```

1.3.2 Modifier une valeur

```
joueur["vie"] = 80        # La vie passe à 80
joueur["niveau"] = 2      # Le niveau passe à 2
```

1.3.3 Ajouter une nouvelle paire clé-valeur

```
joueur["score"] = 1500    # Ajoute la clé "score"
```

1.3.4 Vérifier si une clé existe

```
if "vie" in joueur:
    print("Le joueur a des points de vie")
```

1.4 4. Les dictionnaires pour l'escape game

1.4.1 Structure d'un objet

Chaque objet du jeu est représenté par une **variable** contenant un **dictionnaire** :

```
# Charger l'image et créer le rectangle
door_image = pygame.image.load("porte.png")
rect_porte = pygame.Rect(650, 150, 100, 200)

# Créer le dictionnaire de la porte
porte = {
    "nom": "porte",
    "image": door_image,
    "x": 650,
    "y": 150,
    "largeur": 100,
    "hauteur": 200,
    "rect": rect_porte,
    "visible": True,
    "verrouille": True,
    "description": "Une porte en bois"
}
```

1.4.2 Propriétés communes à tous les objets :

- "nom" : nom de l'objet

- `"image"` : image Pygame à afficher
- `"x"`, `"y"` : position de l'objet
- `"largeur"`, `"hauteur"` : dimensions
- `"rect"` : rectangle Pygame pour détecter les clics
- `"visible"` : True/False - affiche l'objet ou non
- `"description"` : texte à afficher

1.4.3 Propriétés spécifiques :

- `"verrouille"` : pour une porte
- `"ramasse"` : pour un objet ramassable
- `"ouvert"` : pour un coffre
- `"examine"` : pour un objet examinable

1.5 5. Utiliser les objets dans le jeu

1.5.1 Afficher un objet

```
# Dans la boucle de jeu
if porte["visible"]:
    screen.blit(porte["image"], (porte["x"], porte["y"]))
```

1.5.2 Détecter un clic

```
if event.type == pygame.MOUSEBUTTONDOWN:
    pos_souris = event.pos

    # Vérifier si on a cliqué sur la porte
    if porte["rect"].collidepoint(pos_souris):
        print(porte["description"])
```

1.5.3 Modifier l'état d'un objet

```
# Ramasser la clé
cle["ramasse"] = True
cle["visible"] = False

# Ouvrir le coffre
coffre["ouvert"] = True

# Déverrouiller la porte
```

```
porte["verrouille"] = False
```

1.5.4 Vérifier les interactions

```
# Essayer d'ouvrir la porte
if porte["rect"].collidepoint(pos_souris):
    if cle["ramasse"]:  # Si le joueur a la clé
        if porte["verrouille"]:
            porte["verrouille"] = False
            message = "La porte s'ouvre !"
    else:
        message = "La porte est verrouillée"
```

1.6 6. Récapitulatif

 Dictionnaire : Structure de données avec des paires clé-valeur

 Syntaxe : `{"clé": valeur, "clé2": valeur2}`

 Accès : `dictionnaire["clé"]`

 Pour l'escape game : Chaque objet est un dictionnaire

 Avantages : Code lisible, flexible, facile à maintenir

1.7 7. Structure type d'un objet

```
nom_objet = {
    "nom": "...",              # Nom de l'objet
    "image": ...,              # Image Pygame
    "x": ...,                  # Position X
    "y": ...,                  # Position Y
    "largeur": ...,            # Largeur
    "hauteur": ...,            # Hauteur
    "rect": ...,               # Rectangle Pygame
    "visible": True/False,     # Affichage
    "description": "...",      # Description
    # + propriétés spécifiques
}
```