DS - Initiation aux Concepts Objet

Gaël Roustan (Argonaultes)

2024

Abstract

Ce document présente le sujet d'évaluation du cours d'introduction aux concepts de la programmation orientée objet.

DS - Initiation aux Concepts Objet

Système de notation

Un exercice unique permettra d'évaluer différents aspects vus en cours à travers l'implémentation d'une structure bien connue : la pile.

Critère	Barême
Déclaration de classe	5
Formatage	2
Méthode add_to_stack	3
Méthode pop_from_stack	3
Méthode is_empty	2
Méthode is_full	2
Héritage des Exceptions	3

Livrables

Pour effectuer le rendu de votre projet, vous devez fournir l'adresse de votre repo GIT au format `git@.+:.+\.git` via le formulaire Google fourni.

Contraintes

Vous n'avez pas le droit d'utiliser des collections d'objet disponible en Python. Le non respect de cette consigne entraine la note de 0 au devoir.

Consignes

Vous devez créer au moins une classe afin de fournir un moyen de gérer un ensemble d'objets sous forme d'une pile.

Cela signifie que vous avez une structure principale `MyStack` qui rassemble les objets entre eux, quelque soit la nature ou le type de cet objet.

Cette structure principale doit vous permettre de rajouter un élément en haut de votre pile via une méthode qui doit s'appeler `add_to_stack`, et une autre méthode doit vous permettre de récupérer le dernier élément ajouté `pop_from_stack`. Lorsqu'un élément est récupéré, il ne doit plus être disponible dans la pile.

Si jamais la pile est vide, vous devez renvoyer une exception `MyEmptyStackException`. Cette Exception doit hériter de la classe `Exception`.

A la création de votre pile, vous devez spécifier un nombre maximale d'éléments. Au moment de l'ajout, si vous dépassez la taille maximale, vous devez envoyer l'exception `MyOutOfSizeException`.

Afin d'éviter de rencontrer ces 2 exceptions, vous fournirez 2 méthodes `is_full` et `is_empty` pour savoir respectivement si la pile est pleine (donc au prochain appel, ce sera une erreur) ou la pile est video (donc au prochain pop, ce sera également une erreur).

Exemple d'utilisation

```
if __name__ == '__main__':
    myStack = MyStack(3)
    myStack.add_to_stack('hello')
    myStack.add_to_stack('hello')
    print(myStack.is_full()) # False
    myStack.add_to_stack('hello')
    print(myStack.is_full()) # True
    myStack.add_to_stack('hello') # MyOutOfSizeException
    print(myStack.pop_from_stack()) # hello
    print(myStack.is_empty()) # False
    print(myStack.pop_from_stack()) # hello
    print(myStack.is_empty()) # False
    print(myStack.pop_from_stack()) # hello
    print(myStack.is_empty()) # True
    print(myStack.pop_from_stack()) #
     ↪  MyEmptyStackException
```