**Outils Numériques pour l'Ingénieur·e en Physique**

*2025-2026*

*6N-076-PHY / ONIP-2*

**Bloc 4 - Prog. Objet** ( 100%)

**Concepts étudiés**

[Num] Classes et objets

**Mots clefs**

Python; Programmation objet; Classes; Objets; Méthodes; Attributs

**Sessions**

**0** Cours(s) - 1h30
**0** TD(s) - 1h30
**6** TD(s) Machine - 2h00
**0** TP(s) - 4h30

Travail

**Par binôme**

**Institut d'Optique**
Graduate School, *France*
https://www.institutoptique.fr

**GitHub - Digital Methods**

https://github.com/IOGS-Digital-Methods

# Introduction à la programmation orientée objet

Dans le cadre du module **ONIP-2**, vous serez amenés à réaliser un mini-projet orienté photonique parmi deux sujets au choix :

- Carte d'éclairement de sources lumineuses
- Calibration des couleurs d'images numériques
- Tracé de rayons dans des systèmes optiques à dioptres (2D)
- (Plus Difficile) Tracé de rayons dans des systèmes optiques à dioptres en 3D

D'un point de vue programmation, vous devrez développer ce projet selon les règles de la **programmation orientée objet**.

**Aucune fonction ne devra être utilisée en dehors d'un objet.**

## Déroulement du module

Ce module se déroule sur **6 séances** :

**Séance 1** Découverte de la programmation objet
**Séances 2 à 5** Réalisation du mini-projet en binôme
**Séance 6** Evaluation du mini-projet en binôme

## Livrables attendus

**Vous aurez 10 minutes lors de la séance 6 pour présenter l'ensemble de vos résultats et vos analyses.**

Pour valider cette session, vous devez **présenter** les **livrables suivants** :

1. **Classes commentées** (selon la norme PEP 8) pour générer des objets
2. **Graphiques légendés** incluant toutes les données nécessaires à la bonne compréhension des données présentées
3. **Analyse des figures** obtenues

*Les critères d'évaluation et les étapes à suivre sont donnés dans les diapos d'introduction et à la fin de ce document.*

## Ressources

Cette séquence est basée sur le langage Python.

Vous utiliserez l'environnement **PyCharm** et *Anaconda 3*.

Des tutoriels Python (et sur les bibliothèques classiques : Numpy, Matplotlib or Scipy) sont disponibles à l'adresse : http://lense.institutoptique.fr/python/.

# Séance 1 - Programmation orientée objet

Afin de vous familiarisez avec les principes de base, la première séance sera consacrée à **l'étude et la mise en oeuvre d'exemples de la programmation orientée objet** en Python : écriture d'une classe, instanciation d'un objet, interaction entre les objets.

## Acquis d'Apprentissage Visés

1. **Créer des classes** incluant des méthodes et des attributs
2. **Instancier des objets** et les faire interagir
3. **Définir et documenter les méthodes et attributs** de chaque classe

L'ensemble des documents du module ONIP-2 se trouve sur le site du LEnsE : http://lense.institutoptique.fr/ONIP/ . Les exemples pour cette première séance se trouvent dans la rubrique **BLOC 4**. Ajoutez *from ___future___ import annotations* en début de code pour une meilleure gestion des annotations. **Attention**, les classes crées pour ces exercices ne seront pas réutilisées pour les projets.

## Exercice 1 - Classe Point

En vous inspirant de la classe **Animal** (simple) :

- créez un nouveau fichier .py
- définissez une classe **Point**, permettant de modéliser un point dans un espace en 2 dimensions par ses coordonnées x et y
- instanciez deux objets de type Point avec des coordonnées différentes
- redéfinissez la méthode ***___str___*** pour qu'elle affiche les coordonnées d'un objet de type **Point** (voir exemple de la classe **Animal**)
- définissez une méthode barycentre qui prend un objet **Point** en argument et retourne un nouvel objet point correspondant au barycentre
- vérifiez vos différentes méthodes

## Exercice 2 - Classe Rectangle

Dans le fichier précédent et en utilisant la classe **Point** :

- définissez une classe **Rectangle**, permettant de modéliser un rectangle orienté selon les axes x et y à partir de deux objets de type Point (sommets opposés du rectangle)
- définissez des méthodes ***perimetre*** et ***surface*** permettant de calculer le périmètre et la surface d'un objet de type **Rectangle**
- définissez une méthode recouvrement qui prend un objet **Rectangle** en argument et renvoie un booléen indiquant si il y a un recouvrement entre les deux rectangles.
- définir une méthode agrandir qui applique un facteur d'échelle au rectangle
- redéfinissez la méthode ***___str___***
- testez l'ensemble de vos méthodes sur différents objets de type **Rectangle**

## Exercice 3 - Classe Cercle

- même exercice avec un cercle défini par son centre et un point du rayon. Cette fois, la méthode intersetion retournera une liste de zéro à deux objets points correspondants aux points d'interection.
- redéfinissez la méthode ***___str___***

## Outils Numériques

**Fonctions et bibliothèques conseillées** :

- **Numpy** gestion de matrices
- **Matplotlib** affichage de données
- **Scipy** fonctions scientifiques

## Fichiers d'exemple

Classe **Animal** (simple) :

onip_b4_a_classe_simple.py

Classe **Animal** (redéfinition str) :

onip_b4_b_classe_simple _redefinition.py

Classes **Dog** et **Cat** :

onip_b4_c_classe_heritage

# Evaluation du module

Lors de la sixième séance, vous devrez présenter le travail que vous avez réalisé sur l'un des deux projets proposés.

Vous devrez également **au cours des séances** faire **valider l'application minimale** visée. Le descriptif des attendus de l'application minimale est donné dans les sujets des projets.

Vous devrez enfin présenter des résultats sur l'**une des ouvertures proposées** sur chacun des projets.

## Présentation du travail

Vous serez **convoqués par binôme** 15 min avant le début de votre présentation.

Vous aurez alors **7 min** pour présenter les aspects suivants de votre travail :

**1 min** Présentation générale - Problématique
**2 min** Résultats sur le système final
**4 min** Présentation de code

Vous aurez ensuite 5 min de questions par le jury. Vous devrez être en mesure d'exécuter votre code pour montrer son bon fonctionnement. Votre code devra être déposé sur e-campus avant le jour de l'évaluation.

## Critères d'évaluation

Vous serez évalué.e selon les critères suivants :

- **Méthodologie**
  - Bon usage de la programmation orientée objet
    - objets mis en oeuvre
    - attributs et méthodes utiles pour chaque objet
  - Répartition de l'écriture du code
- **Programmation**
  - Respect de la charte PEP8 (noms des variables, méthodes, commentaires...)
  - Utilisation, écriture et validation de classes
- **Physique**
  - Graphiques pertinents et légendés
  - Données pertinentes de test
- **Avancement**
  - Application de base validée
  - Ouverture