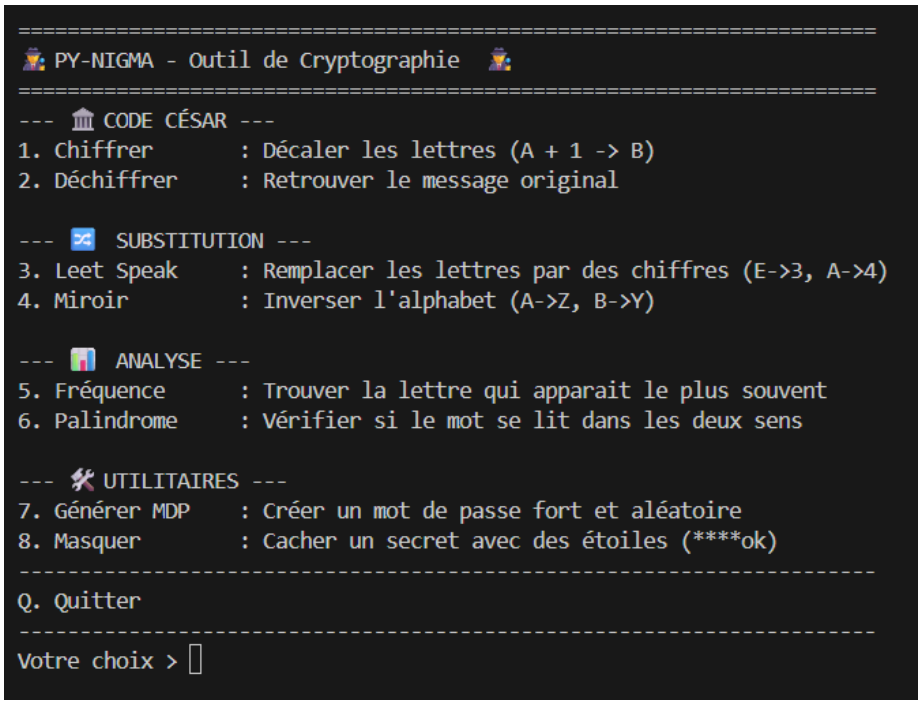
Exercice Commun : Py-Nigma

Il faut une **application** (pour espions) qui aide à chiffrer et déchiffrer des messages.

Un **repo github** existe. Le squelette du projet s'y trouve.
Les **tests unitaires** sont écrits pour valider le projet. (réduits pour être abordables)
Un **Github Project** est prêt pour aider à gérer les tâches.
Un **pipe Github Actions** est prêt pour valider les Pull Requests.
Arriverez-vous à **coder** ensemble ce projet, le **soumettre** et le **lancer** ?



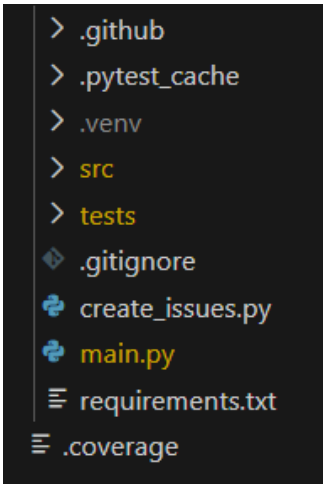
Guide de Contribution au Projet Py-Nigma

Votre mission : Contribuer au développement de la librairie de cryptographie.

🛠️ Étape 0 : L'installation

1. **Fork :**
    - Allez sur le lien GitHub du projet (upstream) fourni par le prof :
      https://github.com/[PROF]/[PROJET]
    - Cliquez sur le bouton **Fork** (en haut à droite)
      Cela crée une copie sur *votre* compte github (origin).
      Un **Fork** est une copie personnelle d'un projet, permettant d'y collaborer.
2. **Clone (Rapatrier sur PC)** :
    - **git clone https://github.com/[VOTRE_PSEUDO]/[PROJET].git**
    - **cd [PROJET]**
3. **La Liaison Upstream** :
    - Pour récupérer le code des autres validés sur le repo du prof,
      il faut spécifier l'upstream pour ensuite l'utiliser :

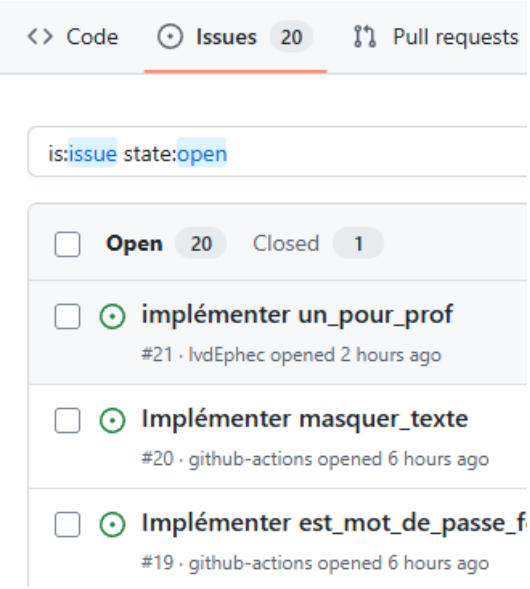
      git remote add upstream https://github.com/[PROF]/[PROJET].git
4. **.venv (environnement virtuel)**
    - (Windows) **python -m venv .venv**
      ou
      (Mac/Linux) **python3 -m venv .venv**
5. **Installer les requirements du projet.** (Ici ça installe juste pytest)
    - (Windows) **venv\Scripts\activate**
      ou
      (Mac/Linux) **source .venv/bin/activate**
    - **pip install -r requirements.txt**



Comprendre, après votre fork, clone, upstream :

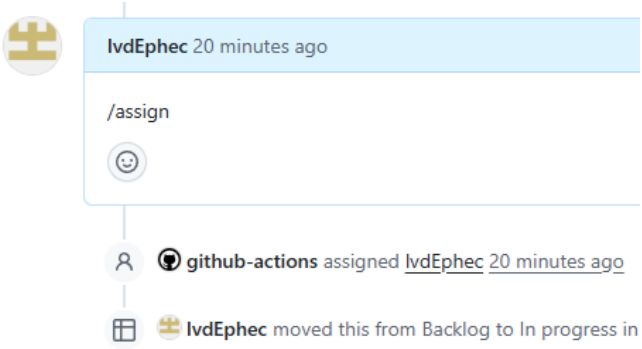
- **origin** : c'est votre repo en ligne
- **upstream** : c'est le repo du prof

🎯 Étape 1 : Choisir une mission



Il y a **20 fonctions** à coder, réparties en **20 tâches**, visibles dans **issues** sur **https://github.com/[PROF]/[PROJET]**

Pour ne pas s'embrouiller, chacun doit indiquer sur quelle tâche il travaille **avant** de commencer (!)

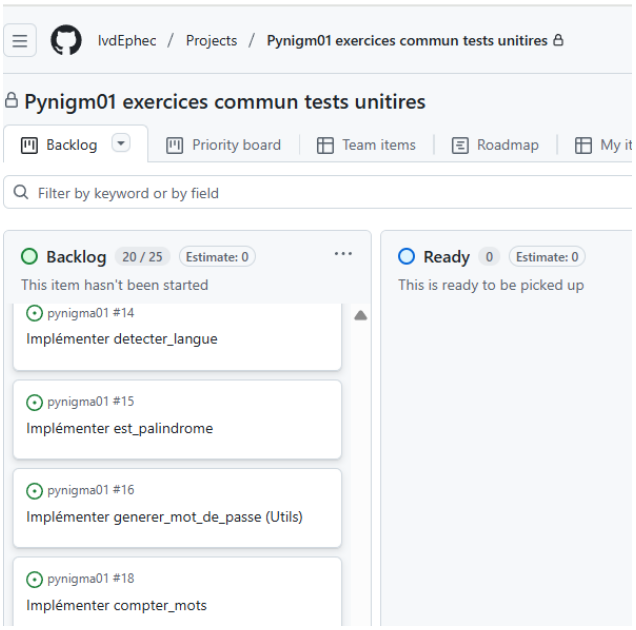


Allez dans l'onglet **Issues** du repo du prof.

1. Choisissez une tâche,
et commentez **/assign** dessus.
2. Notez le numéro (ex: **#12**).

Le prof montrera l'état du Github Project au fur et à mesure de l'avancement.
Les **bonnes colonnes** :

- **Backlog** : les tâches à réaliser du projet.
- **In progress** : pendant qu'on, travaille dessus
- **In Review** : lorsque c'est terminé, en attente de validation
- **Done** : lorsque la Pull Request est validée et le code mergé dans main

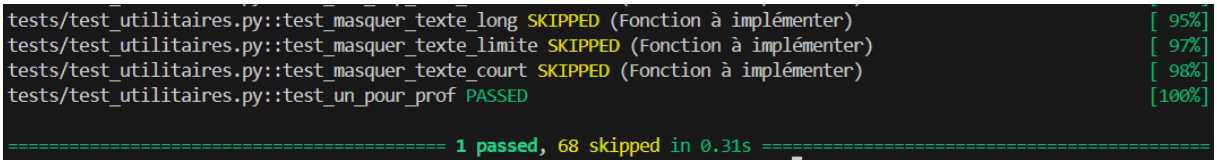


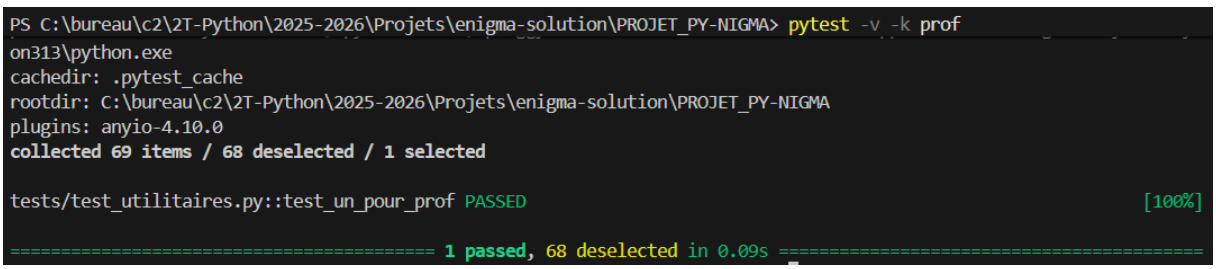
💻 Étape 2 : Coder

1. **Mettez-vous à jour localement** :
    - **git switch main**
      (On retourne sur la base)
    - **git pull upstream main**
      (On télécharge les nouveautés depuis le repo du prof)
    - **git push origin main**
      (On met à jour notre fork en ligne, optionnel mais conseillé)
2. **Créez votre branche de travail** :
    - **git switch -c prenom/ma-fonction**
3. **Codez** votre fonction dans src/.
4. **Testez** :
    - Lancez dans le terminal :
      pytest -v -k mot-clef

      -v : verbose, plus d'infos

      -k : seuls les tests contenant ***mot-clef*** seront lancés
      (Pour ne pas lancer les 20 tests. Les autres vont « skiper ». A tester.)
    - Si c'est VERT ✅ (et pas « skip »), c'est bon !





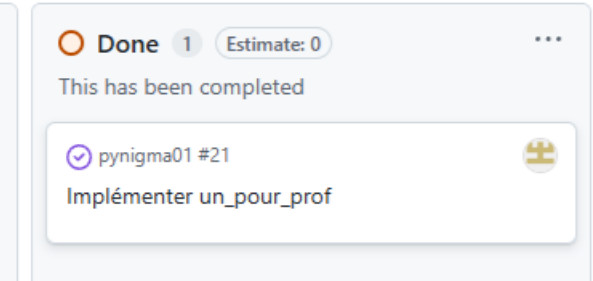
📮 Étape 3 : Envoyer

1. **Sauvegardez** :
    - **git add .**
    - **git commit -m "Finit la fonction X (closes #12)"**
      *Mettre "**closes #numéro**" fermera le ticket **automatiquement** !*
      *Mettre **#numéro** dans le **comit** génère les liens entre tâche / comit !*
2. **Envoyez** sur votre compte :
    - **git push origin prenom/ma-fonction**
3. **Proposez (Pull Request)** vers le repo du prof (upstream) :
    - Allez sur GitHub. Cliquez sur "**Compare & pull request**".
    - Validez et Créez le **pull request**.
    - Le pipe va s'activer et lancer les tests unitaires. (se relance lors de push)
    - Pas d'erreurs lors des tests unitaires ?
      -> Demandez au prof d'accepter la PR !
      Et se lancer dans le codage d'une prochaine tâche s'il en reste.

      Sinon, si erreur (avez-vous bien lancer les test en local ??) :
      corriger, puis :
      git add .
      git commit -m "Correction du bug sur la fonction X"
      git push origin prenom/ma-fonction
      et le PR se met automatiquement à jour

Étape 4 : Récupérer le projet et l'utiliser

Lorsque toutes les tâches du Github Project sont à Done,



Vous pouvez le récupérer et l'essayer.

1. Revenir sur la branche principale
   git switch main
2. Télécharger la version finale du prof (contenant toutes les fonctions)
   git pull upstream main
3. Lancer (les tests unitaires puis) le **main**