Exercice 1.
Un codon est constitué de 3 nucléotides, et a pour base azotée soit A, soit T, soit G, soit C.
1) Ecrire une fonction de vérification qui prend en entrée une chaine de 3 nucléotides, et qui renvoie True si c'est bien un codon et False sinon.
On rappelle quels sont les 3 codons 'stop' : TAG, TAA, TGA.
2) Ecrire une fonction qui cherche le codon stop dans une séquence d'ADN (si il y en a plusieurs alors on s'arrête au premier) et renvoie la position de ce codon dans la séquence. Si il n'y a pas de codon stop alors la fonction renvoie -1.
On prend en compte le fait que la lecture de la séquence se fait de codons en codons (c'est à dire tous les 3 nucléotides).
3) Les nucléotides complémentaires sont : (A-T) et (C-G).
a) Ecrire une fonction qui prend en entrée un nucléotide et qui renvoie son complémentaire
b) Ecrire une fonction qui prend deux brins de même taille et qui vérifie si ils sont complémentaires. Si oui, elle renvoie True, sinon elle renvoie False.
4) On rappelle le phénomène de croisement entre deux brins au niveau d'une position k sur le brin. Par exemple si les brins : 'ATTGTC' et 'CGCTGA' se croisent à la position 3, les deux nouveaux brins sont : 'ATTTGA' et 'CGCGTC'.
Ecrire une fonction qui prend en paramètre deux brins et une position k et qui fait le croisement. La fonction renvoie les deux nouveaux brins.

Exercice 2.

1. Créer une fonction *anagra* qui prend un mot M en entrée et qui teste si M est un anagramme (par exemple $anagra('RIGHHGIR')$ renvoie 1 ou $True$).
2. Créer une fonction *verlan* qui prend un mot en entrée et qui inverse les deux moitiés du mot (par exemple $verlan('DFRTGGZE') =' GGZEDFRT'$).
3. Créer une fonction qui prend en entrée une phrase et qui retourne le nombre de mots.
   etc...

Exercice 3.

1. On modélise ici une marche aléatoire dans $\mathbb{Z}$. A l'instant 0, l'objet est en 0, puis se déplace de $\pm$ 1 à chaque instant, avec une probabilité $1/4$ de se déplacer vers la droite. Créer une fonction qui modélise cette marche et renvoie la position après n déplacements.
2. Idem, mais la fonction renvoie l'instant du premier retour en 0 s'il existe.
3. On modélise ici une marche aléatoire dans une grille G de taille (r, r), dont le départ se fait en $(0, 0)$. Créer une fonction qui modélise les n premiers déplacements, qui se font aléatoirement dans l'une des quatre directions standard, tout en restant dans la grille.
4. Reprendre le programme ci-dessus et renvoyer, pour chaque colonne, le nombre de passages par cette colonne.
   etc...

Exercice 4.
Le Poker est un jeu de cartes utilisant 52 cartes définies par une couleur (coeur, carreau, pique, trèfle) et une valeur ($2-10$, valet, dame, roi, as). Un joueur dispose d'une main de 5 cartes.

1. Une carte sera représentée par un couple [couleur,valeur]. Par exemple $[1,6]$ est le 6 de coeur, $[2,13]$ est le roi de carreau, $[3,1]$ est l'as de Pique. Quelle structure de données représente le jeu ? une main ? Créer le jeu sous Python.
2. Ecrire une fonction qui génère une main de 5 cartes aléatoirement.
3. Ecrire une fonction permettant de savoir si une main donnée contient au moins un carreau.
4. Ecrire une fonction qui permet de détecter un full (3 cartes de même valeur et 2 cartes d'une même valeur) dans une main.

Exercice 5.
On a une liste contenant des nombres (qui peuvent se répéter).

1. Ecrire une fonction $comptage(liste, x)$ qui compte les occurrences d'un nombre.
   Par exemple, $comptage([8,55,7,3,7,55,9,9,9],55) = 2$;
   $comptage([8,55,7,3,7,55,9,9,9],4) = 0$.
2. Ecrire une fonction $supprime(liste, x)$ qui supprime toutes les occurrences du nombre x à l'exception de la première.
   Par exemple, $supprime([8,55,7,3,7,55,9,9,9],55) = [8,55,7,3,7,9,9,9]$;
   $supprime([8,55,7,3,7,55,9,9,9],4) = [8,55,7,3,7,55,9,9,9]$.
3. Ecrire une fonction qui, à partir d'une liste donnée, redonne la même liste mais sans répétition.
4. Evaluer la complexité algorithmique de ce dernier programme en fonction de n, la longueur de la liste.

Exercice 6.
Un musée est composé d'une collection permanente et d'une reserve. Il y a 7 salles et 3 natures d'oeuvre (peinture...).
Par exemple, le musée $M = [['P', 4, 1], ['R', 6, 3], ['P', 6, 1]]$ contient trois oeuvres ; la première oeuvre est dans la collection permanente, dans la salle 4, et de nature 1, etc...

1. Expliquer en une phrase quelle est la structure de données utilisée pour représenter la collection.
2. Créer une fonction qui compte le nombre d'oeuvres cachées au public (dans la réserve).
3. Ecrire une fonction qui compte le nombre d'oeuvres pour chaque salle.
   Par exemple, pour M précédemment écrite, $nombre(M) = [0,0,0,1,0,2,0]$.
4. Ecrire une fonction qui range le musée selon le type d'oeuvre, en renvoyant trois listes, la première contenant les oeuvres de type 1, ...
5. Ecrire un programme qui trie les salles selon le nombre d'oeuvres qu'elles contiennent.