Mini-projet L3

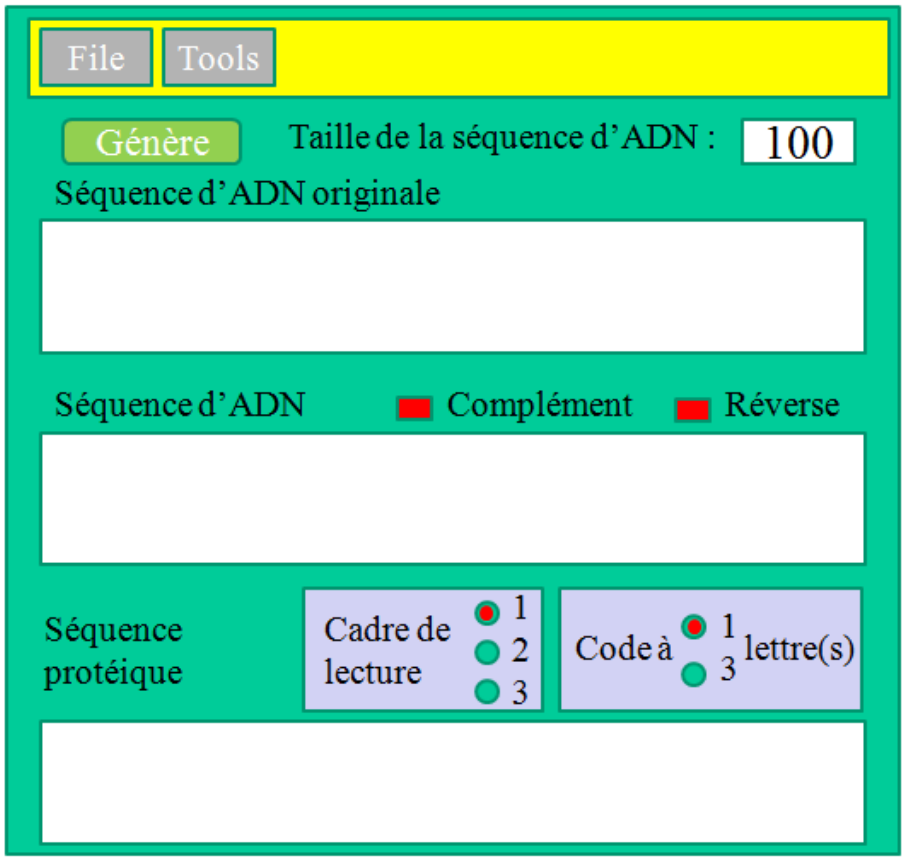
UE : Introduction à l'algorithmique

Nicolas Scalzitti, Jean-Marie Wurtz

Projet à rendre pour le 16 janvier 2022 sous Moodle

Le but de ce mini-projet est de réaliser une application graphique complète. Cette application transcrira un brin un d'ADN génomique en son produit (putatif) protéique en utilisant une table de codon.

L'application se présentera sous la forme suivante :



Une séquence d'ADN peut-être obtenu de 2 manières :

- génération aléatoire (bouton « génère » et champ entier « 100 »)
- lecture d'un fichier au format « fasta » par la commande du menu « File »

La séquence d'ADN sera affichée dans la première fenêtre. Le complément de la séquence et/ou la réversion de la séquence sera à utiliser dans la traduction. La séquence d'ADN "modifiée" ou non sera visualisée dans la deuxième fenêtre.

La traduction des codons ce fait normalement avec les 4 nucléotides A, G, C, U à partir de l'ARN messager. Au niveau de l'application l'ADN sera issu du séquençage d'un génome donc on utilisera les 4 lettres suivantes A, G, C, T. On adaptera les codons en conséquences.

Le produit de la traduction, la séquence protéique, sera visualisé dans la troisième fenêtre.

Les options sont faciles à comprendre je ne les expliciterai pas dans ce document. S'il devait y avoir des doutes nous les règlerons en cours ou en TP/TD.

La barre de menu propose 2 options :

- File : cette option affichera 2 commandes :
    - Lecture d'un fichier : ouverture d'un fichier au format Fasta
    - Quitter : quitter l'application
- Tools
    - Affiche la table des codons

Format obligatoire pour rendre le mini-projet :
Le projet devra être fourni sous forme d'un fichier zip (pas de rar, tar.gz ou autres formats) contenant tout sur le projet et dont le nom aura le format suivant :

<Nom>_<Prénom>._projet_L3python.zip exemple : WURTZ_Jean-Marie_L3python.zip

Une fois extrait, un fichier « ALIRE.txt » doit être visible sous la racine du projet pour expliquer ou trouver le rapport et comment exécuter l'application.

- L'arborescence du projet sauvegardé dans le fichier zip sera la suivante :
    - ALIRE.txt
    - rapport_<nom>_<prénom>_L3python.pdf : le rapport au format pdf
    - src : l'arborescence des modules python
    - le programme principal python à exécuter

Comme exemple de fichiers génomiques vous pourrez prendre ceux utilisés en TP/TD.

Notation : il sera tenu compte au niveau de l'application que vous allez rendre de :

- Rapport 40% : 3 pages (sans la page titre et sans les annexes)
  Faites attention à la lisibilité de votre document, évitez le copier/coller sans référence. Respectez la structure d'un document : page titre, page table des matières, introduction, développement de vos idées en 3 points, conclusion, annexe (figures, pas de code source sauf pour illustrer une situation)
- votre programme 60%

Mise en forme du rapport

1. Page de garde :
    a. Titre du rapport
    b. Nom de l'étudiant
    c. L'université, son Logo
    d. L'UE et l'enseignant responsable
2. Table des Matières
3. Le document
    a. Introduction, votre développement, conclusion
4. 3 pages, interligne 1.5, fonte : times new roman 12

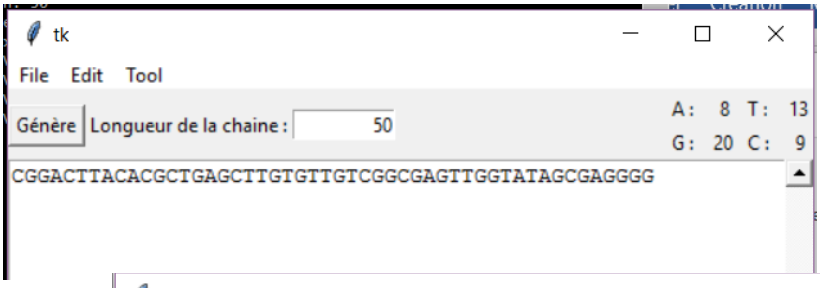
Annexes
Les nucléotides : fr.wikipedia.org/wiki/Nucléotide
Le code génétique : fr.wikipedia.org/wiki/Code_génétique
Les codons : fr.wikipedia.org/wiki/Codon
Format Fasta : fr.wikipedia.org/wiki/FASTA_(format_de_fichier)
Trouver des séquences génomiques

- www.ncbi.nlm.nih.gov/genome/guide/human
- hgdownload.cse.ucsc.edu/goldenPath/hg19/chromosomes/

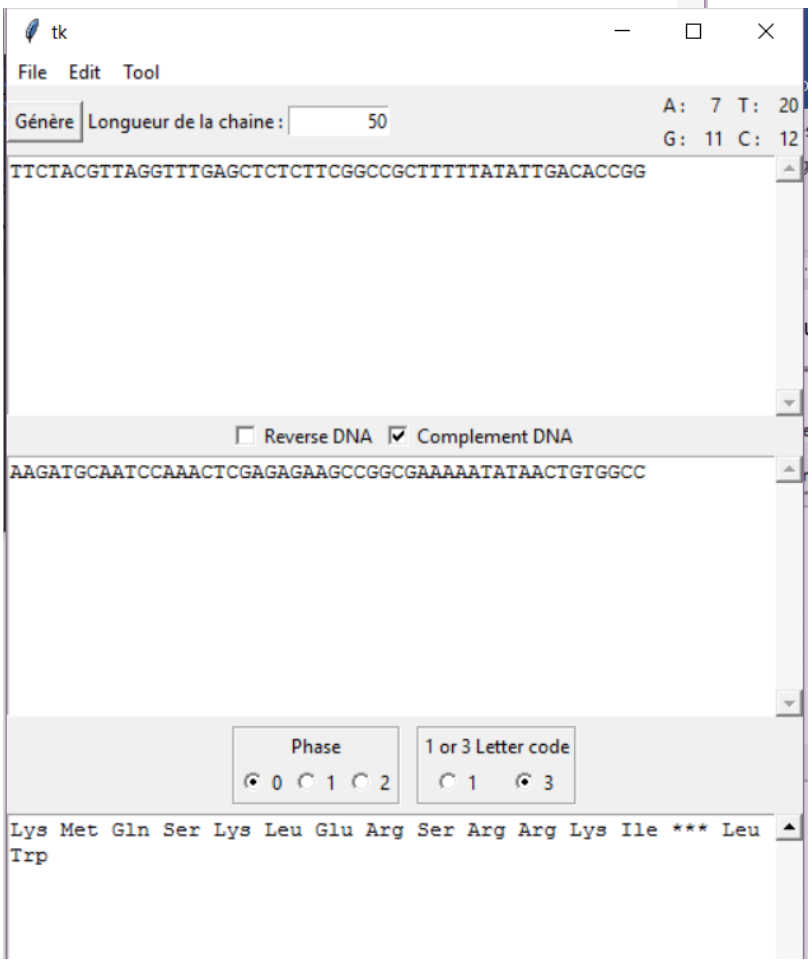
Aide :

1. Ne pas oublier l'option « command » pour les boutons (standard, radio et à cocher) qui vous permet de mettre en place une fonction ou méthode de type « callback »
2. Pour les méthodes « bind » et « bind_all », ne pas oublier l'option « add='+' » qui indique que le/les « callback » précédents sont à conserver

3. Vous aurez à votre disposition le code qui correspond à l'application suivante :



4. que vous devrez étendre de la manière suivante :




```
if __name__ == '__main__':
    print("in main")
    root=tk.Tk()
#le menu du haut
    ftop=tk.Frame()
    ftop.pack(side=tk.TOP, fill=tk.X)

    wgen=WDnaGenerator(ftop)
    wcnt=WNucleotideCounter(ftop)

    wgen.pack(side=tk.LEFT)
    wcnt.pack(side=tk.RIGHT)

    wpres=WDnaPresenter(root)
    wpres.pack(fill=tk.BOTH, expand=1)

    wmen=WDnaMenu(root)
    root.config(menu=wmen)

    wcr=WDnaConRev(root)
    wcr.pack(fill=tk.BOTH, expand=1)

    wtranslator=WDnaTranslator(root)
    wtranslator.pack(fill=tk.BOTH, expand=1)

    root.bind_all("<<NewDNA>>", wpres.onNewDnaSequenceEvent)
    root.bind_all("<<UpdateDNA>>", wcnt.onUpdateDnaSequenceEvent)
    root.bind_all("<<UpdateDNA>>", wcr.onUpdateDnaSequenceEvent, add="+")
    root.bind_all("<<UpdateProt>>", wtranslator.onUpdateProtEvent)

    root.mainloop()
~
```

5. Voici un exemple de code du programme principal avec les 2 nouvelles classe : « WDnaConRev » et « WWDnaTranslator »