REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
ECOLE NATIONALE SUPERIEURE D'INFORMATIQUE

1CS **Année 2019/2020**

Rapport TP THP

Réalisé par :
Assenine Mohamed Sami (G2)
Ait saadi Abdelmadjid (G4)

Section : A

Semestre 1

L'environnement de développement :

Dans ce TP, on a choisi de programmer avec le langage Python3 en utilisant l'IDLE de Python 3.7 et Jupyter de Anaconda.

Comme bibliothèque externe, on a utilisé « consolemenu » qui permet d'afficher un menu console et de le gérer.

Structure de l'automate :

On a travaillé avec une seule classe « Automate » qui contient 5 attributs (les 5 éléments nécessaires pour définir un automate), ces attributs sont :

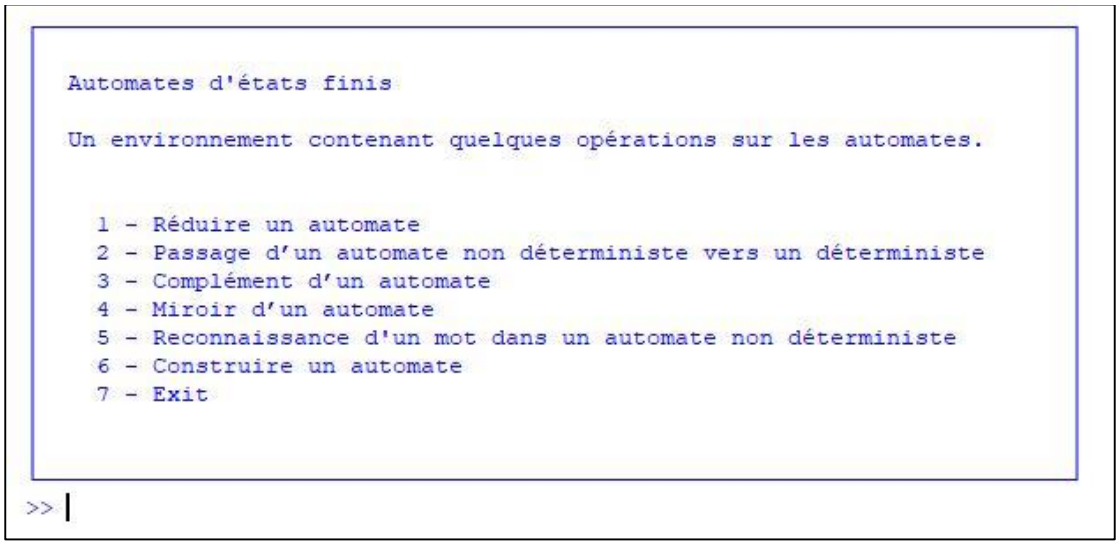
- « alphabet » : une liste de caractères.
- « etats » : une liste de chaine de caractères contenant les noms des différents états de notre automate.
- « etat_initial » : une chaine de caractère contenant le nom de l'état initial de notre automate.
- « etats_finaux » : une liste de chaine de caractères contenant l'ensemble des états finaux de notre automate.
- « instructions » : c'est une structure qui représente l'ensemble des transitions de notre automate, elle est de type Map (Dictionary en Python3). La clé est un couple (Tuplet en Python3) contenant l'état courant (e) et la lettre à lire (l) et la valeur est une liste d'états représentant l'ensemble des états qu'on peut les atteindre à la lecture de l à partir de l'état e.

Les fonctions de notre TP :

Notre TP est composé de 20 fonctions assurant le fonctionnement des 5 fonctionnalisées demandées.

- List etat_accessible(a) : c'est une fonction qui retourne une liste des états accessibles de l'automate « a ».
- List successeur_etat(s, a) : c'est une fonction qui retourne les liste des successeurs de l'état « s » dans l'automate « a ».
- Booléen is_coaccessible(s, a) : c'est une fonction qui retourne vrai si l'état « s » est coaccessible dans l'automate « a » sinon elle retourne faux.
- List etat_coaccessible(a) : c'est une fonction qui retourne une liste des états coaccessibles de l'automate « a ».
- Automate reduction_automate(a) : c'est une fonction qui retourne un automate réduit par rapport à l'automate d'origine « a ».
- Automate elimination_epsilon(a) : c'est une fonction qui retourne un automate équivalent à l'automate d'origine « a » mais sans transitions spontanés.
- List fermeture(etat, automate) : c'est une fonction qui retourne une liste d'états qui représente la fermeture de l'état « etat » dans notre automate « automate ».
- List fonction(liste, lettre, automate) : c'est une fonction qui permettra de remplir de le tableau qui permet de passer d'un automate non déterministe à un automate déterministe.

- List intersection_2_listes(lst1, lst2) : c'est une fonction qui retourne l'intersection des deux listes « lst1 » et « lst2 ».
- Automate non_deterministe_to_deterministe_complet(automate) : c'est une fonction qui retourne un automate déterministe complet équivalent à « automate » qui est à la base généralisé.
- Automate automate_generalise_vers_partiellement_generalise(automate) : c'est une fonction qui retourne un automate partiellement généralisé équivalent à l'automate généralisé « automate ».
- Automate complement_automate(automate) : c'est une fonction qui retourne l'automate complément de l'automate « automate ».
- Automate miroir_automate(automate) : c'est une fonction qui retourne l'automate miroir de l'automate « automate ».
- Booléen word_recognition(mot, etati, l, etatsf, a) : c'est une fonction qui retourne un vrai si le mot « mot » est reconnu par l'automate dont les paramètres état initial, états finaux et transitions sont donnés respectivement par « etati », « etatf » et « a », en plus cette fonction retourne l'ensemble des états parcouru pour reconnaitre ce mot et ça à travers la liste « l ». Dans le cas où le mot « mot » n'est pas reconnu pas l'automate, on retourne faux.
- Void reconnaissance_mot_automate_non_deterministe(mot, automate) : cette fonction permet d'afficher la trace du chemin parcouru par l'automate « automate » pour confirmer l'appartenance du mot « mot » dans ce dernier, et dans le cas contraire afficher un message d'erreur.
- Void afficher_automate(automate) : cette fonction permet d'afficher les différents attributs de l'automate « automate ».
- Booléen chaine_correcte(chaine, alphabet) : c'est une fonction qui retourne vrai si le mot « chaine » contient que des lettres de l'alphabet « alphabet », sinon elle retourne faux.
- Automate construire_automate() : cette fonction permet de créer (construire) un automate à partir des données entrées par l'utilisateur via la console.
- Automate lire_fichier(nomFichier) : cette fonction permet de lire un automate à partir du fichier « nomFichier » et le retourner.
- Void ecrire_fichier(nomFichier, automate) : cette fonction permet d'écrire l'automate « automate » et le sauvegarder dans le fichier « nomFichier ».



Exemple d'affichage console de notre TP