**SOLUTION DE POSITIONNEMENT STANDARD GPS BASÉE SUR PYTHON**

**Elaboré par :**

**L’HAMDOUCHI Driss N°35**

**Encadré par :**

**Professeur Moha EL-AYACHI**

**Introduction :**

Créer un programme python pour obtenir une solution de positionnement pour un récepteur GPS à l'aide de ses fichiers RINEX d'observation et de navigation. Cela permettra l'application des concepts appris dans les cours du fondement GNSS et l’infrastructure géodésique à travers un projet pratique qui allie théorie et pratique réelle. Cela apportera une plus grande clarté sur le fonctionnement du GPS et GNSS généralement.

**Objectives :**

* Développer des fonctions Python pour traiter les fichiers d'observation et de navigation RINEX et stocker les données dans une structure de données.
* Développer une fonction Python pour déterminer la position des satellites à l'aide des fichiers de navigation.
* Développer une fonction Python permettant le traçage de l'orbite des satellites pendant une certaine période de temps.
* Développer une fonction Python permettant la génération de fichiers CSV contenant les résultats des calculs, à savoir la position des satellites.
* Développer une fonction Python permettant d’effectuer des calculs de la position de l’utilisateur par époque par moindre carré simple en exploitant les pseudorange C1
* Autrement dit, Exécutez un ajustement des moindres carrés époque par époque pour déterminer l'emplacement et le décalage d'horloge du récepteur.

**Afin d’atteindre ces objectives, il est nécessaire de :**

* Comprendre la structure d'organisation des données et des informations dans les fichiers de navigation et d'observation rinex.
* Comprendre les algorithmes de calcul de la position des satellites à partir des élements du message de navigation.
* Comprendre les différentes approches du calcul de la position du récepteur GPS, à savoir les Méthodes de compensation

**Rinex :**

RINEX signifie Receiver Independent Exchange Format, il est stocké dans un format ascii et contient des observations GNSS.

Afin d’exploiter les données contenues dans les fichiers de navigation et d’observation, Il a été décidé que ces informations seraient stockées dans une structure de données de **dictionnaire python**. Un dictionnaire contient un ensemble de paires « clé-valeur » uniques qui permettent d'accéder rapidement aux informations, sans avoir à gérer des index de tableau désordonnés. L'implémentation n'est pas importante, il suffit de garder à l'esprit que les dictionnaires permettent de stocker les informations présentes dans les deux fichiers RINEX et d'y accéder rapidement ultérieurement avec des mots-clés significatifs.

**Le programme se compose de quatre parties principales :**

**Parse\_rinex\_navigation\_file** : englobe l’ensemble des fonctions nécessaire pour la lecture du ficher rinex de navigation et l’extraction de ses données.

**Parse\_rinex\_observation\_file** : englobe l’ensemble des fonctions nécessaire pour la lecture du ficher rinex d’observation et l’extraction de ses données.

**SatPos** : les fonctions mathématiques du calcul de la position du satellite qui prennent en argument les objets génères par Parse\_rinex\_navigation\_file et Parse\_rinex\_observation\_file

**Least\_squares** : fonction de de calcul et d’ajustement de la position de l’utilisateur par la méthode de compensation par moindre carré ,ainsi quelques fonctions de transformations géodésique .

**Main** : fonctions de lecture et traçage graphique des résultats, ainsi que programmation de l’interface console de l’application

**Calcul de la position des satellites :**

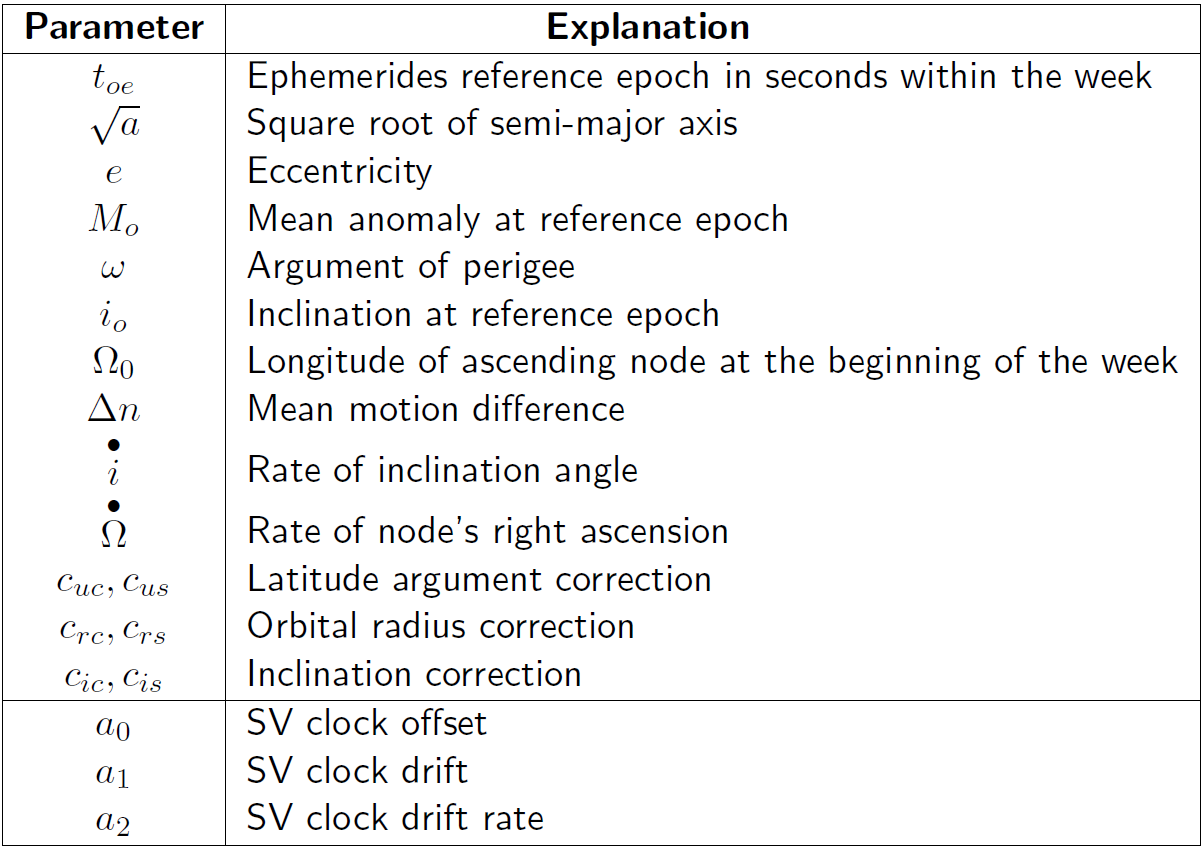
Pour calculer les coordonnées du récepteur, les coordonnées du satellite doivent d'abord être déterminées. Il s'agit d'un processus assez simple pour déterminer les coordonnées du satellite, les éléments képlériens du satellite sont diffusés dans le fichier de navigation. Chaque satellite de la constellation GPS effectue cette diffusion de tous les satellites GPS de la constellation et diffuse sa prédiction de position en orbite toutes les deux heures. Une fois que tous les 29 paramètres du fichier de navigation RINEX sont lus et stockés, le processus de détermination de la position du satellite se produit.

Le processus de détermination des positions satellites a été décrit en détail dans le document ICD. L'algorithme est défini dans la section 20.3.3.4.3 Algorithme utilisateur pour la détermination de l'éphéméride Pages 97-100. Les équations sont définies dans le tableau 20-IV du document ICD

Le tableau 1 fournit les paramètres des éphémérides de diffusion GPS pour calculer leurs coordonnées satellites à n'importe quelle époque d'observation. Ces paramètres sont renouvelés périodiquement (typiquement toutes les 2 heures pour le GPS) et ne doivent pas être utilisés hors du temps prescrit (environ 4 heures), car l'erreur d'extrapolation croît de façon exponentielle au-delà de sa période de validité.

L'algorithme fourni provient du [IS-GPS-200, tableau 20-IV]

NAVSTAR GPS Space Segment/Navigation User Interfaces <https://www.gps.gov/technical/icwg/IS-GPS-200M.pdf>

****

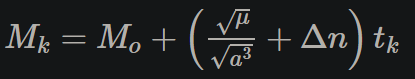
Afin de calculer les coordonnées du satellite à partir du message de navigation, l'algorithme fourni comme suit doit être utilisé.

• Calculer le temps tk à partir de l'époque de référence des éphémérides toe (t et toe sont exprimés en secondes dans la semaine GPS) : tk=t−toe

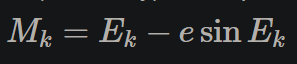
Si tk>302400 sec, soustrayez 604800 sec de tk.

Si tk<−302400 sec, ajouter 604800 sec.

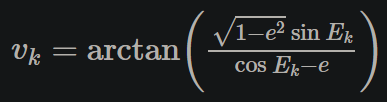
* Calculer l'anomalie moyenne pour tk,



* Résolvez (itérativement) l'équation de Kepler pour l'anomalie d'excentricité Ek :



* Calculez la véritable anomalie vk :



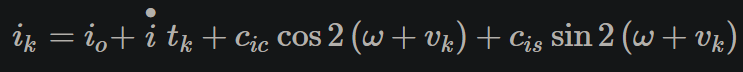
* Calculer l'argument de latitude uk à partir de l'argument du périgée ω, de l'anomalie vraie vk et des corrections cuc et cus :



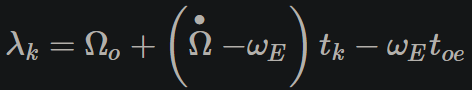
* Calculer la distance radiale rk en tenant compte des corrections crc et crs :



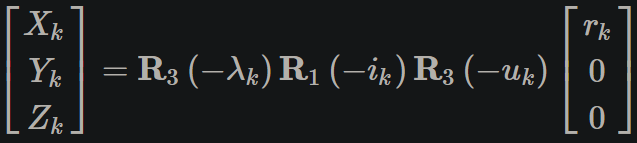
* Calculer l'inclinaison ik du plan orbital à partir de l'inclinaison io au temps de référence toe, et des corrections cic et cis :



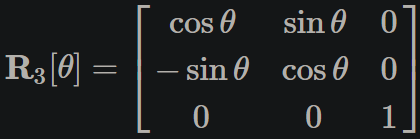
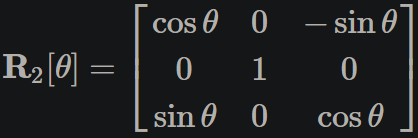
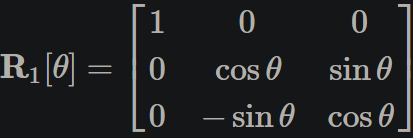
* Calculer la longitude du nœud ascendant λk (par rapport à Greenwich). Ce calcul utilise l'ascension droite du début de la semaine en cours (Ωo), la correction de la variation apparente du temps sidéral à Greenwich entre le début de la semaine et l'heure de référence tk=t−toe, et le changement de longitude de l'ascendant nœud à partir de la pointe de temps de référence :



* Calculer les coordonnées dans le référentiel TRS en appliquant trois rotations (autour de uk, ik et λk) :

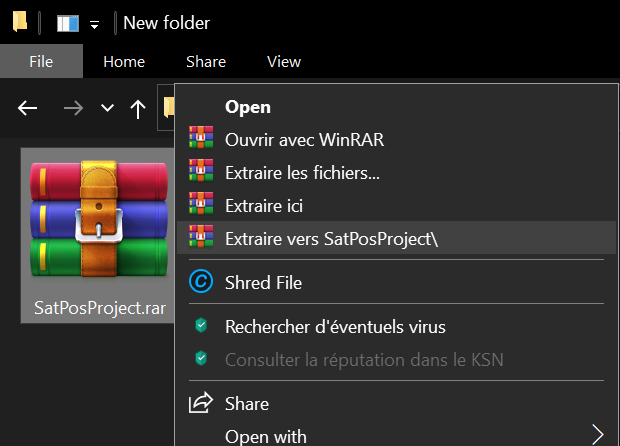


Avec :

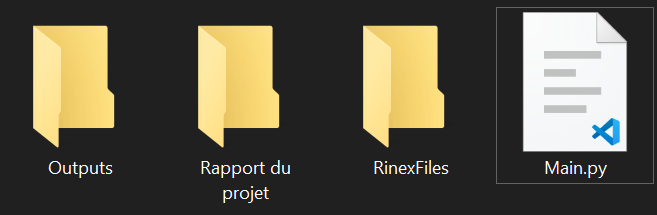


**Guide d’utilisation de l’application :**

1. **Télecharger et extraire le fichier compressé :**

****

1. **Le dossier contient un fichier python et trois sous-dossiers :**

****

**Main.py** : fichier python contenant le code de l’application

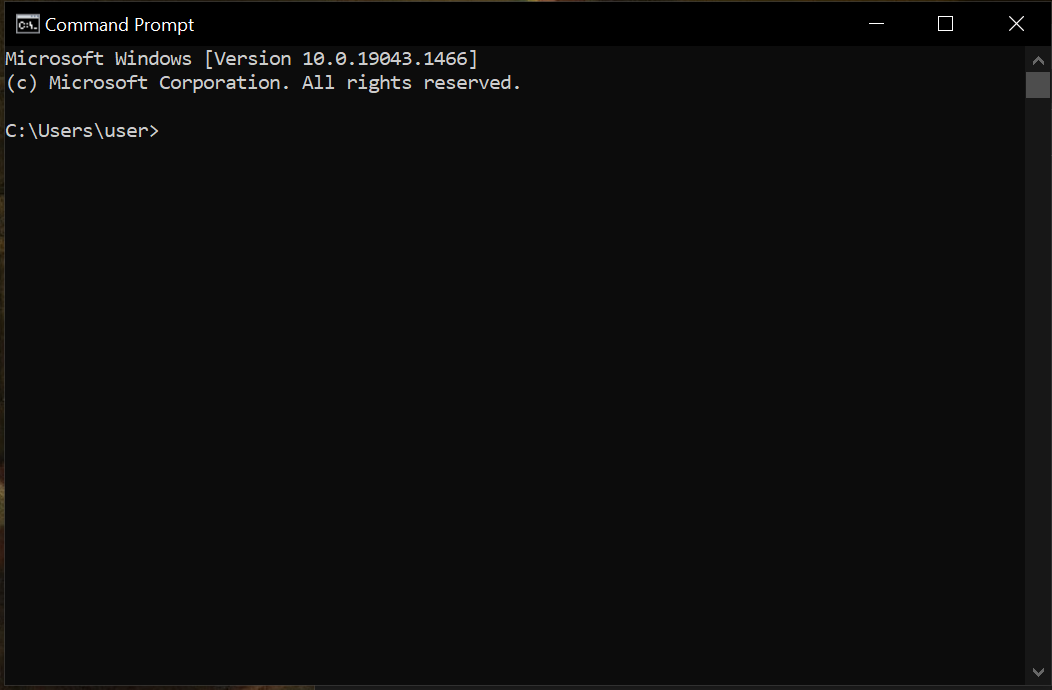
**Outputs** : les résultats aprées éxecution du code ( fichiers csv et txt)

**Rapport du projet** : rapports pdf et word

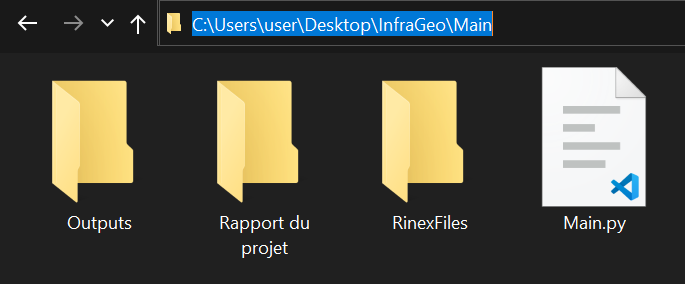
**RinexFiles** :fichiers rinex de navigation et d’observation de la IGS zimm corespondant

À la date 21/12/14 (zimm3480.21n zimm3480.21o)

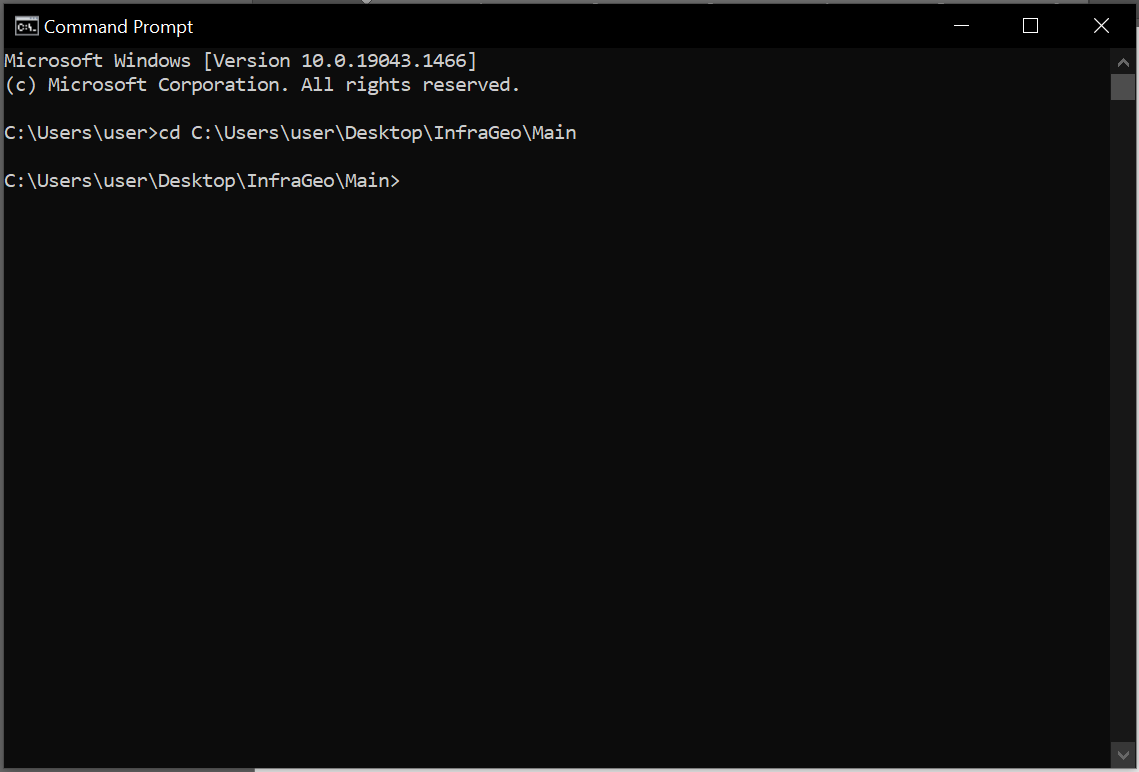
Lancer le code python soit par votre éditeur python soit directement dans le **cmd** comme il est montré ci-dessous



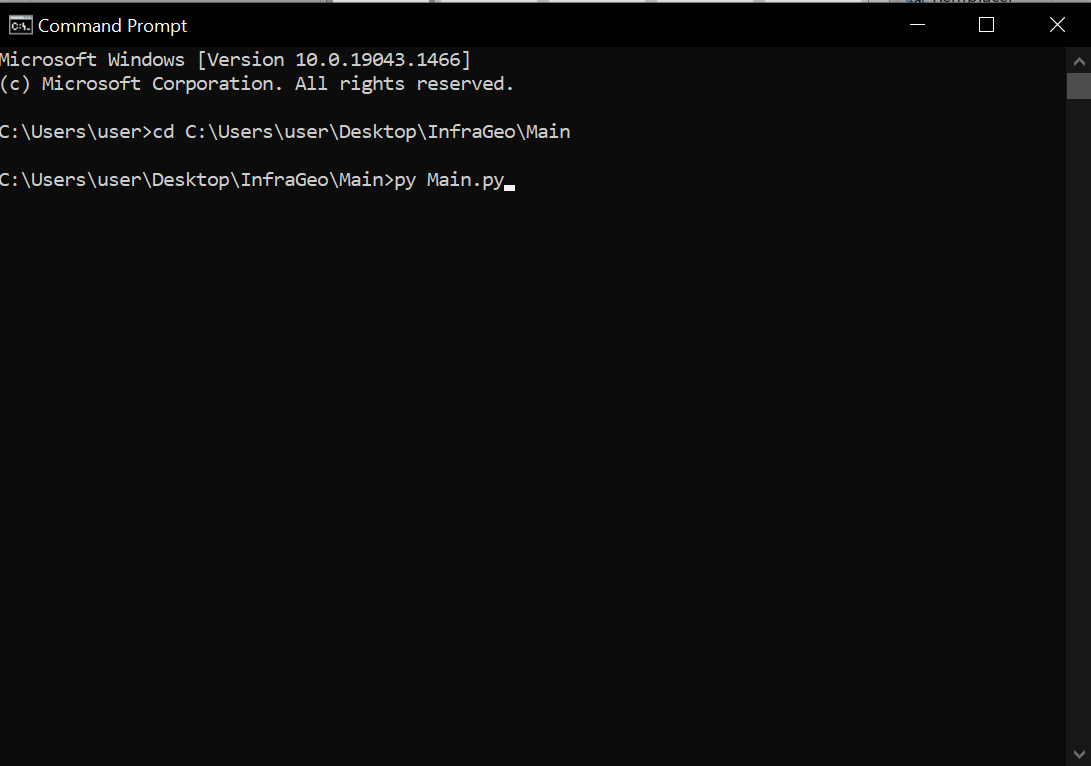
Assurez vous de changer le répertoire courant par la commande **cd**:

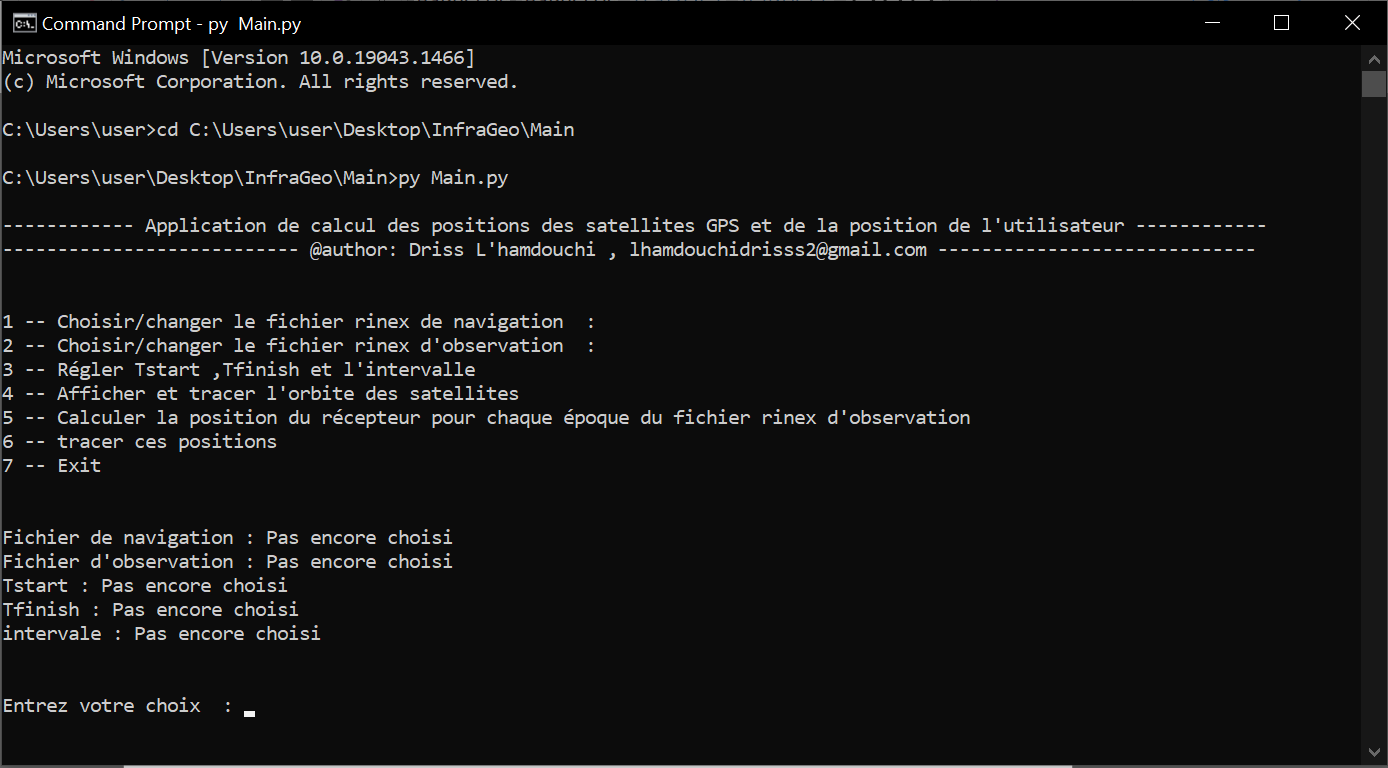
****

Copiez le chemin et utilisez la commande **cd**

****

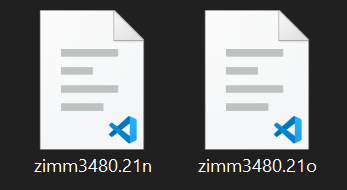
**L**ancer le code python par la commande : **py Main.py**

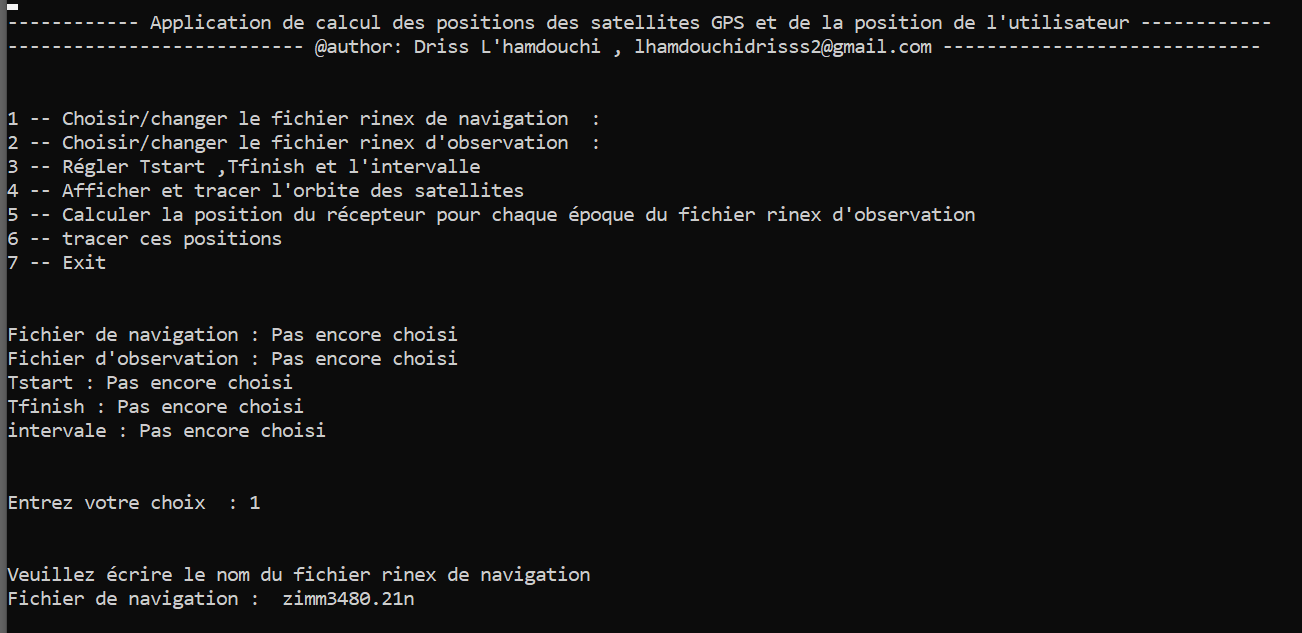
****

****

**Après suivissiez les étapes suivantes :**

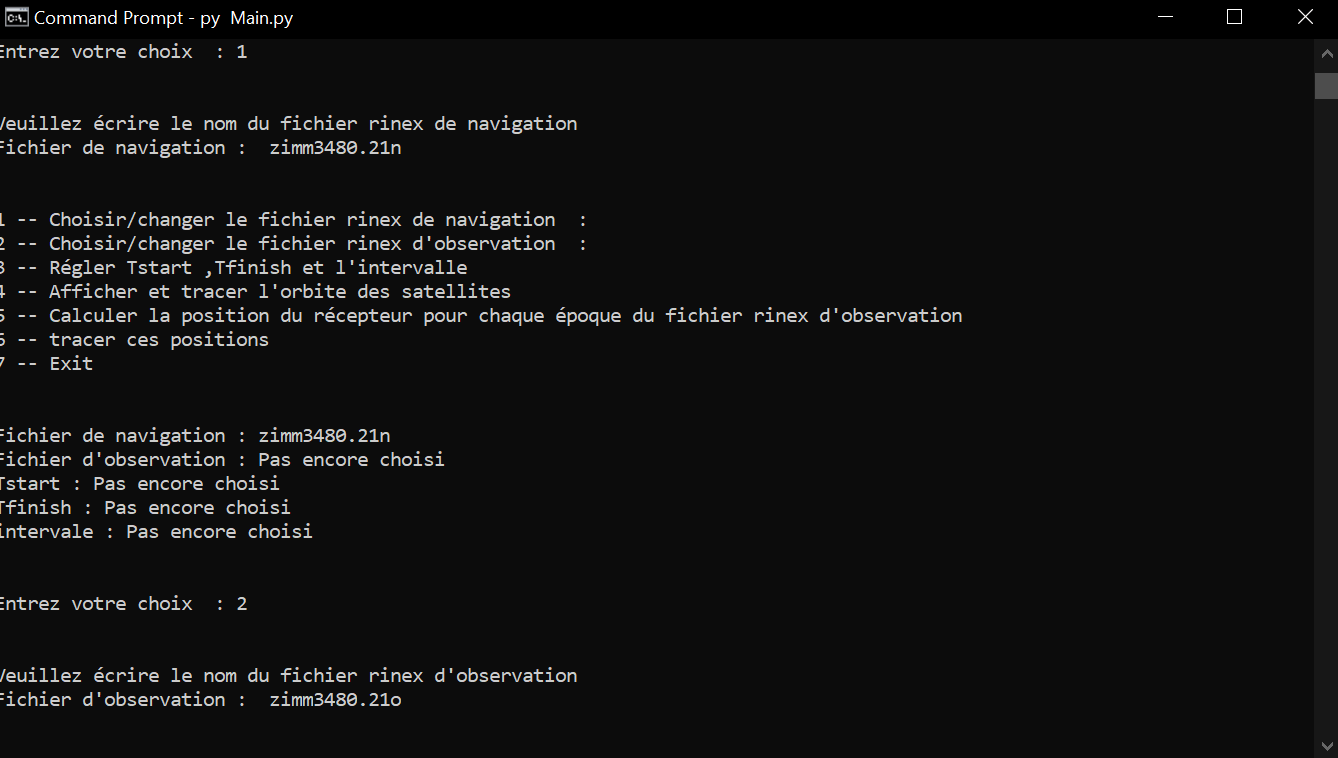
**Tapez 1 et indiquez le nom de votre fichier rinex navigation assurer cous que ce dernier existe déjà dans le dossier RinexFiles**

****

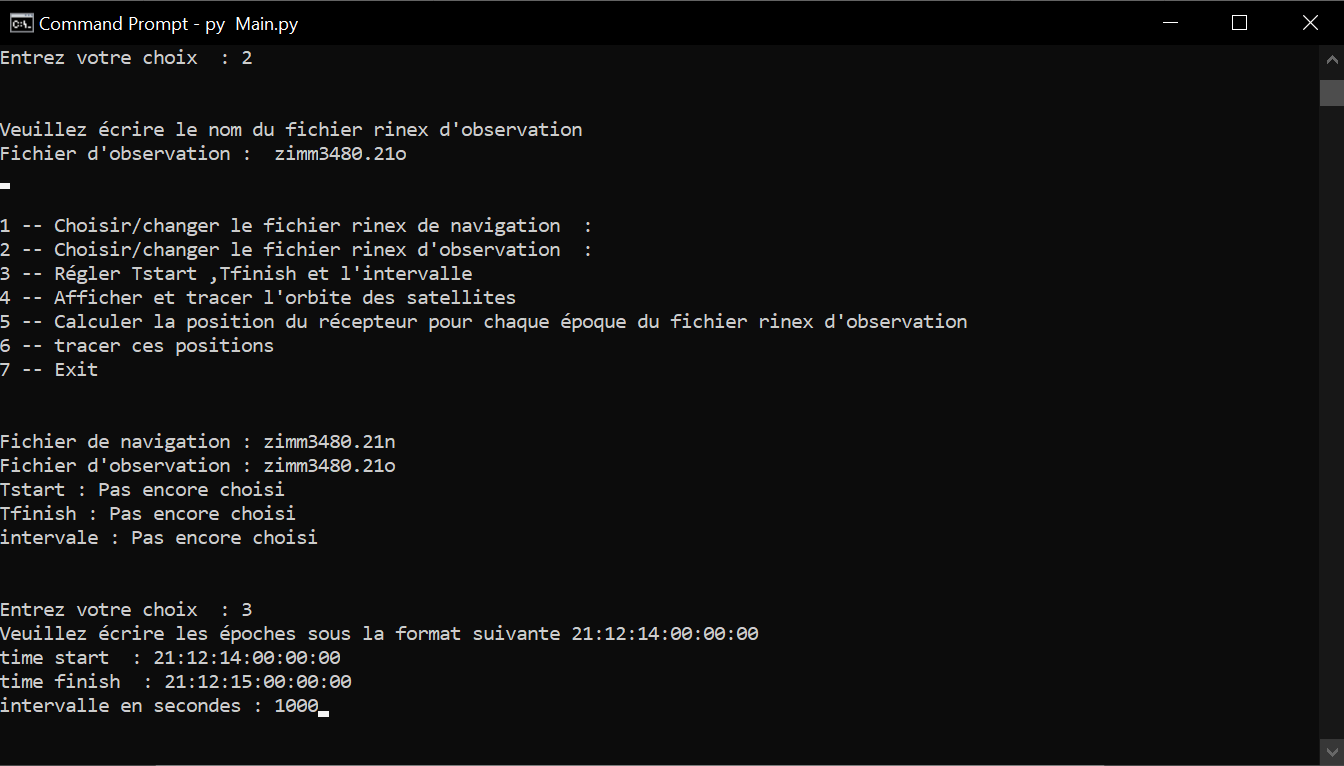
****

**Cliquez sur entrez et faites du même avec le fichier d’observation**

**Remarque : j’ai effectué des modifications sur le fichier d’observation zimm3480.21o, notamment j’ai extrait seulement la partie d’observation jusqu’à 6 heure du matin afin de ne pas alourdir l’application**

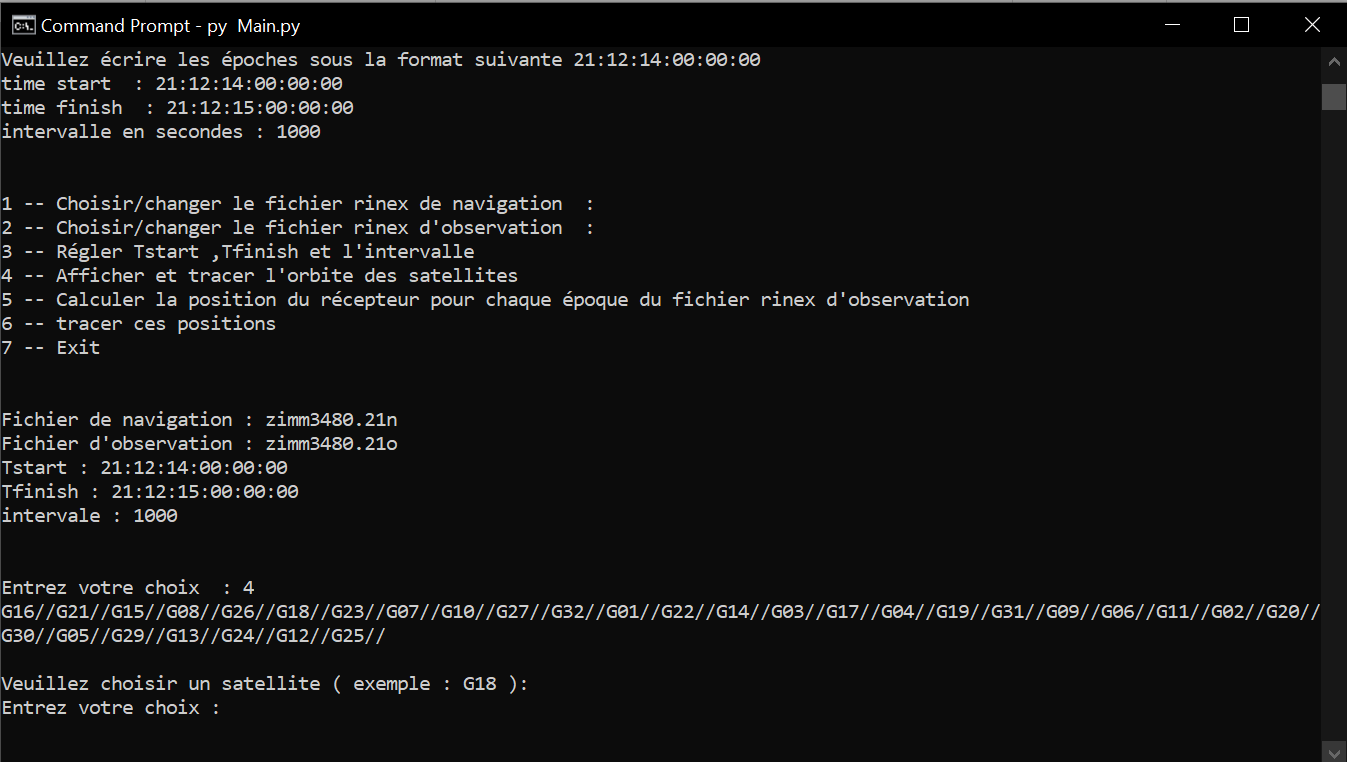
****

**Choisissez l’option 3 :**

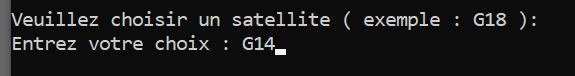
****

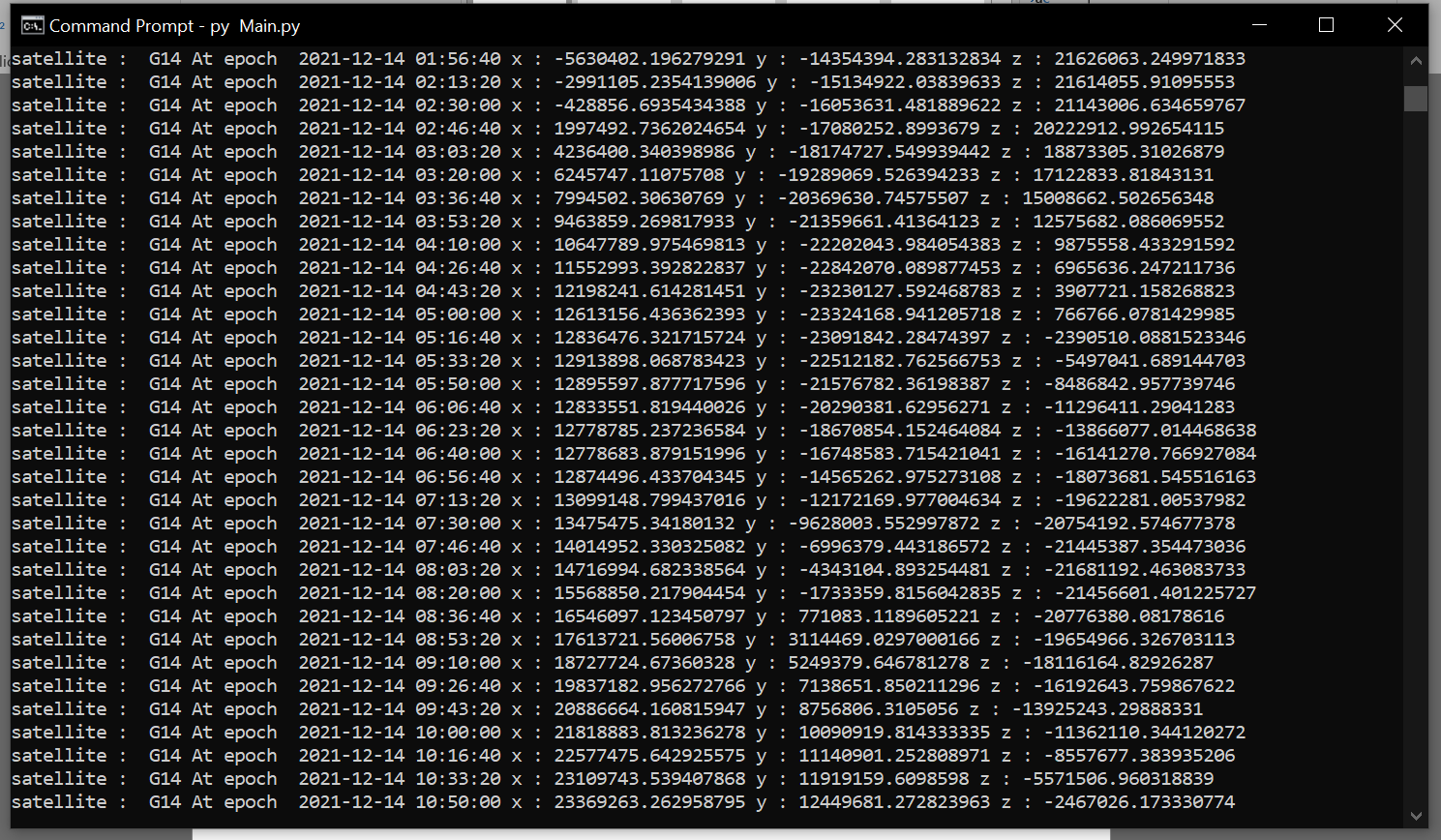
**Indiquez la première époque Time Tstart que vous voulez et la dernière Time Tfinish en respectant la forme comme il est indiqué ci-dessus. Intervalle est en seconde c’est le pas du temps entre une position et une autre**

**Choisissez l’option 4, le résultat est comme montré :**

****

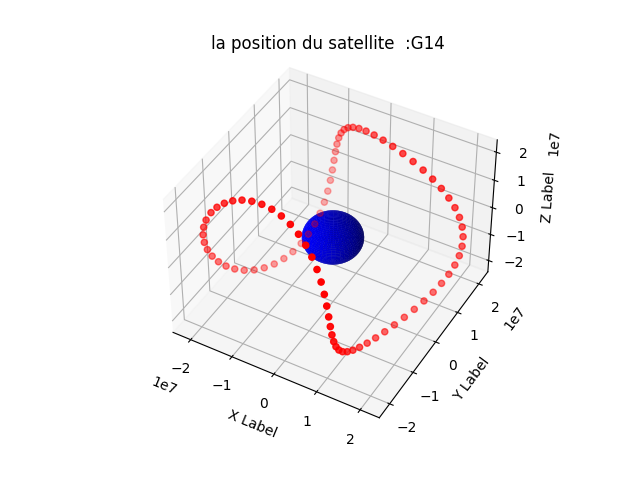
**Après choisissez le satellite que vous voulez visualiser l’orbite :**

****

****

**Le résultat est la position du satellite dans le référentiel ECEF entre Tstart et Tfinish avec le pas intervalle**

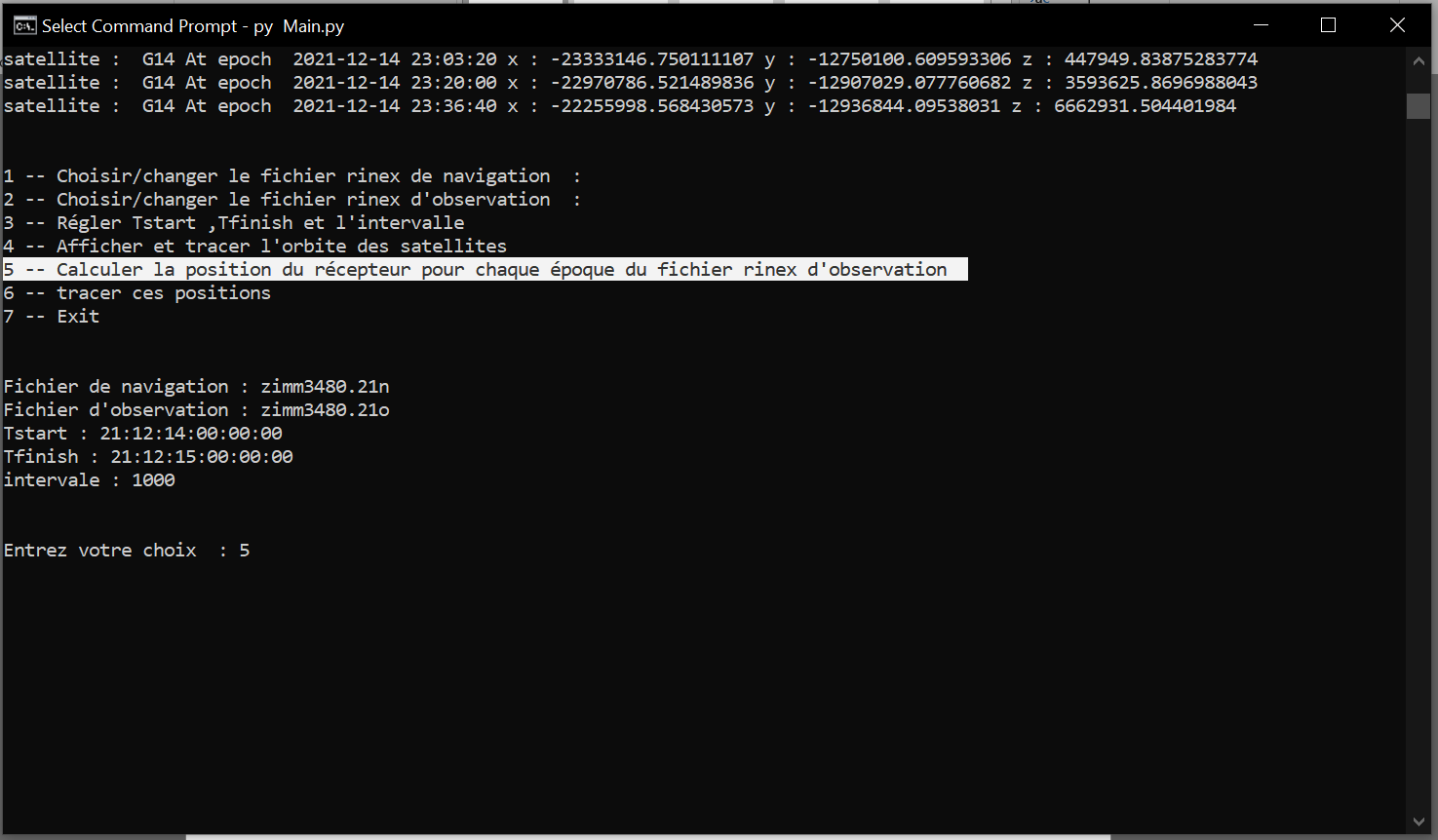
**L’application génère dans le dossier Output un fichier csv contenant les résultats**

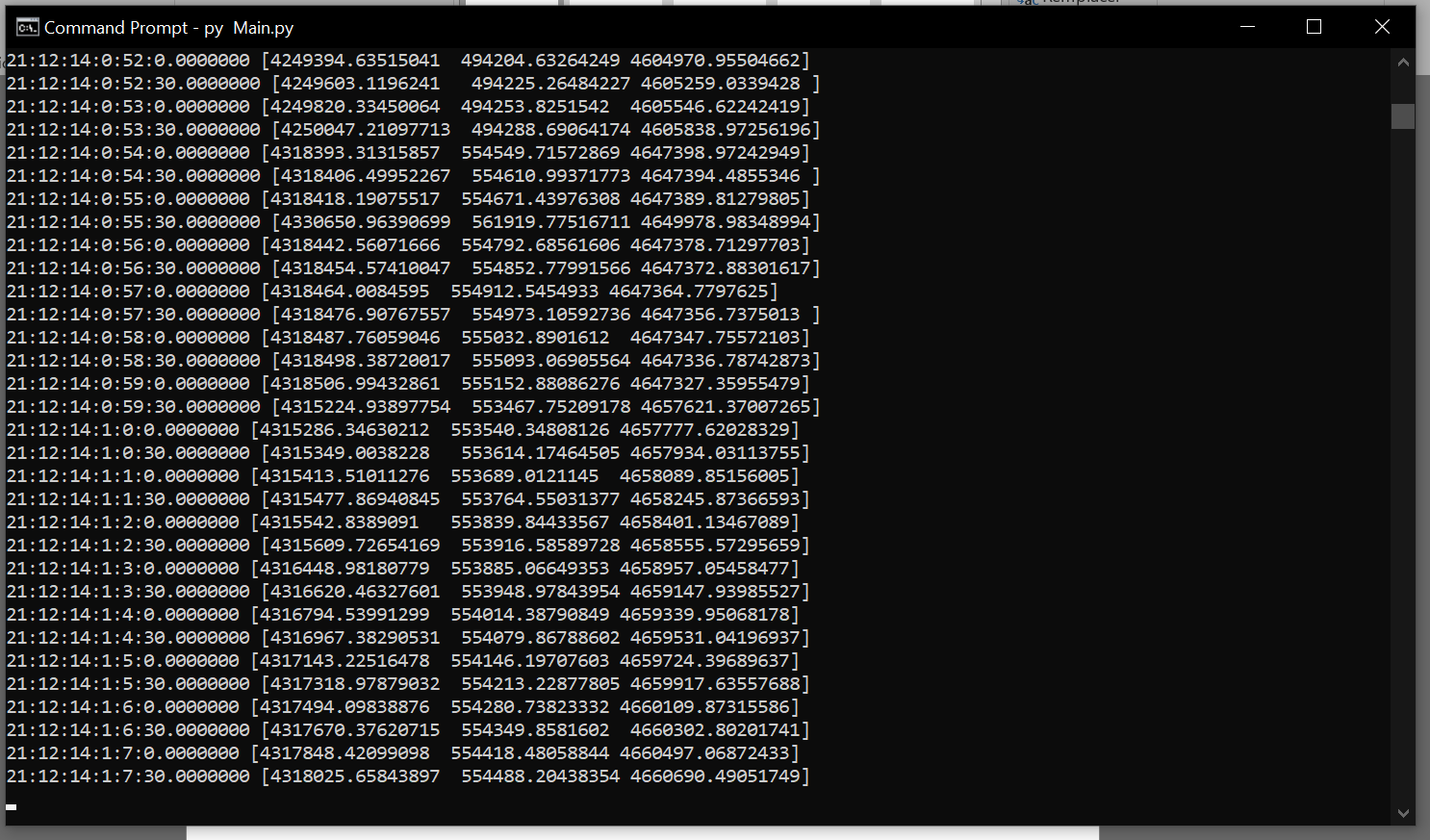


**Ainsi que l’orbite du satellite dans le référentiel ECEF qui est liée à la terre**

**La visualisation est un moyen de vérifier que le calcul est bien fait, c’est exactement la forme de l’orbite des satellites dans ce référentiel qu’est liée à la terre.**

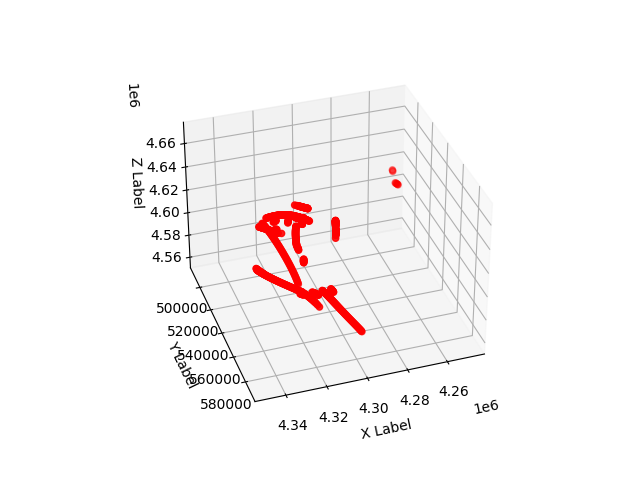
**Pour calculer la position de l’utilisateur choisissiez l’option 5 :**

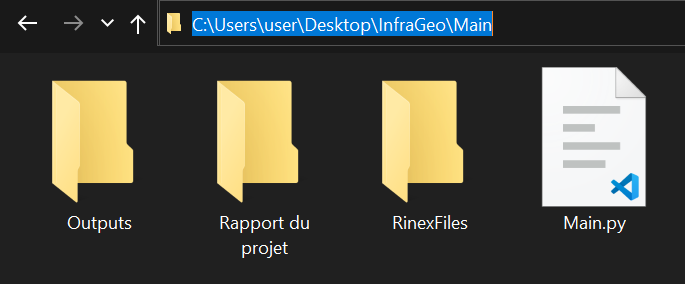
****

****

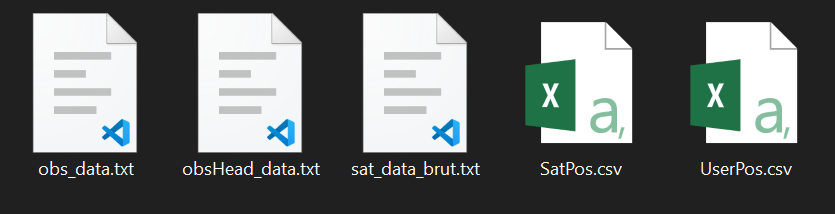
**L’application commence à calculer la position de l’utilisateur à chaque époque**

**Et génère dans le dossier output un fichier csv contenant les résultats**

**Pour visualiser ces position choisissez l’option 6**

****

**Dans le dossier Outputs vous trouverez les résultats de votre manipulation de l’application**

****

**Analyse des résultats et un commentaire général sur le fonctionnement et la phase de recherche et de programmation de l’application :**

Ce rapport implique tous les composants logiciels nécessaires pour calculer une solution de navigation GPS Standard Positioning Service. Le logiciel a été testé pour un cas test de données d'observation, bien qu'une position ait été sortie, la qualité de la solution n'était pas acceptable. Les bogues logiciels à l'origine de l'écart n'ont pas pu être déterminés sur la durée de ce projet en raison de contraintes de temps. S'il y avait plus de temps, le retard troposphérique du signal GPS pourrait également être mis en œuvre et testé ainsi que le modèle de Klobuchar pour déterminer le retard ionosphérique pour un récepteur à fréquence unique.

Enfin je veux remercier le prof pour ce devoir puisqu’il m’a vraiment aidé à approfondir mes connaissances à propos le fonctionnement du GNSS et notamment GPS, ainsi que mes compétences en programmation sur base de la langage Python, le temps était une contrainte, on rend le travail mais ce n’est pas fini pour moi, cette application va --inchallah- avoir des révisions et des améliorations en bien introduisant les corrections

Et en cherchant ou réside exactement le problème. Si vous avez des questions sur le n’hésiter pas de m’envoyer un mail **lhamdouchidriss2@gmail.com**