# Planning\_Maker

Pour automatiser la station de suivi de satellite, il est nécessaire d’avoir un algorithme capable de planifier une journée complète de visualisation, sans avoir de fenêtre qui se superposent mais aussi que la sélection dépende de la priorité de chacun des satellites.

La fonction Planning\_Maker est construite sur la base d’une boucle infinie qui s’arrête sous une certaine condition. Les raisons du choix de cette boucle infinie et de la condition d’arrêt seront détaillées plus tard pour plus de clarté. Cette boucle appelle deux fonctions à la suite.

# chooseWindows

Dans le but de planifier la journée, cette fonction vient traiter toutes les fenêtres les unes après les autres, en fonction de « l’état » de chaque fenêtre (state dans le programme) :

* IDLE : Désigne un passage en attente de traitement ;
* SELECTED : Désigne une fenêtre déjà sélectionnée ;
* EXCLUDED : Désigne une fenêtre exclue de ta planification. Une fenêtre dans cet état-là n’est pas traitée par le programme directement mais peut quand même être rajoutée dans le planning par la suite ;
* OVERLAPS\_PREVIOUS : Désigne une fenêtre dont le temps de début est inférieur au temps de fin de la fenêtre précédente.

## Cas IDLE ou SELECTED

Ce

## Cas OVERLAPS\_PREVIOUS

//Faire screen avec le problème de sélection

# determineAddableWindows

Lors de la conception initiale de cet algorithme, seule la première fonction devait être utilisée. Cependant, en générant le planning de visualisation, une erreur dans l’algorithme a été mise en évidence. Lors de la sélection des fenêtres, certaines n’étaient pas sélectionnées alors qu’elles auraient dû l’être (voir Figure)

//Faire screen avec le problème de sélection

Pour pallier ce problème, une seconde fonction a été ajoutée. Le but de cette dernière est de repérer les passages utilisables au cours de la journée.