IPF - Rapport du projet

Géraldine Bailleul

Pour le vendredi 3 mai 2019

I. Présentation du sujet

Le but de ce projet est de prendre un message en entrée et de donner la suite d'instructions permettant de piloter une (ou plusieurs) antenne(s) afin d'envoyer ledit message.

Nous disposons de roues comportant les 26 lettres de l'alphabet et du caractère espace. Toutes les roues sont initialement positionnées sur le caractère espace. Les commandes disponibles pour manipuler une antenne sont :

- « N » pour passer au caractère suivant ;
- « P » pour revenir au caractère précédent ;
- « E » pour envoyer la commande d'émission du caractère courant.

Le programme devra lire deux lignes sur leurs entrées standards : la première ligne comportant un entier *n* donnant le nombre d'antennes, et la deuxième comportant le message lui-même.

II. Phase 1

Dans la phase 1, nous ne traitons que le cas avec une seule antenne.

Dans ce cas, on parcourt la liste des caractères composants le message. Pour chaque caractère, on cherche à déterminer la suite de commandes nécessaire pour l'envoyer, tout en prenant en considération la position courante de la roue. Ainsi, la fonction *code_attendu()* prend en argument le caractère à envoyer ainsi que l'état actuel de la roue, et calcule l'écart entre le caractère que l'on cherche à écrire et le caractère sur lequel est positionnée la roue. À cette fin, elle utilise les fonctions *letter_to_number()* et *number_to_letter()* qui permettent de convertir un caractère en nombre et inversement, ce qui permet de déterminer l'écart. Enfin, elle prend l'écart le plus court entre avancer et reculer, puis génère grâce à la fonction *create_liste()* la liste des commandes permettant de positionner la roue sur le bon caractère puis de l'envoyer.

III. Phase 2

Ici, le nombre de roues n'est plus restreint.

Comme précédemment, on parcourt le message à envoyer caractère par caractère. La différence réside ici dans le fait que pour chaque caractère du message, on cherche à déterminer quelle roue requiert le moins de rotations afin d'être positionnée dessus (à l'aide de la fonction *dist_min()*). Une fois cela fait, on passe sur cette roue et on envoie le caractère (cela prend 1s pour changer de roue contre 5s pour tourner une roue), avec la fonction *code_attendu()*.

IV. Résolution finale

Pour terminer, on convertit la *list of char* en *string* avec la fonction *convert_to_string()*, puis on renvoie la chaîne de caractères sur l'entrée standard avec *fonction_finale()*.

V. Difficultés rencontrées

Les principales difficultés rencontrées concernaient des problèmes de typage. Plus particulièrement, quand j'ai dû insérer les chiffres dans la phase 2 alors que je travaillais avec des listes de caractères, j'ai dû recourir à des fonctions « intermédiaires » pour gérer mes problèmes de typage (*number_to_list_of_char()* dans ce cas).

Pour la concaténation des listes de listes, j'ai utilisé les propriétés du module *List* (*List.concat*).