Évaluation Haskell : Manipulation de Listes et Commandes

1. Crée une fonction `swap_first_elements` qui prend une liste et échange les deux premiers éléments.

Si la liste est vide ou ne contient qu'un élément, la fonction doit renvoyer la liste inchangée.

2. Crée deux fonctions `rFlags` et `rrFlags` :

- `rFlags` décale la liste d'un élément vers la gauche, déplaçant le premier élément à la fin.
- `rrFlags` décale la liste d'un élément vers la droite, déplaçant le dernier élément au début.

3. Crée une fonction `moveOtherList` qui prend deux listes et une chaîne de caractères représentant une commande ("pa" ou "pb").

La fonction doit déplacer l'élément en tête d'une liste vers l'autre selon la commande donnée.

4. Crée une fonction `dispatchFlags` qui prend deux listes et une commande et applique l'opération correspondante à la bonne liste

ou à l'ensemble des deux listes.

5. Crée une fonction `checkPushswap` qui prend une liste de commandes et deux listes d'entiers.

La fonction doit appliquer les commandes successivement aux deux listes et renvoyer les listes modifiées.

6. Crée une fonction `isSort` qui vérifie si une liste d'entiers est triée en ordre croissant.

7. Crée une fonction `checkResult` qui prend deux listes et retourne un booléen.

La fonction vérifie si la première liste est triée et la deuxième liste est vide.

8. Crée une fonction `printResult` qui affiche "OK" si le tri est correct, ou "KO" suivi des deux listes si le tri est incorrect.

9. Crée une fonction `checkArgs` qui vérifie que les commandes données sont valides.

10. Crée une fonction `errorManagement` qui vérifie la validité des arguments et des commandes.
    Si une erreur est détectée, la fonction doit quitter le programme avec le code d'erreur 84.

11. Implémente la fonction `main` qui récupère les arguments en ligne de commande, lit les commandes depuis l'entrée standard,
    gère les erreurs, et affiche le résultat final.

Makefile :

1. Crée un Makefile pour compiler et nettoyer le programme Haskell avec GHC.
2. Le Makefile doit inclure les cibles `all`, `clean`, `fclean`, et `re`.

Exemple de Makefile :

```
HC      = ghc
SRC     = pushswap_checker.hs
NAME    = pushswap_checker

all: $(NAME)

$(NAME):
    $(HC) $(SRC) -o $(NAME)
```

```
clean:
	$(RM) *.hi *.o

fclean: clean
	$(RM) $(NAME)

re: fclean all

.PHONY: $(NAME) all clean fclean re
```