CNAM 2018/2019

Projet NSY103 linux

Auditeur : Charles GERMAIN

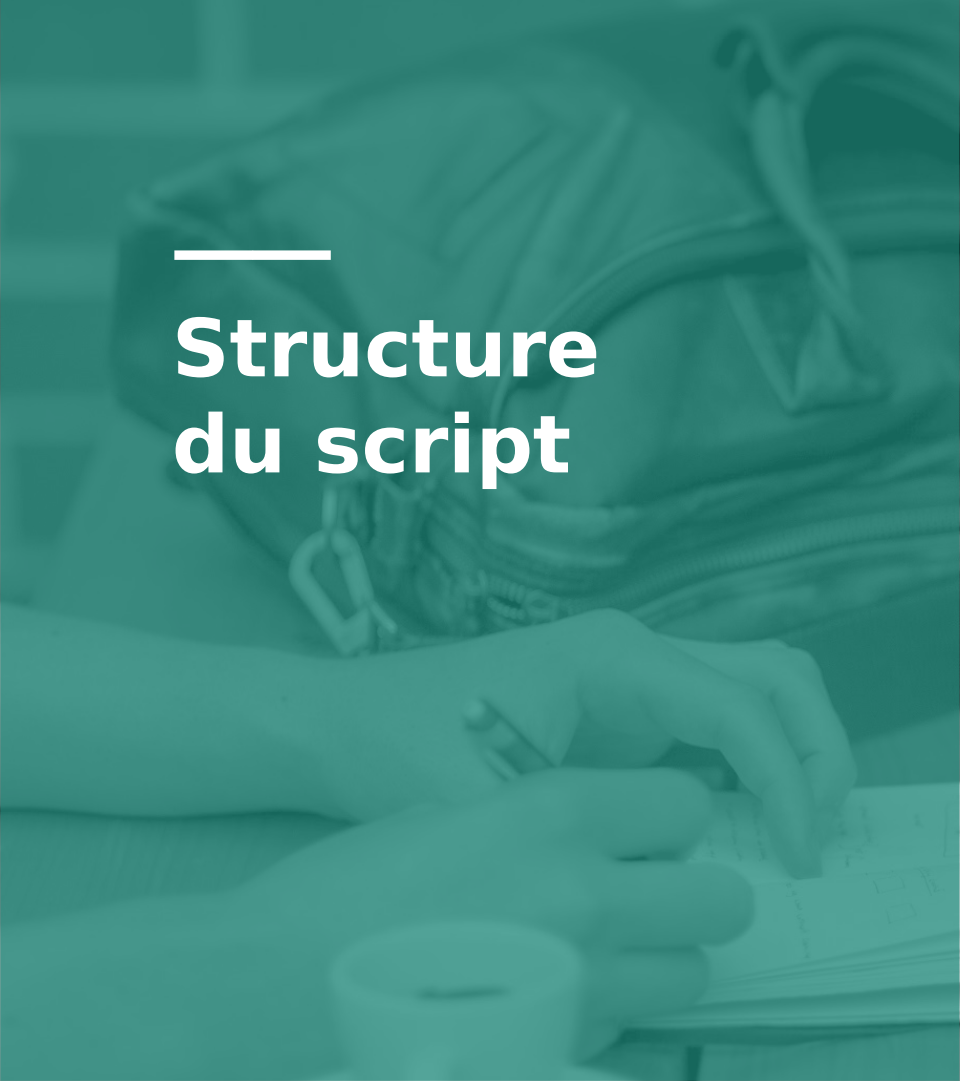
Lien GitHub :
https://github.com/CharlesAuCNAM/conv_vers_romain

Résumé

L’énoncé

1) Le projet doit être écrit, en langage shell (bash, ksh etc...) ou en langage C.
Le code doit être au minimum contenu dans une page de format A4.
Au code vous joindrez une présentation powerpoint descriptive de votre projet d'au minimum trois diapositives.

2) Une fois terminé, mettre votre projet sur la plateforme https://github.com, dans un dossier nommé NSY103_2018_2019. Dans un sous dossier du dossier nommé NSY103_2018_2019 vous créerez un dossier avec le nom de votre choix qui contiendra le projet.



Structure du script

Le script “*chiffre_romain.sh*” transcrit un chiffre entier en chiffre romain avec une interaction préalable avec un utilisateur.

Use case :

Un utilisateur a le choix de saisir manuellement un chiffre entier, ou le générer automatiquement. Cet entier est ensuite traduit en chiffre romain.

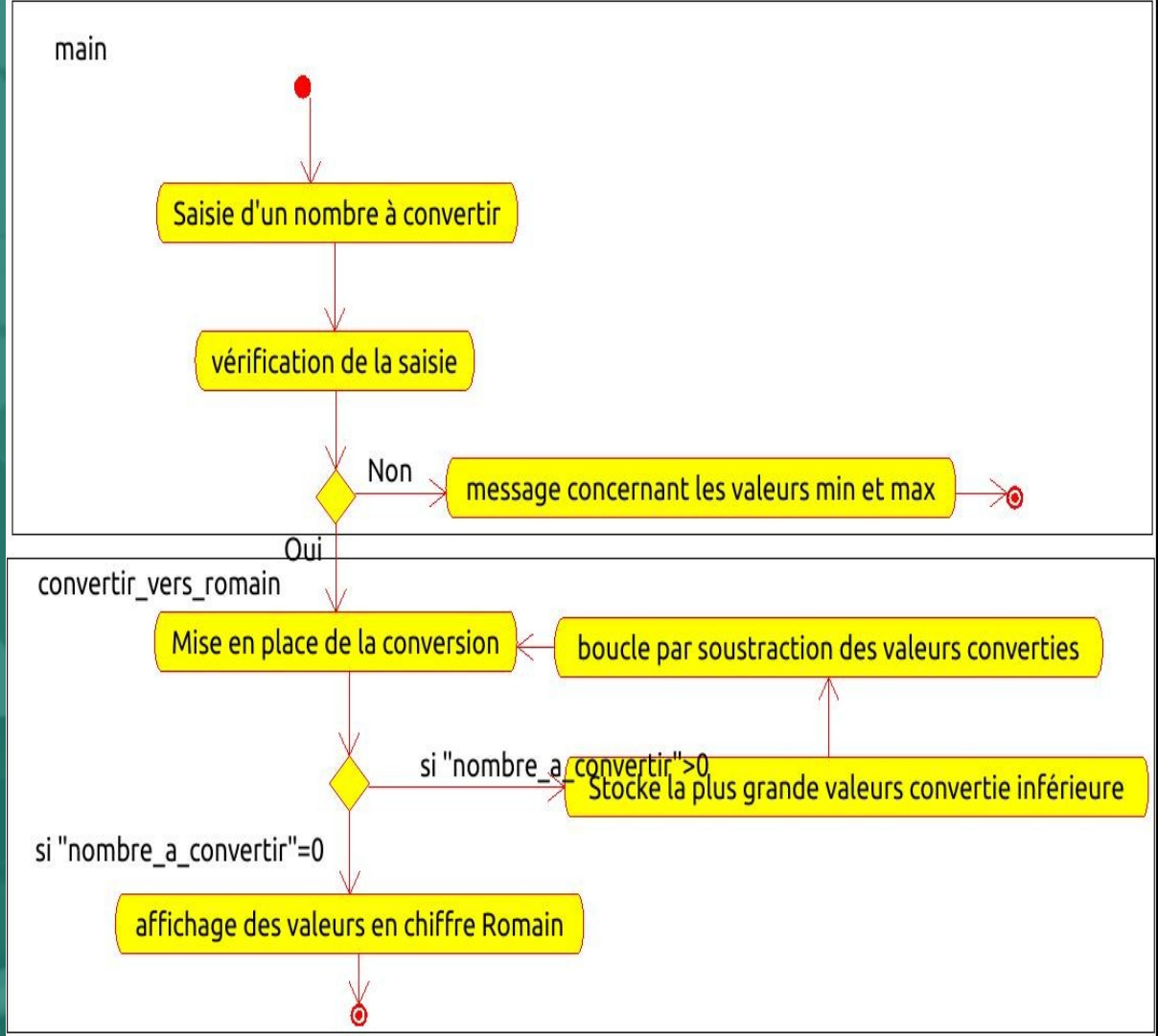
Spécificités & contraintes :

Le script contrôle la saisie de l’utilisateur
Limite du nombre de caractère à 8 (temps de calcul)
Possibilité de rejouer le script à l’infinie

Environnement technique :

Script shell développé en bash
Développé dans l’environnement Linux de la distribution Ubuntu (18.1)

Structure du script



L’interaction avec l’utilisateur 01

L’utilisateur peut entrer un chiffre entier manuellement en le passant comme argument.

Fichier : /home/charger/conv_vers_romain.sh Page 1 sur 1

```
#!/usr/bin/env bash

convertir_vers_romain(){
    local nombre_a_convertir=$1
    local valeurs_en_romain=""
    #Nous définissons un tableau de conversion
    declare -a
tableau_conversion=([1000]="M" [900]="CM" [500]="D" [100]="C" [90]="XC" [50]="L" [40]="XL" [10]="
    #et un tableau d'équivalence permettant de réaliser les conversions
    declare -a valeurs_a_substituer=(1000 900 500 400 100 90 50 40 10 9 5 4
1)

    #On lance la boucle de filtrage tant qu'on a pas retiré tous les chiffres
    while [ "${nombre_a_convertir}" -ne 0 ]; do
        val=0
        #On restitue les valeurs en chiffre(s) romain et on l'affiche (en décalant
vers la gauche)
        for val_test in ${valeurs_a_substituer[*]}; do
            #on test la valeure dès qu'on est en dessous on peut commencer à
sortir les valeurs (break, ici fait office de ceinture de sécurité pour ne pas
tester les valeurs plus faibles dès le début)
            (( "${val_test}" <= "${nombre_a_convertir}" )) && { val=${val_test} ;
break ; }
        done
        #ici s'opère la conversion de valeure en romain, on va chercher dans notre
tableau la correspondance
        valeurs_en_romain+=${tableau_conversion[$val]}
        #une fois la partie élevée du tableau inférieure, on la soustraie
        ((nombre_a_convertir-=val))
    done
    echo "${valeurs_en_romain}"
}

main(){
    #On fixe un nombre maximal que l'on ne peut pas dépasser
    local nb_max=3999
    #On teste la valeure pour s'assurer qu'elle corresponde bien à l'interval
proposé.
    if [ "$#" -ne 1 ] || [[ "${1}" -lt 1 || "${1}" -gt "$nb_max" ]]
    then
        echo "Le script doit s'exécuer de la façon suivante: conv_vers_romain.sh
<chiffre à convertir> cette valeur doit etre comptise entre 1 and $nb_max."
        exit 0
    else
        #dès lors, on peut commencer la conversion
        convertir_vers_romain "${1}"
    fi
}

main "$@"
```

L'interaction avec l’utilisateur

- ➔ Structure conditionnelle pour la saisie manuelle (IF)
- ➔ Saisie manuelle
  - ◆ Contrôle du type et du nombre de caractère
  - ◆ Utilisation de tableaux de conversion
  - ◆ Utilisation de boucle WHILE pour réaliser la conversion
  - ◆ Utilisation de BREAK

La relance du script

- ➔ Structure conditionnelle IF pour créer l'alternative
- ➔ Exécution du script ./conv_vers_romain.sh <nombre à convertir>
- ➔ Boucle récursive et tableau réutilisable pour d'autres applications (exemple : message codé simple ou tableau d'équivalence de conversion,…)

Conclusion

L’algorithme de traduction est récursif, optimisé et commenté

Le contrôle des saisies utilisateurs est fait en 2 fonctions :
- contrôle de la valeur (main)
- conversion de la valeur (convertir_vers_romain)

MERCI POUR VOTRE ATTENTION