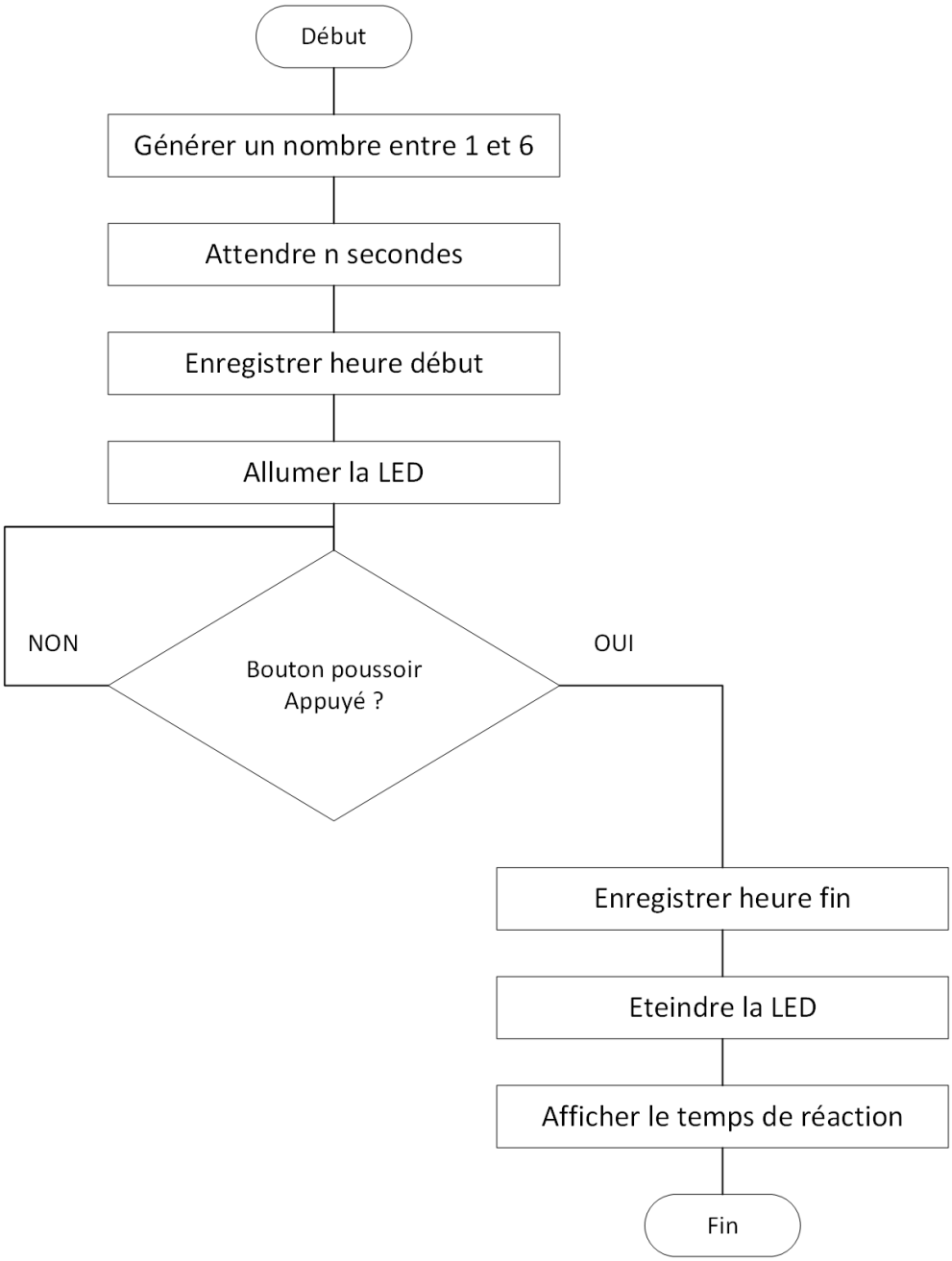
Mesure du temps de réaction

https://www.tuteurs.ens.fr/unix/shell/fonction.html

Cahier des charges : Un menu permet de choisir si on veut jouer ou sortir du script. Si on joue, la LED s'allume de façon aléatoire (entre 1 et 6 secondes). Le joueur appuie sur un bouton poussoir dès qu'il voit la LED allumée. Le programme affiche son temps de réaction.

Organigramme de la partie « jeu »



Ouvrez l'éditeur de texte `nano` :

```
pi@raspberrypi ~ $ nano cde_LED_04.sh
```

Saisissez le script destiné à mesurer le temps de réaction entre l'allumage d une LED connectée sur le GPIO 20 et l'appui sur un bouton poussoir connecté sur le GPIO16.

```
#!/bin/bash
# Script cde_LED_04.sh
# Mesurer le temps de réaction entre l allumage d une LED
# et l appui sur un bouton poussoir
#
# Au lancement du programme (touche 1) la LED est éteinte
# Elle s allume après un temps aléatoire (de 2 à 6 secondes)
# Le joueur appuie sur le bouton poussoir pour valider
# La touche  1  permet de rejouer
# Le programme s arrête en appuyant sur la touche 2 (quitter)

# Effacer l'écran
clear

# Rendre le répertoire /sys/class/gpio actif
cd /sys/class/gpio

############### Configuration port GPIO du bouton poussoir

# Créer l accès au port GPIO 16
# Pour lire l'état du bouton poussoir
# Ne rien faire s il existe déja
if [ -d "gpio16" ]; then
 echo "gpio16 existe dejà"
else
 echo "gpio16 : Création"
 echo 16 > export
fi

# Rendre le répertoire gpio16 actif
cd gpio16/

# Configurer le port GPIO 16 en entrée
# Normalement il y est par défaut… mais on ne sait pas ce qui a pu se passer avant
echo in > direction


############### Configuration port GPIO de la LED

# revenir dans le dossier /sys/class/gpio
cd ..

# Créer l accès au port GPIO 20
# Pour commander la LED
# Ne rien faire s il existe déja
if [ -d "gpio20" ]; then
 echo "gpio20 existe dejà"
else
 echo "gpio20 : Création"
 echo 20 > export
fi

# Rendre le répertoire gpio20 actif
cd gpio20/

# Configurer le port GPIO 20 en sortie
echo out > direction

# Eteindre la LED
echo 0 > value
```

```
# Définition de la fonction  jeu
# Cette fonction contient la totalité du jeu
# Génération d un nombre aléatoire entre 2 et 6
# Allumage de la LED
# Attente du bouton poussoir
# Affichage du temps

jeu () {

    # Générer un nombre aléatoire entre 1 et 6
    min=1
    max=6
    number=$[($RANDOM % ($[$max - $min] + 1)) + $min]
    # Attendre le nombre de secondes
    sleep $number

    # Enregistrer le temps au début du jeu
    debut="$(($(date +%s%N)/1000000))"

    # Allumer la LED
    echo 1 > /sys/class/gpio/gpio20/value

    # Lire l etat du bouton poussoir
    z="$(cat /sys/class/gpio/gpio16/value)"

    # Boucler tant que le bouton poussoir n est pas appuyé
    while [ $z -eq 0 ]
    # Le bloc inclus entre do et done est exécuté
    # par la boucle while

    # Début du bloc d instructions
    do
    # On lit l état du bouton poussoir
        z="$(cat /sys/class/gpio/gpio16/value)"
    # le bouton poussoir a été appuyé, on peut continuer
    done

    # Enregistrer le temps à la fin du jeu
    fin="$(($(date +%s%N)/1000000))"

    # Calcul du temps
    temps=$(($fin-$debut))
    echo "Temps de réaction en millisecondes : $temps"

    # Pour utiliser l'affichage de grande taille (banner)
    # Il faut installer sysvbanner :  sudo apt-get install sysvbanner
    # banner $temps

    # Eteindre la LED
    echo 0 > /sys/class/gpio/gpio20/value

    # Attendre 2 secondes
    sleep 2
}

############################  Boucle principale du programme ############################

# Initialiser la variable x à 1
x="1"
```

```
# Boucle while : cette boucle s exécute tant que
# la condition est vérifiée
# ici la condition est toujours vraie :


while [ $x -gt 0 ]
# Le bloc inclus entre do et done est exécuté
# par la boucle while

# Début du bloc d instructions
do

# Définition de la fonction    jeu

############################  Affichage du Menu

select item in "- Jouer -" "- Sortir -"
      do
              echo "Vous avez fait le choix $REPLY : $item"
              case $REPLY in
                      1)
                              # Appel de la fonction jeu
                              echo "Attention la LED va s'allumer !"
                              jeu
                              ;;
                      2)
                              echo "Fin du script"
                              exit 0
                              ;;
                      *)
                              echo "Choix incorrect"
                              ;;
              esac


    # La commande sleep suspend l exécution du programme
    # pendant la durée spécifiée (en secondes)
    sleep 1
    clear
    echo "1) - Jouer -"
    echo "2) - Sortir -"
    # Fin du bloc d instructions
    done

done
```