*.*

Lancer un autre processus:

* Programme = descirption
* Processus = instance d’exécution
* Dans unix, pour changer de programme, on fait un autre appel system ; ***exec***
  + Execv -> on donne nom du programme à utiliser + tableau de paramètres
  + ExecL -> on donne le programme et on met les paramètre un à un
  + Le but d’exec est de remplacer le programme existant.
    - Remise à 0, variables comprises
  + Le problèmes des fichiers ouverts lors d’un exec :
    - Unix garde les mêmes fichiers ouvert. Le processus conaitra toujours les fichiers ouvert.
    - Permet d’avoir un même programme quel que soit l’input.

Dans main, il y a comme parametres :

* Int argc,
* Char \*\* argv / char \* argv[]
* Il y a un troisième : \*envp[]
  + Tout ces arguments sont des arguments ***positionel***.
    - On y accède en fonction de l’ordre des arguments rentrés
    - Le positionel est plus rapide.
  + La seconde méthode, ce sont les parametres par ***keyword***. C’est (\*envp[]) qui contient ces informations là.
    - Exemple :
      * HOME = …
      * IPC = …
    - Permet de ne pas devoir gérer l’ordre. Il envoie donc toutes les informations sous formes de tableaux. Plus long car en moyenne on compare le nombre équivalent à la moyenne du tableau

Arreter le processus :

Pour terminer l’appel system, on fait un ***exit***. Exit(valeur)

* Dans la philo Unix,
  + Exit ( 0 ) 🡪 tout ok
  + Exit ( autre ) 🡪 il y a un problème.
    - Cette gestion de correspondance numero/problème est propre au programmeur.
    - On peut également se référer à la doc d’Unix pour en savoir plus
      * <https://shapeshed.com/unix-exit-codes/>
    - Cette information est destiné au processus père

Attendre la réponse d’un processus :

* Wait
  + Attend qu’un fils se termine.
  + Donne l’information :
    - Exit du fils
    - Et son PID
    - Cause de la mort (crash du processus)