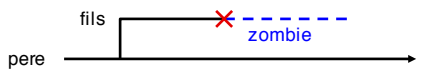
Systèmes d'Exploitation
l'Examen de 2008-2009

Olivier Dalle
Université de Nice - Sophia Antipolis
http://deptinfo.unice.fr/

Question 1

- Ecrivez un programme python qui produit trois processus *zombie* et se met en attente d'une saisie clavier. Une fois que la saisie clavier a eu lieu, le programme doit éliminer les 3 zombies et se terminer a son tour.
- Décodage:
  1. Faire 3 zombie
     - Zombie? Un processus dont la mort est ignorée par son père.



  2. Attendre une saisie clavier
     - Comment attendre? Utiliser input() ou rawinput() ...
  3. Eliminer les zombies?
     - Il suffit de consommer leur code de retour avec wait() ou waitpid()!!

Réponse à la question 1

```
#!/usr/bin/python

import os,sys

for i in range(3):
    pid = os.fork()
    if pid == 0:          # Le i-eme fils...
        sys.exit(i)       # Zombie tant pere ne fait pas wait/waitpid
    else:                 # Le pere
        print "Je viens de creer le fils %d" %(pid)
# Le pere
rep = raw_input("blah...")
for i in range(3):
    pid,status = os.wait()
    print "Mort du fils %d" %(pid)
```


Question 2

- Ecrivez un programme python qui cree un processus fils lorsqu'il recoit le signal SIGUSR1, et affiche le message ``le fils pid=xxxx est mort'' lorsque l'un de ses processus fils meure (où xxxx est le pid du processus qui est mort). Ce programme doit s'exécuter indéfiniment.
- Decodage:
  1. Recevoir SIGUSR1
     - Etre pret à recevoir : installer un traitant
  2. Créer un fils...
     - A chaque reception de USR1, donc DANS le traitant
  3. Afficher le numéro du processus mort
     - Comment savoir qu'un fils est mort?
       - pid,status - os.wait()
       - Fait attendre, ca tombe bien!
  4. S'executer indéfiniment
     - Boucler sur l'attente de la mort d'un fils

Réponse à la question 2

```
#!/usr/bin/python
import os,sys,signal,time
def handler(sig,ignore):
    pid = os.fork()
    if pid == 0:      # Occuper le fils
        time.sleep(2)
        sys.exit()
signal.signal(signal.SIGUSR1,handler)
while True:
    try:
        pid,status = os.wait()
        print "Mort du processus %d" %(pid)
    except:
        pass   # Ignore exceptions
```


Question 3

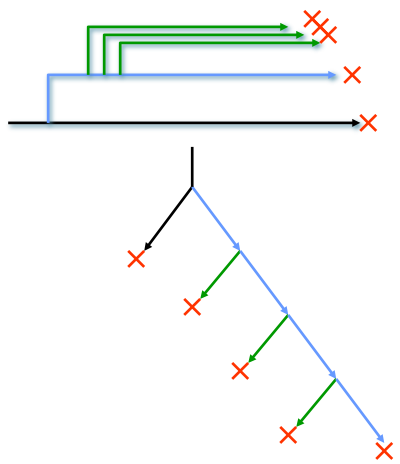
- **Expliquez comment modifier le programme précédent de sorte que le programme s'arrête automatiquement au bout d'une minute en utilisant la fonction alarm et le signal SIGALRM.**
- **Décodage:**
  - **Utiliser alarm() + SIGALRM**
    - **Signal, donc installer un nouveau traitant!**

Réponse à la question 3

```
#!/usr/bin/python
import os,sys,signal,time
def handler(sig,ignore):
    pid = os.fork()
    if pid == 0:      # Occuper le fils
        time.sleep(2)
        sys.exit()
def handler_ALRM(sig,ignore):
    sys.exit(0)
signal.signal(signal.SIGUSR1,handler)
signal.signal(signal.SIGALRM,handler_ALRM)
signal.alarm(60)
while True:
    try:
        pid,status = os.wait()
        print "Mort du processus %d" %(pid)
    except:
        pass   # Ignore exceptions
```

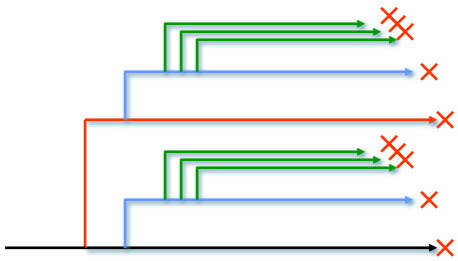

Question 4 : Dessiner Généalogie

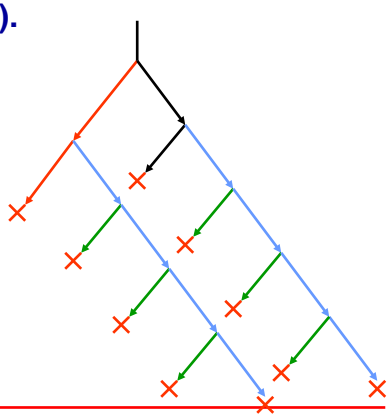
```
import sys,os
# un commentaire (voir 2e question)
pid = os.fork()
if pid != 0:
    os.wait()
    sys.exit(0)
else:
    pid = os.fork()
    if pid == 0:
        time.sleep(10)
        sys.exit(0)
    else:
        pid = os.fork()
        if pid == 0:
            time.sleep(10)
            sys.exit(0)
        else:
            pid = os.fork()
            time.sleep(10)
            sys.exit(0)
```



Question 5

- **Représentez maintenant la généalogie qu'on otiendrait en remplaçant la ligne *# un commentaire (voir 2e question)* par une nouvelle instruction os.fork() (sans toucher aux autres).**





Question 6

- **Ecrivez un programme python qui utilise les sockets pour récupérer le contenu de la page web http://deptinfo.unice.fr/~dalle/index.html et la sauve dans un fichier de nom index.html dans le répertoire courant.**
- **Décodage:**
  1. Récupérer contenu web = socket coté client
  2. Construire message requête
  3. Ouvrir fichier en écriture
  4. Envoyer requête
  5. Lire réponse
  6. Ecrire réponse dans fichier

Réponse question 6

```
import sys,socket
clisock = socket.socket(socket.AF_INET,socket.SOCK_STREAM,0)
addr = ("deptinfo.unice.fr",80)
clissock.connect(addr)
requete = "GET /~dalle/index.html HTTP/1.0\n\n"
clisock.send(requete)
clisock.shutdown(socket.SHUT_WR)   # ecritures
fd = os.open("./index.html",os.O_WRONLY|os.O_CREAT|os.O_TRUNC, 0o644)
msg = "empty"
while len(msg) != 0:
    msg = clisock.recv(1024)
    towrite = len(msg)
    while towrite > 0:
         towrite = towrite - os.write(fd,msg)
         if towrite > 0:
                  msg = msg[-towrite:]
os.close(file)
clisock.close()
```


Petites Questions

- Que se passe-t-il si on utilise la primitive os.lseek() pour déplacer la position courante de lecture/écriture plus loin que la fin du fichier ?
- On veut faire en sorte que les écritures vers la sortie standard d'un programme python soient systématiquement redirigées vers un fichier de nom toto.txt. Expliquez comment faire avec la primitive python os.dup2() (donnez juste les lignes de code python correspondantes, sans faire un programme complet).
- Même question que précédemment, mais cette fois en utilisant la primitive os.dup().