ASSIGNMENT REPORT : FORK, PROCESS AND THREAD IMPLEMENTATION

CENG2034, OPERATING SYSTEMS

Osman Batuhan Şahin
osmanbatuhansahin@posta.mu.edu.tr
github.com/osmanbatuhansahin

Sunday 7th June, 2020

Abstract

In this assignment I code a script with python. My goals were download files from a list of urls and compile them for finding duplicates. I used os, requests, multiprocessing, uuid and hashlib libraries for this tasks. Using multiprocessing methods and child process was necessary in this assignment.

1 Introduction

Purpose of this assignment was understand multiprocessing methods and fork() with using different libraries. I will show you comparison for multiprocessing later of report. I also face with orphan process problem in this assignment. You can see what I do to solve this problem and other problems in assignments part.

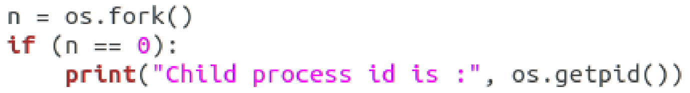
2 Assignments

I tried to solve 4 questions-problems in this assignment.

2.1 Create a new child process with syscall and print its PID.

```
n = os.fork()
if (n == 0):
    print("Child process id is : ", os.getpid())
if (n > 0):
    print("parent : ", os.getpid())
    os.kill(os.getpid(), signal.SIGSTOP)
print("pid is", os.getpid())
```

At my first attempt to print child process, I killed parent process for first and second assignment. Later on I notice, I should use parent for other assignments.

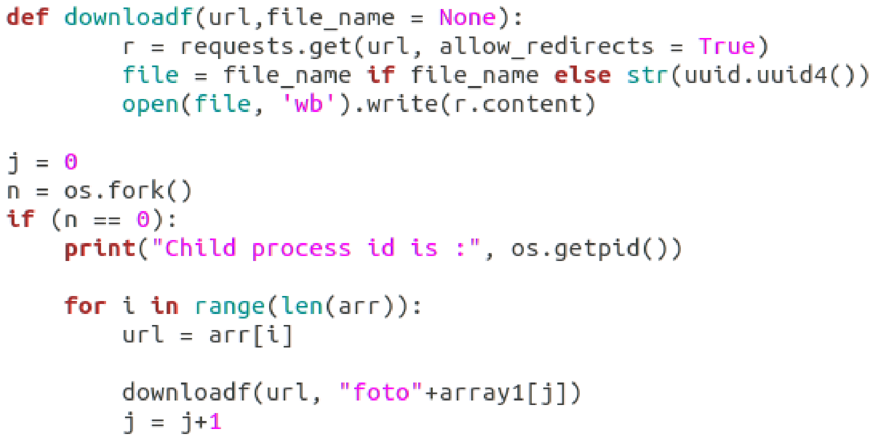


```
n = os.fork()
if (n == 0):
    print("Child process id is :", os.getpid())
```

Then I code this simple if statement and fork() to print child.

2.2 With the child process, download the files via the given URL list.

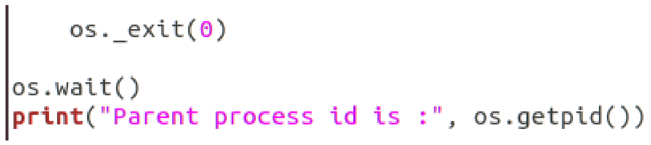
I called download downloadf function in if (n==0), so I downloaded files with child process.



```
def downloadf(url,file_name = None):
        r = requests.get(url, allow_redirects = True)
        file = file_name if file_name else str(uuid.uuid4())
        open(file, 'wb').write(r.content)

j = 0
n = os.fork()
if (n == 0):
    print("Child process id is :", os.getpid())

    for i in range(len(arr)):
        url = arr[i]

        downloadf(url, "foto"+array1[j])
        j = j+1
```

2.3 How can you avoid the orphan process situation with syscall?

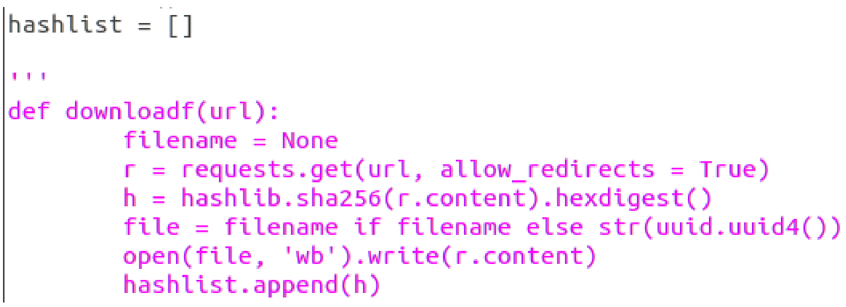
If parent process ends before child process, there is orphan process. My code has an orphan process too. I used os.wait after exit child process and I solved this problem.



```
    os._exit(0)

os.wait()
print("Parent process id is :", os.getpid())
```

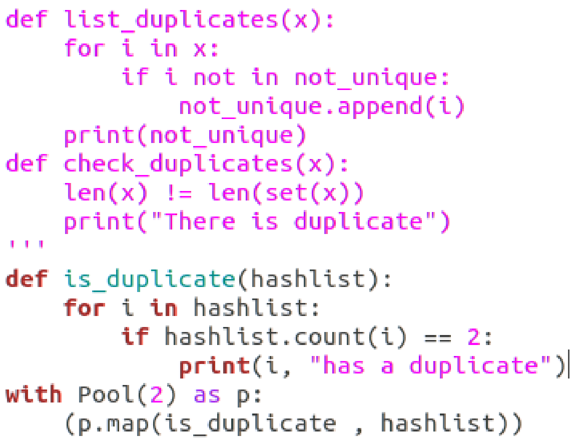
2.4 Control duplicate files within the downloaded files of your python code.

I used hashlib and pool from multiprocessing for this part. I had a lot of problem in this part. I solved some of them but could not solve some.



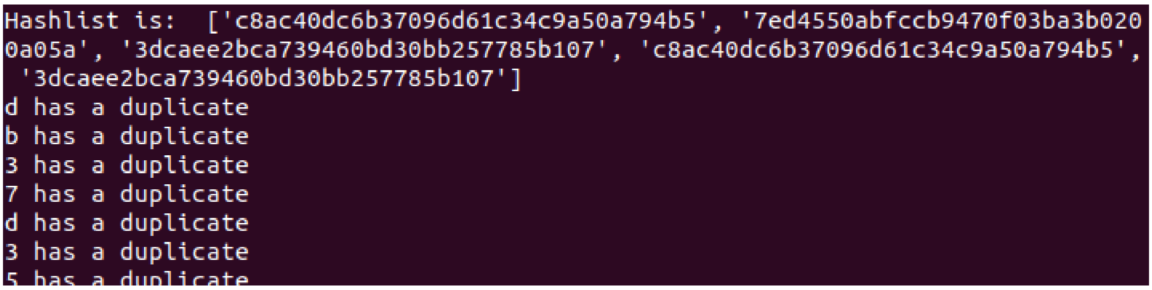
```
hashlist = []

'''
def downloadf(url):
        filename = None
        r = requests.get(url, allow_redirects = True)
        h = hashlib.sha256(r.content).hexdigest()
        file = filename if filename else str(uuid.uuid4())
        open(file, 'wb').write(r.content)
        hashlist.append(h)
```

At first I try to hash files and append them to hashlist inside the downloadf function. Then I notice I was calling this function inside child process. No way to multiprocessing like this. My solution to this problem is hash and append files one by one. I could do it because I determined file names.



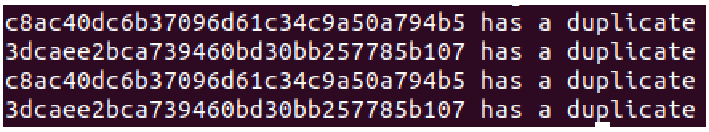
```
def list_duplicates(x):
    for i in x:
        if i not in not_unique:
            not_unique.append(i)
    print(not_unique)
def check_duplicates(x):
    len(x) != len(set(x))
    print("There is duplicate")
'''
def is_duplicate(hashlist):
    for i in hashlist:
        if hashlist.count(i) == 2:
            print(i, "has a duplicate")
with Pool(2) as p:
    (p.map(is_duplicate , hashlist))
```

Then I have to find duplicates in hashlist. I code a lot of function to list duplicates. But everytime I had same problem. When my hashlist enters to function, strings in that list becomes 1 character. Here is three of my function trials.



```
Hashlist is:  ['c8ac40dc6b37096d61c34c9a50a794b5', '7ed4550abfccb9470f03ba3b020
0a05a', '3dcaee2bca739460bd30bb257785b107', 'c8ac40dc6b37096d61c34c9a50a794b5',
 '3dcaee2bca739460bd30bb257785b107']
d has a duplicate
b has a duplicate
3 has a duplicate
7 has a duplicate
d has a duplicate
3 has a duplicate
5 has a duplicate
```

My output was like this when I call functions with pool. I code different functions but I could not have the output what I want. As you can see Hashlist is normal.

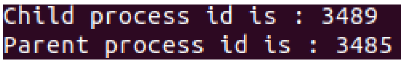


```
c8ac40dc6b37096d61c34c9a50a794b5 has a duplicate
3dcaee2bca739460bd30bb257785b107 has a duplicate
c8ac40dc6b37096d61c34c9a50a794b5 has a duplicate
3dcaee2bca739460bd30bb257785b107 has a duplicate
```

When I call function without multiprocessing, my output is like this.

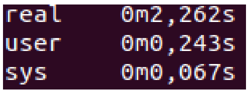
3 Results

In this assignment my first duty was creating a child process and prints its process id.



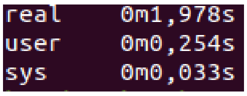
```
Child process id is : 3489
Parent process id is : 3485
```

Parent and child process have different process id. They can work separate or together. If parent process ends before child process, there is orphan process. I learned how to avoid this situation with os.wait() command. I also learned using hashlib. This library allows us to encrypt our datas. I hashed files and list them in hashlist.
I checked runtimes multiprocessing method and normal function calling way to compare them and see time difference between them. But there is no big difference. I expected multiprocessing method is faster but it was not.



```
real	0m2,262s
user	0m0,243s
sys	0m0,067s
```

calling function with multiprocessing



```
real    0m1,978s
user    0m0,254s
sys     0m0,033s
```

calling function without multiprocessing

4 Conclusion

In conclusion, first, I learned creating a child process and work with it. Child process can work solo or with parent process. But if child process works with parent process, there can be orphan process situation. We can avoid this with use os.wait command to parent process.
I also learned download files from an url list and encrypt them with using hashlib library. Duplicate files had same string value with this method. I could compare them. But I could not take my output good because of the reasons I mentioned about at assignment part of my report. I ran my isduplicate function with multiprocessing method pool and with normal way and I compare their runtimes. There was not a big difference. Maybe if I had a bigger script and better system, I could see advantages of multiprocessing.