

# Laborator 4

## Utilizarea limbajului Python în administrarea sistemului de operare

### 1 Scopul lucrării

În laboratorul curent se vor prezenta biblioteci Python uzuale pentru lucrul cu sistemul de operare.

### 2 Fundamente teoretice

#### 2.1 Biblioteca os

Biblioteca `os` conține funcții care permit interacțiunea cu elemente ale sistemului de operare. Câteva funcții utile din această bibliotecă sunt:

- `os.getenv()` primește numele unei variabile de mediu și returnează valoarea acesteia.
- `os.listdir()` returnează o listă cu elementele din folderul dat ca parametru.
- `os.path.isfile()` și `os.path.isdir()` verifică dacă elementul primit ca parametru e un fișier, respectiv director.
- `os.path.join()` unește două sau mai multe elemente din sistemul de fișiere pentru a construi o cale, folosind separatorul de pe sistemul de operare curent (`'/'` pe Linux și `'\'` pe Windows).

Funcția de mai jos listează toate subdirectoarele din directorul primit ca parametru.

```
1 def listSubdirs(dirPath):
2     for fname in os.listdir(dirPath):
3         fpath = os.path.join(dirPath, fname)
4         if os.path.isdir(fpath):
5             print(fpath)
```

Lista completă a funcțiilor împreună cu detaliile acestora se găsește la <https://docs.python.org/3/library/os.html>.

#### 2.2 Biblioteca shutil

Biblioteca `shutil` oferă funcții de nivel mai înalt asupra sistemului de fișiere. Câteva funcții utile sunt:

- `shutil.copy()` - copiază din fișierul sursă în cel destinație. Dacă destinația este un folder se va crea în el un fișier cu același nume cu sursa.
- `shutil.copytree()` copiază un folder de la sursă la destinație în mod recursiv.
- `shutil.rmtree()` șterge în mod recursiv un folder.

Lista completă a funcțiilor împreună cu detaliile acestora se găsește la <https://docs.python.org/3/library/shutil.html>.

## 2.3 Biblioteca subprocess

- `subprocess.Popen()` creează un nou proces folosind lista de argumente primită ca parametru și returnează un obiect de tip proces.
- `subprocess.call()` și `subprocess.run()` rulează procesul primit la fel ca `Popen` și așteaptă terminarea acestuia.
- Metoda `wait()` așteaptă terminarea unui proces.
- Metoda `communicate()` poate transmite date pentru intrarea standard a procesului (*stdin*) și citi data de la ieșirea standard (*stdout*) și de eroare (*stderr*).

Lista completă a funcțiilor împreună cu detaliile acestora se găsește la <https://docs.python.org/3/library/subprocess.html>.

## 3 Mersul lucrării

### 3.1 Probleme propuse

1. Afișați numele fișierului cel mai mare (ca număr de octeți din conținut) dintr-un director dat ca parametru. Pentru punctaj maxim trebuie să intrați recursiv în subdirectoare.
2. Apelați programul `/bin/ls` pentru a obține conținutul unui folder și comparați-o cu rezultatul funcției `os.listdir()`. Țineți cont de faptul că elementele pot apărea în ordine diferită, deci o idee bună ar fi să sortați rezultatele înainte de a le compara.
3. Scrieți un script care realizează o copie a unui folder primit ca parametru, apoi șterge din copie toate directoarele goale. Pentru a nu șterge ce nu trebuie, cereți utilizatorului să confirme ștergerea fiecărui director, afișând calea completă a fiecărui director ce urmează a fi șters și așteptând ca utilizatorul să tasteze **yes** sau **no**.
4. Scrieți o funcție Python echivalentă cu comanda `which`, care primește un nume de program și îl caută în folderele din variabila de mediu `$PATH`, returnând prima cale în care a fost găsit. De exemplu dacă apelăm `which("ls")` ar trebui să returneze `"/bin/ls"`.