

Laborator 3

Elemente de bază ale limbajului Python

1 Scopul lucrării

În laboratorul curent se vor exersa elementele de sintaxă ale limbajului Python 3 împreună cu operațiile pe fișiere.

2 Fundamente teoretice

2.1 Scrierea și rularea unui script Python

Un script Python este un fișier text cu extensia `.py`. Acesta poate fi editat în orice editor, cum ar fi cele în mod text (`nano`, `vim`, `mcedit`) sau vizual (`gedit`, `Sublime Text`, `VS Code`).

Structura minimală a unui script Python ar trebui să fie următoarea:

```
1 |#!/usr/bin/env python3
2 |
3 |def main():
4 |    pass
5 |
6 |if __name__ == "__main__":
7 |    main()
```

Prima linie din exemplu se numește *shebang* și indică ce interpretor trebuie să se folosească pentru restul script-ului. În cazul nostru dorim să se folosească interpretorul de Python 3, deoarece interpretorul implicit de Python pe mai multe distribuții de Linux este încă Python 2.

Condiția de la linia 6 verifică dacă script-ul nostru a fost executat direct, sau a fost încărcat ca bibliotecă de un alt script. În general, orice script Python poate fi încărcat de un alt script ca bibliotecă folosind directiva `import`. Atunci când un script este încărcat, se execută tot codul neindentat. Am dori să se apeleze funcția `main()` doar dacă script-ul este rulat direct, altfel doar să expunem funcțiile și variabilele globale existente.

Ca să putem executa un script, ar trebui să îi dăm întâi drepturi de execuție. Dacă avem un script care se numește `myscript.py`, putem folosi comanda de mai jos.

_____ Acordarea dreptului de execuție _____
`chmod +x myscript.py`

Odată ce un script e executabil, putem să îl rulăm din folderul curent prefixându-i numele cu `./`, precum mai jos.

_____ Rularea unui script _____
`./myscript.py`

O variantă alternativă de a rula un script Python, chiar dacă acesta nu e executabil, este să invocăm direct interpretorul:

_____ Variantă alternativă de rulare a unui script _____
`python3 myscript.py`

3 Mersul lucrării

3.1 Probleme propuse

1. Scrieți un script Python care primește ca argumente în linia de comandă două numere întregi și afișează pe ecran cel mai mare divizor comun al lor. Extindeți script-ul astfel încât să accepte oricâte argumente și să calculeze cel mai mare divizor comun pentru toate.
2. Se dă o listă Python ce conține mai multe numere întregi. Folosind eventual un dicționar, numărați de câte ori apare fiecare element din listă și afișați pe ecran elementele unice împreună cu numărul de apariții.
3. În fișierul `/etc/passwd` se găsesc informații despre utilizatorii din sistem. Fiecare linie conține mai multe câmpuri, separate prin `:`.
 - afișați numărul de utilizatori din sistem
 - știind că al treilea câmp de pe fiecare linie reprezintă ID-ul utilizatorului, afișați utilizatorul cu cel mai mare ID (username-ul e primul câmp de pe fiecare linie)
 - al 7-lea câmp (ultimul) de pe fiecare linie reprezintă shell-ul utilizatorului. Un utilizator care are `/usr/sbin/nologin` sau `/bin/false` ca shell nu se poate loga. Afișați lista utilizatorilor care se pot loga.