


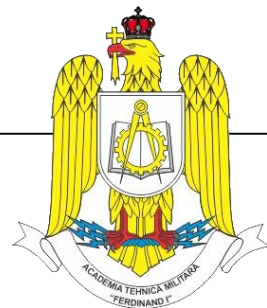


Laborator 01

Setup infrastructură

- Instalați Windows Subsystems for Linux.
 - Control Panel >> **Windows Features** >> Selectați **Windows Subsystems for Linux** >> **OK**
- Instalați Ubuntu 20.04.
 - Microsoft Store >> Search Ubuntu >> Ubuntu 20.04 >> Install >> Launch
- Copiați [cheia](#) într-un fișier.
- Instalați [Putty](#) .
 - Host Name: [username@wiki.mta.ro](#)
 - Connections >> SSH >> Auth >> Browse... pentru a pune cheia.
- Instalați [WinSCP](#) .
 - New Site
 - Host name: wiki.mta.ro
 - Port number: 22
 - User name: cel de pe wiki.mta.ro, **fără** @wiki.mta.ro
 - Advanced...
 - SSH >> Authentication >> Private Key File [...] >> OK
 - Save >> Login
 - Stânga mergeți în directorul dorit - Dreapta folder-ul de pe server
 - Mergeți în folder-ul labs pe server.
 - Stânga sus apăsați  **Synchronize** pentru a copia fișierele de pe server.
 - Stânga sus apăsați  urmat de **Start** pentru a muta continuu fișierele din folder-ul local pe server.
- Instalați compilator și make pe Linux.
 - sudo apt-get install gcc
 - sudo apt-get install make
 - sudo apt-get install gdb
- Instalați [Visual Studio Code](#) .
- Instalați extensii Visual Studio Code:
 - C/C++ (IntelliSense) – autor Microsoft (**trebuie instalat în WSL**)
 - Remote-WSL – autor Microsoft
- Setati Visual Studio Code să folosească WSL (Windows Subsystems for Linux).
 - Stânga jos buton verde două săgeți 
 - Remote-WSL: New Window
 - Dacă aveți mai multe distribuții instalate e bine să apăsați Remote-WSL: New Window using Distro... și apoi să o selectați pe cea cu Ubuntu 20.04
 - Open folder...
 - Scrieți /mnt/ în loc de /root . Selectați partiția și acum sunteți prezentat cu lista de directoare Windows. Folosiți directorul în care doriți să lucrați.
 - **Trebuie să apară în Visual Studio subfolderul .vscode**



Exerciții

1. Compilați și rulați codul din **helloWorld.c** .
2. Aflați numărul de core-uri ale procesorului folosit, din linia de comandă și din codul C.
 - Căutați pe Google cum se afișează numărul de core-uri din CLI pe Linux
 - Căutați pe Google “sysconf() number of cores”
3. Modificați codul astfel încât acesta să ruleze cu 2 de thread-uri.
4. Modificați codul astfel încât acesta să ruleze cu 100 de thread-uri.
 - De ce funcționează un program cu mai multe thread-uri decât core-uri?
5. Modificați codul astfel încât acesta să ruleze pe un număr de thread-uri egal cu numărul de core-uri.
 - Asigurați-vă că numărul de thread-uri e setat automat.
 - Dacă programul rulează pe un calculator cu număr diferit de core-uri, le va folosi pe toate?
 - Rulați codul pe serverul wiki.mta.ro. Câte afișări apar?
6. Modificați funcția numită **threadFunction**.
 - Mesajul “Hello World” se va fi afișat de 100 de ori.
 - Adăugați numărul iterației în afișare.
 - Programul va rula cu 2 thread-uri.
 - Cum arată afișarea? Explicați.
7. Modificați codul din **twoThreadsTwoFunctions.c** .
 - Programul va porni două thread-uri.
 - Unul din thread-uri va folosi funcția existentă **threadFunction**.
 - Al doilea thread va folosi o nouă funcție numită **threadFunction2**.
 - threadFunction2 va afișa mesajul “Salutare Planetă!”.
 - Ați închis corect thread-urile?
 - Mai este nevoie de toate for-urile?

Exercițiile de la 1 la 7 sunt obligatorii. Conceptele explorate sunt esențiale pentru obținerea notei **minime** de promovare.

Vă recomandăm, pentru a crește șansele de a obține o notă cât mai mare să explorați și următoarele exerciții:

8. Scrieți cod care să identifice care este numărul maxim de thread-uri pe care îl puteți porni.