**Laborator 02**

**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
* Asigurați-vă că ați primit cheia (de la asistent pe Teams) și salvați-o.
* Instalați [Putty](https://www.putty.org/) .
  + Host Name: username@20.52.209.189
    - username este cel de pe wiki.mta.ro
  + Connections >> SSH >> Auth >> Browse… pentru a pune cheia.
* Instalați [WinSCP](https://winscp.net/eng/index.php).
  + Când instalați de la *User Interface Style* selectați *Commander*.
  + New Site
    - Host name: 20.52.209.189
    - 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.
  + La începutul laboratorului copiați de pe server pe local.
    - **Atenție** copiați din folderul labs (folderul 01 conține un folder .vscode care este invizibil).
  + **La sfârșitul laboratorului copiați de pe local pe server.**
* Instalați compilator și make pe Linux.
  + **sudo apt-get update**
  + **sudo apt-get install gcc**
  + **sudo apt-get install make**
  + **sudo apt-get install gdb**
* Instalați [Visual Studio Code](https://code.visualstudio.com/) .
* Instalați extensii Visual Studio Code:
  + C/C++ (IntelliSense) – autor Microsoft **(trebuie instalat în WSL)**
  + Remote-WSL – autor Microsoft
* Setați 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**

**Pentru fiecare exercițiu se va scrie în fișierul \_REPORT.txt rezultatul rulărilor și răspunsul la întrebări.**

1. (**stackSize.c**) Doar pentru sisteme cu LINUX, pe Windows mărimea stivei este de 1MB. Scrieți cod în C care să afișeze dimensiunea stivei (Căutați pe Google apel system() și ulimit -a)
2. (**bigVector.c**) Declarați un vector de caractere numit **a,** de mărime **N,** cu **N** fiind un număr definit cu #define.
   * Declararea lui **a** se va face în interiorul funcției **main()**.
   * Umpleți acest vector cu 0-uri și afișați ultimul element.
   * Testați programul anterior cu **N** setat la 100 și cu **N** setat la o valoare mai mare decât mărimea stack-ului.
   * Declarați un pointer la un șir de caractere **b** și alocați pentru acesta 1GB de memorie.
   * Repetați alocarea precedentă de un milion de ori. Adăugați verificare, când pointerul este **NULL** afișați numărul iterației la care s-a ajuns.
   * Cât spațiu RAM ați putut aloca?
3. (**sizeof.c**) Analizați diferența dintre diferitele metode de a aloca vectori:
   * Alocați un vector de 1000 de caractere, inițializat cu șirul “Hello World!”.
   * Alocați un pointer la un șir de caractere inițializat cu șirul “Hello World!”.
   * Alocați un pointer de caractere neinițializat pentru care alocați dinamic 1000 de caractere.
   * Afișați pentru toate 3 mărimea datelor (sizeof) și mărimea șirurilor de caractere.
   * Afișați cei 3 pointeri și adresele lor.
   * Scrieți în **\_REPORT.txt** ce observați din afișări.
4. (**complex.c**) Scrieți un program pentru lucru cu numere complexe.
   * Se definește o structură ce reprezintă un număr complex (două float-uri).
   * Se definesc ca funcții operațiile de bază: adunare, scădere, înmulțire.
   * Se implementează cod de evidențiere a funcționalității celor 3.
5. (**allStruct.c**) Definiți o structură ce conține **toate** tipurile de date pe care le cunoașteți.

**Exercițiile de la 1 la 5** 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:**

1. (**3dMatrix.c**) Declarați o matrice tridimensională, inițializați toate valorile sale cu 1 și implementați funcția care calculează suma tuturor valorilor sale.