A Visual Studio Code használata C programok fejlesztéséhez

A Visual Studio Code (a segédletben a továbbiakban VSC) egy kódszerkesztő keretrendszer, amely kiterjesztésekkel számos fejlesztő környezet kialakítását teszi lehetővé. A VSC már önmagában több száz beépített funkcióval rendelkezik, amit a kiterjesztések még bővítenek. A jelen segédlet ezek közül azt a néhányat ismerteti, ami C nyelven írt gyakorló példák készítéséhez, futtatásához szükséges.

1. Előfeltételek

A segédlet feltételezi, hogy a VisualStudioCode_install segédletben leírtak alapján történt a rendszer és a kiegészítések telepítése.

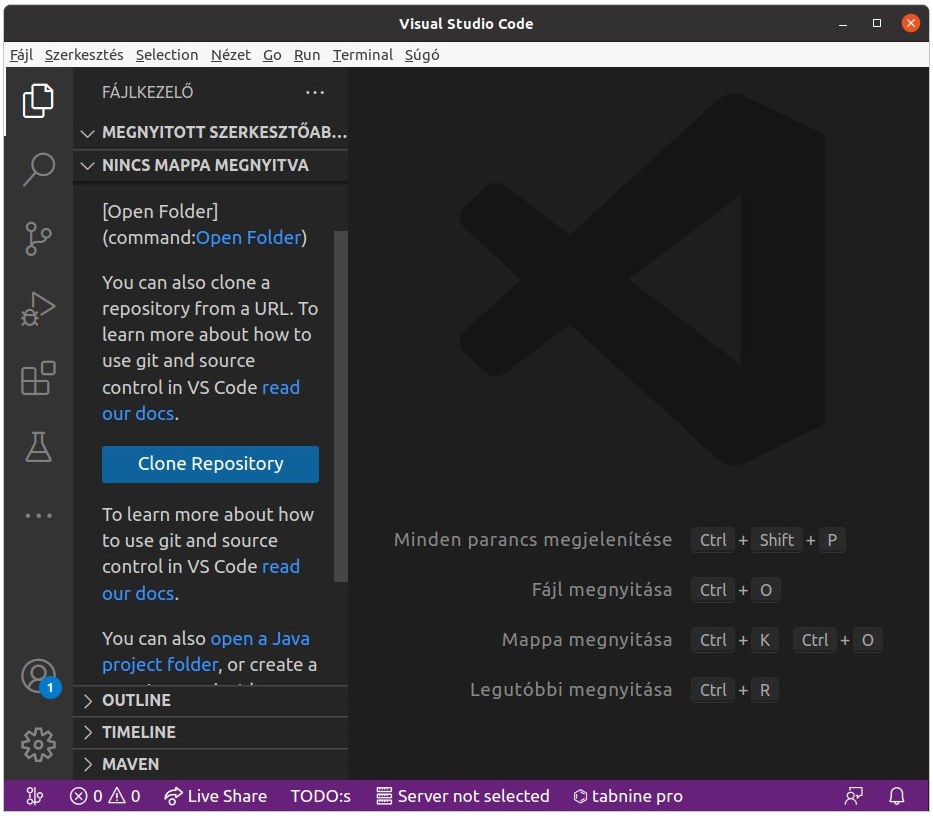
2. C nyelvű programok készítése

2.1 Forráskód szerkesztése

Hozzunk létre egy mappát, amiben majd a gyakorló feladatokat tárolni szeretnénk.

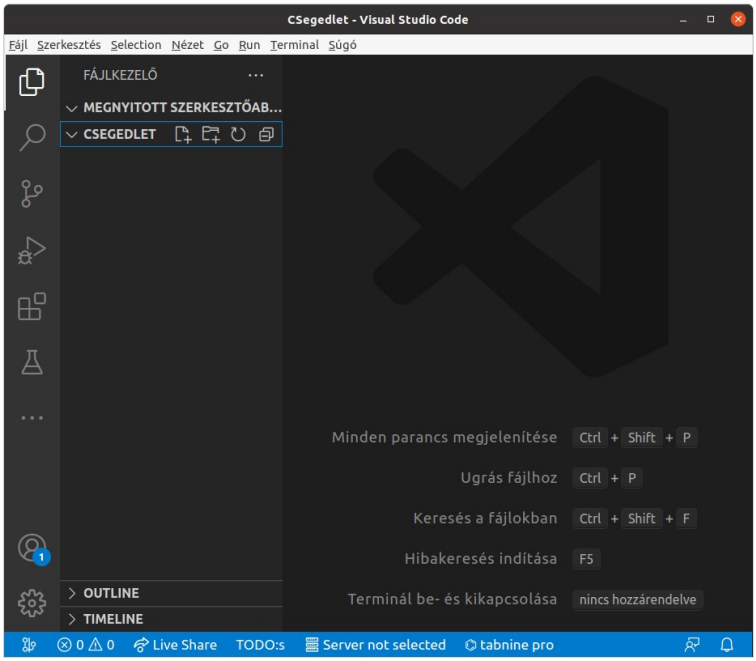
Nyissuk meg a VSC alkalmazást. Alapesetben a címsor alatt egy főmenüt, a jobb szélen egy eszköztárat látunk.

Az eszköztár első elemére (Filekezelő – piros jelzés) kattintva az alábbi lehetőségeket kapjuk:



Ebből most válasszuk a zölddel jelzett „Open Folder” funkciót. Ekkor egy szokásos fájl kezelő ablakot kapunk, amiben kijelölhetjük és megnyithatjuk az előzőleg létrehozott mappát. (Példánkban a CSegedlet nevűt). A mappa kiválasztása után nyugtáznunk kell, hogy ez megbízható forrás („Yes, I trust the authors”).

A képernyőkép ezután:



A mappanév utáni első ikon a „New File” erre kattintva és egy „.c” kiterjesztésű fájlt létrehozva elkezdhetjük szerkeszteni a jobb oldalon megjelenő szerkesztő ablakban a forrásfile-t.

2.1.1 Kódkiegészítés

A VSC a .c kiterjesztés ismeretében feltételezi, hogy a szerkesztendő szövegfile C forráskód, és ennek megfelelően alkalmazza a kódkiegészítés funkciót, azaz miközben gépelünk, az éppen begépelt szöveget megpróbálja értelem szerűen kiegészíteni, amivel gépelési munkát tudunk megtakarítani, és csökkenthetjük az elgépelésből származó hibákat. Ennek a működését majd a gyakorlatban tudjuk kitapasztalni.

2.2.2 Kódszínezés

Miután a VSC tudja, hogy C programkódot írunk, a gépelés során folyamatosan értelmezi a szöveget, és a különböző nyelvi szerkezeteket más-más színnel jeleníti meg.

2.3. Program fordítás, futtatás

A „Run” főmenü „Start debugging” (F5) és „Run Without Debugging (CTRL+F5) funkciók alapesetben azonos módon működnek: az aktuális ablakban található forrásfile **legutoljára lementett** verzióját lefordítják és lefuttatják.

Figyelem! Nem feltétlenül az a programszöveg fut, ami a szerkesztő ablakban látható, hanem amit utoljára mentettünk. Hogy ez ne okozzon meglepetést, a File menüben kattintsuk be az „Auto Save” funkciót, mert így a futtatás előtt automatikusan mentődik a szerkesztő ablak tartalma.

Ha a fordítás során hibajelzést kapunk, az a szerkesztő ablak alatt kijelölhető „Problémák” ablakban látható.

A program standard outputra írt kimenete a „Terminál” ablakban látható.

2.4 Program debuggolása

Ha a program nem az elvárt eredményt adja, a hiba megkereséséhez lehetőség van a program futásának megállítására, utasításonként való végrehajtására, illetve bizonyos változók pillanatnyi értékének kiíratására. Ezt a folyamatot szokás „debuggolás”-nak nevezni.

Alapesetben a debug üzemmódban is végigfut a program. Ahhoz, hogy lépésenként futtathassuk, töréspontot (breakpoint) kell elhelyezni a forrásfile-ban. Ezt a sorszám elé kattintva tehetjük meg. Ekkor ez a sor egy piros ponttal lesz megjelölve, és debug módban indítva a programot, a futása ennek a sornak a végrehajtása előtt megáll. Ezután az alábbiakat választhatjuk?

- Step Over (F10) – A következő sor végrehajtása
- Step Into (F11) – Ha a következő sor függvényhívás, a függvény első sorára ugrás
- Step Out (Shift+F11) – A függvény további sorainak végrehajtása és kilépés a függvényből
- Continue (F5) – A program további utasításainak végrehajtása most már megállás nélkül