2. LABOR

A C++, MINT EGY JOBB C NYELV

Általános információk

A lenti feladatok megoldásával nem szerezhető iMSc pont, ellenben a mellékelt „02_LAB_IMSC.zip” fájlban foglaltakkal (további információkat pontszerzéssel kapcsolatban abban találsz).

Kötelező feladatok

1. Konstans típus gyakorlása

Tekintsétek közösen a mellékelt *Constant* solutiont.

2. C Stack

A mellékelt *Stack* solutiont - a kommentekbe foglalt pontoknak megfelelően – hozd működőképes állapotba.

A Stack egy veremmemória jellegű tároló: az utoljára (vagy később) a paraméterként átadott struktúrába behelyezett (stack_push) elemet tudjuk levenni (stack_pop), ellentétben mondjuk egy FIFO tárolóval (First in, first out). Nem tudjuk, mennyi elemet fognak benne tárolni (amíg el nem fogy a memória, addig pakolhatunk bele…) A stack_pop függvény a legutoljára betett elemet nemcsak, hogy leveszi a paraméterként átvett stack tetjéről (törli az adatszerkezetből), de vissza is tér vele, további felhasználásra. Idézd fel a tavaly tanultakat a dinamikus memória felhasználásról.

3. C++ Stack: pointerek helyett referencia

Írd át úgy az előző feladat forráskódját, hogy pointerek helyett referenciát használjon paraméter átadásra.

- Mivel referencia nem létezik C-ben, át kell állni C++ fordítóra Visual Studio-ban.
- Ezt úgy teheted meg, hogy módosítod a stackMain.c kiterjesztését *.cpp-ra.
- VSCode esetén a tasks.json állományban is átírod a *.c kiterjesztések fordítását *.cpp-re
- Ezután már elkezdheted átírni a forráskódot.

4. C++ Stack: függvénynév túlterhelés

Miután már átálltál C++ fordítóra, képes vagy függvénynevet túlterhelni. Szedd ki a kommentet a második stack_init(…) függvény körül, és implementáld az ott definiált kommentek segítségével.

- Legyen képes egy már létező stack alapján (other) inicializálni egy új stacket (s).
- Ügyelj arra, hogy ne mutasson az other elemeire, hanem másolja is le őket.
    - Dinamikusan foglalj új memóriaterületet.
    - Elemenként másold át az other tartalmát az s stackbe.
- Ne a stack_push(…)-t használd elem átmásolásra, találj hatékonyabb megoldást!
- A main(…)-ben írj tesztelő kódrészletet a most megírt függvényhez!

5. Main argumentum, inline függvény

Készíts egy új solution-t *RapidSquare* néven.

- Írj egy inline függvényt, amely paraméterként int-et vár, int-et ad vissza
  - térjen vissza a paraméter négyzetével
- A program argumentumként várjon egész számokat szóközzel elválasztva
- Menjen végig az argumentumként megadott számokon
- Írd ki őket egymás mellé szóközzel elválasztva úgy, hogy alkalmazod rájuk az első pontban megírt függvényt

Gyakorló feladatok

- Konstansok használata