6. LABOR

NÉVTEREK, KONSTANS ÉS STATIKUS DEKLARÁTOROK

Általános információk

1 iMSc pont jár a 4. és 5. feladat együttes teljesítéséért. Az 5. feladatnál elég egy rövid kódrészlet, amiben bemutatod, hogy tényleg csak egyszer lehet példányosítani a Logger osztályt.

Kötelező feladatok

1. Névütközés elkerülése

Oldd meg, hogy a *NameConflict* solution *nameConflict.cpp*-je lefusson az alábbiakat figyelembe véve:

- Nem módosíthatsz a main()-en
- Nem rakhatsz bele plusz kiíratást
- Ne legyen névütközés
- A függvényeket nem nevezheted át
- A paraméterlistájukat nem változtathatod meg
- Minden függvényt meg kell hívnod legalább egyszer és az eredményüket fel is kell használnod
- A program írja ki a standard kimenetre a 9-es számjegyet

2. Computer osztály

Egészítsd ki a megadott *Computer* solution állományait a következőknek megfelelően:

- A *computer.h*-ban pótold a hiányzó *static*, *friend* és *const* kulcsszavakat!
- A *computer.h*-nak megfelelően írd meg a *computer.cpp* fájlt.
- Ügyelj arra, hogy a mellékelt *computerTest.cpp* helyesen lefusson.

3. String osztály

Egészítsd ki a megadott *String* solution állományait a következőknek megfelelően:

- A megkapott *string.h* állományt egészítsd ki a hiányzó függvénydeklarációkkal!
- Pótold a hiányzó *const* kulcsszavakat ott, ahol annak jelentősége van!
- Add át a paramétereket konstans referenciával, ahol ez megéri
  - a beépített típusoknál nem, ott érdemesebb másolni
- Írd meg a függvényeket a *string.cpp* állományban! Figyelj a névterekre!
- Futtasd le debuggerben a *main(…)* függvényt!
  - Hol hívódik destruktor és másoló konstruktor? Miért?
- Egy statikus függvényt át lehet alakítani nem statikussá is?

4. Logger singleton osztály

Egy általad tervezett beágyazott rendszerre írtál egy programot C++-ban és szeretnél globális eseménynaplózást megvalósítani (hálózati kapcsolat megszűnt/helyreállt, abnormálisan magas hőmérséklet a processzorban (junction temperature). Úgy döntöttél, hogy a jól bevált Singleton tervezési mintát alkalmazod a probléma megoldására.

Készíts egy Logger singleton osztályt, ami legyen képes az alábbiakra:

- A megadott *loggingTest.cpp* állománynak megfelelően írt meg a *logging.h* és a *logging.cpp* fájlokat.
- Ügyelj a helyes névtér használatra.
- Használj enumerációt ott, ahol kell.
- Biztosítsd a következő naplózási szinteket:
  - DEBUG: minden eseményt naplóz
  - INFO: sikeres/sikertelen végrehajtást naplóz
  - WARN: a jövőre nézve veszélyes eseményeket naplóz
  - ERROR: hibaeseményt naplóz
- Legyen beállítható a minimális naplózási szint
  - csak egy bizonyos szint és az afeletti szintek eseményeit naplózza
  - használd ezt a sorrendet: DEBUG, INFO, WARN, ERROR

5. Biztos, hogy singleton?

Győződj meg róla, hogy akárhányszor hívod meg a Logger::getInstance() függvényt, mindig ugyanazt a példányt kapod vissza.