

11. LABOR

GENERIKUS ADATSZERKEZETEK

Általános információk

1 iMSc pont szerezhető az összes feladat hibátlan teljesítéséért.

Kötelező feladatok

1. Vector osztály

Alakítsd át a megadott, *int* típusú adatokat tartalmazó *Vector* osztályt a tárolandó típussal parametrizált osztályra!

Mik a tárolt típussal szemben támasztott követelmények?

Megjegyzés: a generikus osztályokat parametrizált osztályoknak is szokás hívni.

2. Véges automata

Ismertető

Ez a feladat egyrészt ízelítőként szolgál a majdani Informatika 2 tárgyhoz miközben gyakorolhatod a templatek használatát. Ebben a feladatban egy *véges automatát* (*finite automaton*) kell megvalósítani, ami nagyon **hasonlít a Digitális technika 1-ből megismert állapotgépekhez**, csak most a bemenet bármilyen típus lehet, valamint az állapotátmeneti élekhez nem rendelünk semmilyen cselekvést az állapotváltáson kívül.

Az ilyen automaták egyetlen feladata a tartalmazás kérdésének eldöntése. Ily módon hatékonyan(=gyorsan) lehet **mintákat felismerni** akár **szövegekben**, **DNS láncokban**, vagy most éppen **kutyák egy bizonyos részszekvenciáját**, de használják szintaktikai elemzőként is.

Néhány egyszerű fogalom a teljesség igénye nélkül, nagyon vázlatosan

Az **automata** kívülről nézve nagyon hasonló egy függvényre, ami ha kap egy bemenetet, megmondja, hogy elfogadja-e vagy sem. A különbség, hogy most a **kimenet értéke függ attól, hogy a kapott bemeneti elemek hatására milyen állapotba került végül az automata**. Ezt az automatát most a *FiniteAutomaton* osztály valósítja meg. Az automata állapotokat (*State*) valamint szabályokat (*Rule*) tárol, amik az állapotokon értelmezett állapotátmeneteknek feleltethetők meg. Azért hívjuk az automatát végesnek, mert véges sok állapota van.

Érdekesség, hogy a véges automata egy olyan speciális esete a *Turing-gépnek* (Alan Turing után), amikor a gépnek egyetlen, csak olvasható szalagja van, amin előre felé halad.



1. ábra Turing-gép

Teendők

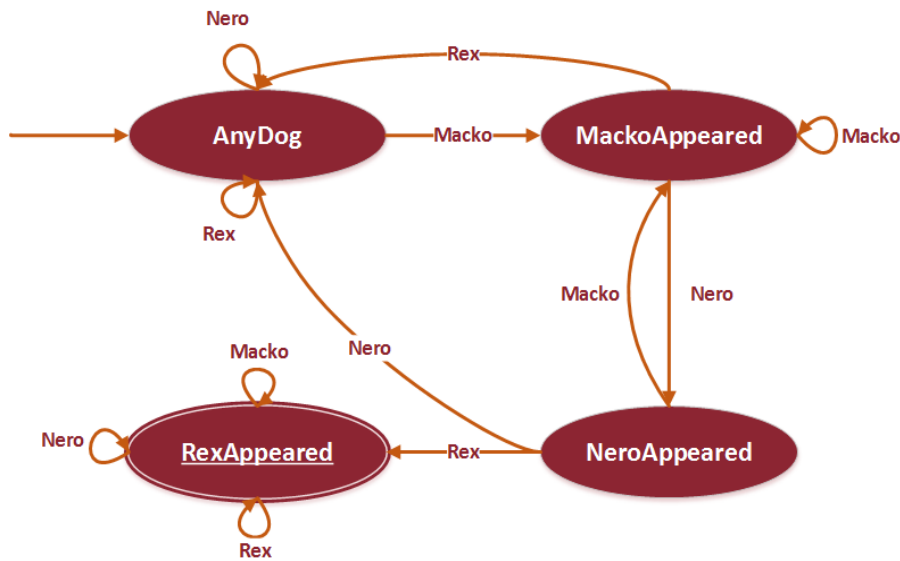
Az előző feladat *Vector* projektjéhez **add hozzá a következő, már létező fájlokat** (jobb klikk az adott mappára → Add → Existing Item...):

- **Header Files:** *finiteAutomaton.hpp*, *rule.hpp*, *state.hpp*
- **Source Files:** *finiteAutomatonTest.cpp*

Ezekben a fájlokban a számozott és TODO kommentek alapján **írd meg a hiányzó részeket**.

Ezután a **test.cpp**-ből a **kikommentezett rész elöl töröld a komment jeleket**, hogy tudd tesztelni a véges automatát.

Az tesztautomata értelmezése



2. ábra A *finiteAutomatonTest.cpp*-ben definiált determinisztikus véges automata

Az automataba a *Mackó*, *Néró* és *Rex* nevű kutyák véges ismétléses variációja érkezik.

Vegyük például a következő bemenetet: *Néró*, *Mackó*, *Rex*, *Rex*, *Néró*, *Néró*, ***Mackó***, ***Néró***, ***Rex***, *Rex*, *Mackó*

Az automata a kivastagított részt fogja felismerni. Magyarán tartalmaznia kell a ***Mackó***, ***Néró***, ***Rex*** részszekvenciát. Mindaddig az ***AnyDog*** kezdőállapotban marad, amíg nem érkezik *Mackó*, mert ekkor a *MackoAppeared* állapotba ugrik át. A többi átmenet ugyanígy leolvasható az ábráról. A bekarikázott ***RexAppeared*** állapot az automata elfogadó állapota. Ez azt jelenti, hogy ha a bemeneti szekvencia hatására történt állapotokon való ugrálás **végén** *RexAppeared* állapotban vagyunk, elfogadjuk a szekvenciát, különben pedig nem.

Gyakorlófeladatok

1. [Generikus verem osztály készítése](#)
2. [Generikus halmaz osztály készítése](#)
3. [Generikus sztring osztály készítése](#)
4. [Generikus mátrix osztály](#)
5. [Generikus asszociatív tár \(a jobbaknak\)](#)
6. [Generikus hash-alapú asszociatív tár \(a nagyon jóknak\)](#)