Adatszerkezetek és algoritmusok 1. kis házi feladat

A feladatok során a bemenetek nem léphetnek túl a limiteken, ezt nem kell külön ellenőrizni.

Mátrix műveletek

Írj egy olyan osztályt, amely példányai egy-egy egész számokat tartalmazó mátrixot reprezentálnak. A mátrix méretét és az elemeket előre nem ismerjük. A mátrixot egy tömbben tároljuk el oszlopfolytonosan. A mátrixot egy fájlból kell feltölteni a típus egyik konstruktorában.

Írd meg az összeadás és szorzás műveleteket (operátokat)! Az összeadás és szorzás műveleteknél feltehetjük, hogy végrehajthatóak, ezt nem kell külön ellenőrizni (technikálisabban megfogalmazva: inkompatibilis méretű operandusokkal meghívni őket Undefined Behaviour). Írj egy transzponálást végrehajtó függvényt! Írj egy kiíró műveletet, ami az elemeket áttekinthető formában megjeleníti! Írj egy műveletet, aminek segítségével lekérhetők a mátrix méretei. Írj egy at nevű függvényt aminek segítségével elérhetőek a mátrix elemei, ha érvénytelen indexpárral hívják meg, ez a függvény dobjon egy `std::out_of_range` exceptiont.

Az osztály implementációjába beletartozik a megfelelő konstruktorok implementálása. A copy konstruktor és assignment operátorokat, ha úgy gondolod, hogy helyes implementációt ad, default kulcsszóval jelöld, ha nem, akkor megfelelően implementáld. A move konstruktor és assignment operátort delete-eld, vagy implementáld helyesen.

Bemeneti fájl formátum

A fájlban az első sorban a sorok száma (n) van, a második sorban az oszlopok száma (m), a fájl többi sorában (összesen még $n \times m$ sor van) pedig a beolvasandó értékek, sorfolytonosan. Figyelem! Megváltozik a sorrend, az indexekre gondolni kell.

Limitek

- Méret: $1 \leq n \leq 256$, $1 \leq m \leq 256$
- Időlimit: tesztesetenként 0.1s
- Memórialimit: 100 MiB

API

A feladat megoldásához implementáld a következő osztályt:

```
class matrix {
  /* TODO */
public:
  explicit matrix(std::filesystem::path);

  [[nodiscard]] matrix operator+(const matrix &) const noexcept;
  [[nodiscard]] matrix operator*(const matrix &) const noexcept;
  [[nodiscard]] int &at(size_t, size_t);
  [[nodiscard]] const int &at(size_t, size_t) const;
  [[nodiscard]] std::pair<size_t, size_t> size() const noexcept;
  void transpose() noexcept;
  void print() const noexcept;
};
```

A megadott függvények szükségesek a megoldáshoz, de nyugodtan bővíthetőek.

nodiscard, noexcept, explicit

A `[[nodiscard]]` egy függvény attribútum, a compiler jelez, ha a függvény eredményét nem használjuk semmire. Ez is csak magunk felé jelölés, olyan függvényeket érdemes megjelölni vele, amik kiszámolnak egy értéket és mást nem csinálnak, ezért ha az eredményét se használnánk, akkor nincs értelme megívni. További infó itt. Pl:

```
[[nodiscard]] int foo() { return 5; }

int main() {
  foo();          // warning: Ignoring return value of function declared with
                  // nodiscard attribute
  int a = foo();  // no warning
  int b = 3 + foo();  // no warning
}
```

A `noexcept` kulcsszó jelentése annyi, hogy az adott függvényből nem dobódhat kivétel (kivételekről a második gyakorlaton lesz szó).

A konstruktornál látható `explicit` kulcsszó az implicit castolást zárja ki.
A `matrix m = std::filesystem::path("in");` kifejezés nem fordul, hiszen nem castolható egy `path` mátrixá.

Ha zavarnak, ezek használata nem kötelező.

Woodpecker

A kiadott main a woodpecker-t használja, ami egy teszt keretrendszer. Minden `TEST` macro egy-egy teszteset-nek felel meg. Lényegében mindegyik 1-1 fügvény lesz, amit a main-ből meghívünk. A main-t a `WOODPECKER_TEST_MAIN` generálja a kódba. A macrok kifejtései a `woodpecker.hpp`-ben találhatóak, de ezekkel nem kell foglalkozni.

A `main.cpp`-t nyugodtan szerkeszthetitek, akár használhattok másikat is. A házi beadásakor a `main.cpp` egy kiegészített verziójával fognak futni a tesztek, ha minden tesztre `PASS` érkezik, akkor a szerveren a memóriakezelést is ellenőrizzük, az eredményekről `svn update` után (pár másodperc késleltetéssel) kaptok választ.

A keretrendszerrel és a házival kapcsolatban is nyugodtan kereshetitek a gyakorlatvezetőket vagy segédeket. Bármilyen észrevételt, javaslatot szívesen látunk.