Užduotis 4. Išimtinės situacijos

Nuo šiol klasėms reikalavimai:

1. Kiekvienos klasės antraštės ir implementacijos turi būti atskiruose failuose (.hpp ir .cpp).
2. Konstruktoriai pilnai sukuria objektą – priskiria visiems laukams reikšmes (kad ir *defaultinį* 0 ar pan.)
3. Visais klaidų atvejais metami *exceptionai* ir reikia išgaudyti (akivaizdžius) atvejus.

Truputį teorijos apie C++ išimtines situacijas (*exceptions*, klaidos). Klaidas metame ir apdorojame tokia tvarka

Kažkur kode, pvz., klasės viduje, metame klaidą

```
throw ( // throws an exception when a problem shows up  )
```

Kitur kode, pvz., kitoje klasėje ar `main`, tą klaidą/as sugauname. Normalus kodas yra `try` bloke, klaidas apdorojantis kodas - `catch`.

```
try {
   // protected code
}
catch( ExceptionName &e ) {  //Naudojame &, kad nebutu e kopijos
  // code to handle ExceptionName exception
}
catch(...) {
  // code to handle any exception
}
```

Pavyzdžiai

1. Korektiškai ir gerai išaiškintos priežastys, kodėl taip reikia naudoti https://stackoverflow.com/questions/2718779/correct-exceptions-in-c
2. Neblogai paaiškinta (tik nereiktų mesti *stringų*) https://www.tutorialspoint.com/cplusplus/cpp_exceptions_handling.htm
3. Exceptions veikia, bet visais atžvilgiais netinkamai. Šitaip nenaudokit! https://www.w3schools.com/cpp/cpp_exceptions.asp

1. Time/Date klasė su išimtinėmis situacijomis

1. Savo Time/Date klasėje perkelkite argumentų korektiškumo patikrinimus į setterius ir meskite išimtį `invalid_argument`, jei parametras netinkamas. Meskite išimtis visais galimais klaidos atvejais – tai turite užtikrinti pats programuotojas.

   Hint: nepamirškite `#include <stdexcept>`

2. Realizuokite `main`'e laiko/datos nuskaitymą į vektorių iš terminalo. Skaitymą tęskite, kol korektiškai įvedami *n* laikai/datos, *n* – nustatote patys. Jei įvedami nekorektiški duomenys, turi būti sugaunama klaida, išmetamas pranešimas parodantis, kur buvo netikslumas bei tęsiamas nuskaitymas (t.y. programa nesustoja/nelūžta).

   Pastaba: Tokiu būdu matysite, kad klaidos objektas keliauja per kodą setteris -> konstruktorius -> `main`.

3. Iš `main` pamėginkite su setteriu pakeisti kokį nors parametrą ir gaudykite klaidą. Pademonstruokite, kad klaidos objektas sėkmingai keliauja setteris -> `main`.

2. Taikomoji klasė su išimtimis

Visiems: pratęskite darbą su pasirinkta taikomąją klase.

Tiems, kas tos klasės nepadarė ar padarė nedaug: reikia išpildyti užduoties 3 reikalavimus (galima gauti papildomai taškų atsigriebiant už negautus užd. 3). Dirbti su šia klase reikės ir toliau, todėl teks padaryti ☹

1. Papildykite savo klasę 1 – 2 metodais, kurie kažką darytų specifiško jūsų klasei, pavyzdžiui, kažką skaičiuotų, dirbtų su *stringais*, keistų pačioje klasėje ir pan.

   Pavyzdys: studento klasei – metodas, skaičiuojantis pažymių vidurkį (aišku, klasėje turėtų atsirasti vektorius su pažymiais).

2. Pasirūpinkite, kad jūsų klasę apdorotų išimtines situacijas mesdama išimčių objektus. Realizuokite kokio nors kitos rūšies klaidos suradimą nei `invalid_argument`. Galimi variantai:
   - Bandome pasiekti neegzistuojantį vektoriaus/masyvo elementą.
   - Daliname iš 0.
   - Atidarinėjame failą, kurio nėra, ar failą, kuriame ne tokie duomenys, kaip norėjome.

   Pavyzdys: studento klasė meta `runtime_error`., kai bandoma skaičiuojant vidurkį neturint nei vieno pažymio (gautume dalybą iš 0).

3. Padarytus pakeitimus ištestuokite `main`. Pademonstruokite, kaip sugaunate klaidas. Parodykite naujai suprogramuotų specifinių klasės metodų veikimą.