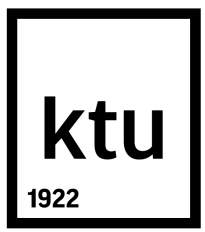
# Lygiagretusis programavimas (LP)
# P170B328
# Concurrent Programming
## Bendroji informacija apie modulį



# Paskaitos turinys



ktu 1922

# Dėstytojai

- Teorija
  - Karolis Ryselis
- Laboratoriniai darbai
  - Dominykas Barisas
  - Eligijus Kiudys
  - Evaldas Guogis
  - Karolis Ryselis
  - Mindaugas Jančiukas
  - Mindaugas Vasiljevas

# Apie dėstytoją

- Programuotojas UAB “Esperonus” nuo 2012 metų – patirtis dirbant su saityno technologijomis;
- Docentas – praktikas KTU nuo 2017 metų;
- Galima kreiptis tiek lygiagretaus programavimo klausimais, tiek ir bet kokiais kitais.

# Kodėl reikia mokytis kurti lygiagrečiąsias programas?

- Šiuolaikiniai procesoriai turi daug branduolių, kurie gali dirbti lygiagrečiai. Pvz.:
  - Intel Core i5-13400 — 10 fizinių / 16 loginių branduoliai
  - AMD Ryzen 9 5900X — 12 fizinių / 24 loginiai branduoliai
  - Qualcomm Snapdragon SM8550-AB — 8 fiziniai branduoliai
- Kiekvienas branduolys gali vykdyti po skirtingą užduotį vienu metu.
- Norint savo programoje išnaudoti aparatūrinės įrangos teikiamas galimybes, reikia kurti lygiagrečiąsias programas.

# Kodėl reikia mokytis kurti lygiagrečiąsias programas?

- Lygiagrečiosios programos gali veikti greičiau, nei nuoseklios.
- Pvz., dviejų masyvų visų elementų bendra suma su 15M elementų (24 loginiai branduoliai):
  - Java streams — 82 ms
  - Java parallel streams — 14 ms

# Kodėl reikia mokytis kurti lygiagrečiąsias programas?

- Ne visus uždavinius galima išspręsti naudojant tik nuoseklaus programavimo priemones.
- Tinklo serveris — nuoseklus serveris palaiko tik vieną prisijungimą vienu metu.
- Programos su grafine sąsaja — naudojant vieną giją grafinė sąsaja „užšąla“, jei programa kažką skaičiuoja.

# Tikslai ir uždaviniai

- susipažinti su gijų paleidimu ir vykdymu;
- susipažinti su bendros atminties valdymo principais;
- susipažinti su paskirstytos atminties modeliais;
- susipažinti su programavimu grafiniams procesoriams.

# Paskaitų tematika

- procesų (gijų) kūrimas, vykdymas, nutraukimas, naikinimas;
- bendra ir paskirstyta atmintis: procesų komunikavimas ir sinchronizavimas;
- lygiagretieji algoritmai;
- asmeninių kompiuterių ir superkompiuterių lygiagrečiojo programavimo kalbos ir priemonės;
- lygiagretumas grafikos procesoriuose;
- lygiagrečiojo ir funkcinio programavimo sąsaja;
- asinchroninis programavimas.

# Mokymo planas

**paskaitos** 32 val. (2 val. per sav.);

**lab. darbai** 32 val. (2 val. per sav.);

**sav. darbas** 96 val. (lab. darbai, inžinerinis projektas, egzaminas);

**kreditai** 6.

# Priemonės dėl COVID-19 prevencijos

- Auditorijoje stengtis neliesti kompiuterio ir monitoriaus įjungimo/išjungimo mygtukų.
- Rekomenduojama auditorijoje visada sėsti į tą pačią vietą.

# Literatūra (knygos)


- **Javier Fernández González**. Mastering Concurrency Programming with Java 9. Packt Publishing, 2017.
- **Stephen Cleary**. Concurrency in C# Cookbook, 2014.
- **David B. Kirk, Wen-mei W. Hwu**. Programming Massively Parallel Processors (2nd Edition). Elsevier Inc., 2013.
- **Simon Marlow**. Parallel and Concurrent Programming in Haskell, O'Reilly, 2013.
- **Riccardo Terrell**. Concurrency in .NET. Manning Publications, 2018.
- **Anthony Williams**. C++ Concurrency in Action: 2nd edition. Manning Publications, 2019.

# Literatūra (interneto šaltiniai)


- **OpenMP**. http://www.openmp.org/.
- **Google Go**. The Go Programming Language. http://golang.org/.
- **CUDA**. CUDA Toolkit Documentation. https://docs.nvidia.com/cuda/.
- **MPI.** Message Passing Interface Forum. http://www.mpi-forum.org/.
- **Paskaitų medžiaga.** https://moodle.ktu.edu/
- **Paskaitų kodo pavyzdžiai.** https://bitbucket.org/ryselis/p170b328-code-examples/src/master/

# LD ir IP programavimo kalbos

1. **Numatytosios:**
   - laboratorinių darbų programoms: C++, C#, C++ & OpenMP, C++ & CUDA, Google Go, Rust.
   - inžinerinio projekto programai: su dėstytoju suderinta laboratorinių darbų metu nenaudota programavimo kalba.
2. **Individualiai pasirenkamos**[1].

[1] pasirinkimą reikia derinti su dėstytoju

# Laboratoriniai darbai

- Laboratoriniai darbai susideda iš 1 arba 2 programų ir kontrolinio.
- Programos pristatomos laboratorinių darbų dėstytojui pagal tvarkaraštį.
- Kontroliniai vyksta paskaitų metu.
- Programų pristatymui semestro metu suteikiami du bandymai, kontroliniams — **vienas**.

# Laboratoriniai darbai

- L1 — bendra atmintis, susideda iš **2 programų** ir **kontrolinio** (prie auditorijos kompiuterių suprogramuoti duotą užduotį). Didžiausios apimties lab. darbas.
- L2 — paskirstyta atmintis, susideda iš **1 programos** ir **kontrolinio** (prie auditorijos kompiuterių suprogramuoti duotą užduotį).
- L3 — GPU programavimas, susideda iš **1 programos** ir **kontrolinio** (testas Moodle, 20 min.).

# Laboratorinių darbų atsiskaitymo tvarka

- Programos demonstruojamos dėstytojui lab. darbų metu.
- Studentas pristato programą, dėstytojas užduoda klausimus.
- Ataskaita L1, L2 ir L3 darbams nereikalinga – pakanka pateikti programą ir duomenų / rezultatų failus.
- L1 ir L2 kontroliniai rašomi prie auditorijos kompiuterių 90 min. lab. darbų metu paskirtą savaitę.
- L3 kontrolinis rašomas 20 min. prisijungus prie Moodle iš auditorijos kompiuterio.

# Individualus projektas

- Priemonės, su kuriomis atliekamas, derinamos su dėstytoju.
- Galima atsiskaityti išlygiagretintą skaitinių metodų lab. darbą arba savo pasirinktą ir su dėstytoju suderintą užduotį.
- Reikalinga ataskaita.
- Išlygiagretinti duomenų lygiagretumo uždavinį ir išanalizuoti realizacijos vykdymo parametrus
- Galima rinktis:
  - išlygiagretinti skaitinių metodų lab. darbą
  - išlygiagretinti pasirinktą, su dėstytoju suderintą uždavinį

# LD ir IP programų pateikimo tvarka

- LD ir IP programų, duomenų bei rezultatų failus įkelti į LP puslapį moodle.ktu.edu;
- Failus įkelti:
  - LD: iki programos atsiskaitymo savaitės pabaigos;
  - IP: iki darbo pateikimo savaitės pabaigos.
- LD reikia įkelti kodo failus (.cpp, .rs, .cs, .go ir t.t.), duomenų ir rezultatų failus (galima įkelti zip archyvą).
- IP reikia įkelti kodo failus, duomenų, rezultatų failus ir ataskaitą (pdf.)

# L1: Bendra atmintis

**C++, Rust, C#, Go** 4 taškai (monitoriai)

**OpenMP** 2 taškai (kritinės sritys arba užraktai)

**kontrolinis** 4 taškai (laivai pasirenkamos priemonės)

# L2: Paskirstyta atmintis

**MPI, Go** 6 taškai (kanalai arba žinutės)

**kontrolinis** 4 taškai (laisvai pasirenkamos priemonės)

# L3: Lygiagretusis programavimas CUDA

**CUDA** 6 taškai (programavimas GPU)

**testas** 4 taškai (12 klausimų Moodle)

# Galutinis įvertinimas

- 3 lab. darbai — 40%;
  - L1 — 20%;
  - L2 — 10%;
  - L3 — 10%;
- projektas — 20%;
- egzaminas — 35%;
- aktyvumas paskaitose — 5%;
- įskaita:
  - LD ir IP įvertinimai (kiekvienas) $\geq 5$.

# Egzaminas (35%)

- kompiuteriu/raštu sesijos metu:
  - Du programavimo klausimai po 10 taškų;
  - Trys atviri teoriniai klausimai po 5 taškus.
- individuali užduotis sesijos metu tiems, kam po 16 sav. AIS siūlomas pažymys $\geq$ 5,2 (maksimalus galimas – 6,5):
  - programos realizacija ir gynimas — 30%;
  - atsakymas į egzamino klausimą — 5%;

# Paskaitų lankomumas

- Kiekvienos paskaitos pabaigoje bus testas iš tos paskaitos medžiagos.
- Įtaka galutiniam pažymiui — 5%.
- Testo tikslai — skatinti aktyvumą, tikrinti lankomumą, gauti grįžtamąjį ryšį.
- Nereikalaujama surinkti pusės taškų.

# Atsiskaitymų grafikas

**5 sav.** L1 a programos gynimas.

**6 sav.** L1 b programos gynimas.

**8 sav.** L1 kontrolinis.

**10 sav.** L2 programos gynimas.

**12 sav.** L2 kontrolinis.

**13–14 sav.** individualaus projekto gynimas.

**15 sav.** L3 programos gynimas.

**16 sav.** L3 kontrolinis.

Su dėstytoju galima **iš anksto** suderinti individualų grafiką.

# Pakartotinių atsiskaitymų grafikas

**7 sav.** L1 a ir b programų pakartotinis gynimas.

**11 sav.** L2 programos pakartotinis gynimas.

**16 sav.** L3 programos pakartotinis gynimas ir individualaus projekto pakartotinis gynimas.