Optymalizacja Kodu Na Różne Architektury
Zadanie 2

1 Ogólne zasady zaliczenia

- Zadanie oceniane jest w skali od 0-50 punktów.
- Warunkiem koniecznym jest indywidualne oddanie zadania.
- Ocenie podlega m.in. stopień zrozumienia mechanizmów odpowiedzialnych za obserowanane zmiany w wydajności.
- Weryfikacja posiadanego modelu procesora i dopasowanie do jego specyfikacji zooptymalizowanego kodu (cache i jednostki wektorowe, w tym ich typ i generacja) jest konieczne do uzyskania pozytywnej oceny.
- Zadania umieszczone w systemie Moodle/UPEL po terminie będą bezwględnie oceniane na maksymalnie 25 punktów.

2 Zadanie

Proszę o wykonanie optymalizacji algorytmu LU-dekompozycji wzorując się na ćwiczeniu 2, ćwiczeniu 3 oraz poniższym instruktażu.
https://github.com/flame/how-to-optimize-gemm
Proszę o umieszczenie w systemie UPEL/Moodle sprawozdania w formacie pdf. Mile widzine będą zwięzłe sprawozdania.

Można użyć jako startowego kodu do optymalizacji funkcji **LUPDecompose** z angielskiej strony wikipedii.
https://en.wikipedia.org/wiki/LU_decomposition

Wzorując się na ćwiczeniu 3 proszę napisać prosty algorytm weryfikujący, czy dla zadanego rozmiaru macierzy nie zmienia się zwracane przez optymalizowaną funkcję rozwiązanie.