GDAŃSK UNIVERSITY
OF TECHNOLOGY

Subject card

Subject name and code	Concurrent and parallel programming, PG_00060227		
Field of study	Technical Physics		
Date of commencement of studies	October 2023	Academic year of realisation of subject	2025/2026
Education level	first-cycle studies	Subject group	Optional subject group Subject group related to scientific research in the field of study
Mode of study	Full-time studies	Mode of delivery	at the university
Year of study	3	Language of instruction	Polish
Semester of study	5	ECTS credits	3.0
Learning profile	general academic profile	Assessment form	assessment
Conducting unit	Division of Computational Chemical Physics -> Institute of Physics and Applied Computer Science -> Faculty of Applied Physics and Mathematics -> Wydziały Politechniki Gdańskiej		
Name and surname of lecturer (lecturers)	Subject supervisor	dr hab. Jan Franz	
	Teachers		

Lesson types and methods of instruction	Lesson type	Lecture	Tutorial	Laboratory	Project	Seminar	SUM
	Number of study hours	15.0	0.0	30.0	0.0	0.0	45
	E-learning hours included: 0.0						
	eNauczanie source addresses: Moodle ID: 1308 Programowanie współbieżne i równoległe https://enauczanie.pg.edu.pl/2025/course/view.php?id=1308						

Learning activity and number of study hours	Learning activity	Participation in didactic classes included in study plan	Participation in consultation hours	Self-study	SUM
	Number of study hours	45	5.0	25.0	75

Subject objectives	The course introduces the principles and techniques of concurrent and parallel programming in Java, with special emphasis on their application in simple physics-related problems. Students learn to design, implement, and analyze multi-threaded programs, understand classical synchronization issues and design patterns, and evaluate performance and scalability of parallel solutions.

Learning outcomes

Course outcome	Subject outcome	Method of verification
[K6_K01] Understands the need to learn and improve professional and personal competencies. Can inspire and organize other people’s learning process	The student knows the scenarios where the use of concurrent or parallel programming is necessary to solve the problem.	[SK5] Assessment of ability to solve problems that arise in practice
[K6_W05] Has knowledge of programming methodology and techniques, and the use of selected IT tools in physics and technology.	The student learns how to solve classical concurrency problems.	[SW1] Assessment of factual knowledge
[K6_U03] Knows programming languages and can use basic software packages	The student practices concurrent programming using the Java programming language.	[SU1] Assessment of task fulfilment

Subject contents	1. Introduction and Motivation Lecture: Introduction to concurrency, motivation from physics simulations (many particles, many events), overview of threads and processes. Lab: Create and run simple threads, simulate parallel projectiles in free fall. 2. Thread Life Cycle Lecture: Thread states, creation and termination, control methods (start, join, sleep, interrupt), performance overheads. Lab: Implement harmonic oscillators in parallel, compare runtime with sequential execution. 3. Synchronization Basics Lecture: Race conditions, critical sections, and basic synchronization with synchronized. Lab: Simulate radioactive decay with many nuclei, update shared counter safely. 4. Performance Scaling Lecture: Amdahls Law and Gustafsons Law, parallel speedup, efficiency in physics computations. Lab: Monte Carlo π estimation with different numbers of threads, compare measured speedup with theoretical scaling. 5. Locks and Conditions Lecture: Lock objects, reentrant locks, and condition variables for controlled access. Lab: Simulate random walkers on a line, coordinate updates using locks and conditions. 6. ProducerConsumer Pattern Lecture: Classical problem of producerconsumer, blocking queues, fairness, avoiding overflow/underflow. Lab: Implement detectoranalyzer system, where producers generate particle events and consumers analyze them. 7. Case Study I: Parallel Data Analysis Lecture: Parallel collection and processing of data, synchronization strategies in physics data analysis. Lab: Implement parallel histogramming of particle energies using thread-safe collections. 8. ReaderWriter Pattern Lecture: Readerwriter problem, managing concurrent read and write access, fairness and starvation. Lab: Implement shared dataset of measurements with concurrent readers and occasional writers. 9. Circular Resource Allocation Lecture: Abstract formulation of circular resource allocation, deadlock, livelock, starvation, and resolution strategies. Lab: Implement system of concurrent processes each requiring two resources, demonstrate deadlock and apply avoidance strategies. 10. Thread Pools and Executors Lecture: Executor framework, fixed and cached pools, tasks, callables, and futures. Lab: Simulate many random walks with a thread pool, use futures to collect displacement statistics. 11. Advanced Synchronization Tools Lecture: Semaphores, barriers, and phased coordination of concurrent tasks. Lab: Synchronize harmonic oscillators at timesteps using barriers, simulate wave-like collective evolution. 12. ForkJoin Framework Lecture: Divide-and-conquer algorithms, ForkJoin tasks, and recursive decomposition. Lab: Implement ForkJoin summation of a large dataset, e.g. particle counts from simulated experiments. 13. Parallel Streams and Arrays Lecture: Stream API, parallel operations, reduction and aggregation. Lab: Compute kinetic energies of many particles in parallel, reduce results to averages and maxima. 14. Case Study II: Parallel Physics Simulations Lecture: Larger examples of parallel simulations, e.g. diffusion, Ising model, scattering. Lab: Implement 2D diffusion of particles in parallel, measure performance scaling. 15. Summary and Outlook Lecture: Recap of concurrency concepts, design patterns, and performance lessons; outlook to HPC and clusters. Lab: Integrated challenge: simulate radioactive decay with histogramming and analyze parallel efficiency vs. sequential.		
Prerequisites and co-requisites			
Assessment methods and criteria	Subject passing criteria	Passing threshold	Percentage of the final grade
	Laboratory: programming exercises	50.0%	75.0%
	Lecture: exam	50.0%	25.0%
Recommended reading	Basic literature	1. B. Wittman, T. Korb, A. Mathur, Start Concurrent: An Introduction to Problem Solving in Java with a Focus on Concurrency, Purdue University Press, Ashland, Oregon, 2014. 2. T. Rauber, G. Rünger, Parallel Programming: for Multicore and Cluster Systems, Springer Nature, Berlin, 2010. 3. S. Selikoff, J. Boyarsky, OCA/OCP Java SE 8 programmer: practice tests, Sybex, Indianapolis, Indiana, 2017.	

	Supplementary literature	1. M. Ben-Ari, "Principles of Concurrent and Distributed Programming", 2nd edition, Addison-Wesley, Upper Saddle River, NJ, 2006. 2. R.-G. Urma, M. Fusco, A. Mycroft, Modern Java in Action, Manning Publications, Shelter Island, 2018. 3. B. Goetz, T. Peierls, J. Bloch, J. Bowbeer, D. Holmes, D. Lea, Java Concurrency in Practice. Addison-Wesley, Upper Saddle River, NJ, 2006. 4. B. J. Evans, J. Clark, M. Verburg, The Well-Grounded Java Developer, Second Edition, Manning Publications, Shelter Island, 2023
	eResources addresses	Basic https://docs.oracle.com/javase/tutorial/essential/concurrency/ - This is Oracle’s official Java tutorial on concurrency. It introduces the basic concurrency support in Java (threads, synchronization, locks) and reviews higher-level concurrency utilities in the java.util.concurrent package (executors, lock objects, concurrent collections, Fork/Join, atomic variables, etc.). It also covers common issues such as thread interference, memory consistency errors, deadlock, livelock, and conditional synchronization (guarded blocks).
Example issues/ example questions/ tasks being completed	1. Calculate the Speedup of a concurrent program. 2. Convert a sequential program in a concurrent program.	
Work placement	Not applicable	

Document generated electronically. Does not require a seal or signature.