COURSE OUTLINE

(1) GENERAL

SCHOOL	ENGINEERING		
ACADEMIC UNIT	ELECTRICAL AND COMPUTER ENGINEERING DEPT.		
LEVEL OF STUDIES	Undergraduate		
COURSE CODE	**ECE_INF920**	**SEMESTER**	**9**
COURSE TITLE	Parallel Systems & Programming		

INDEPENDENT TEACHING ACTIVITIES *if credits are awarded for separate components of the course, e.g. lectures, laboratory exercises, etc. If the credits are awarded for the whole of the course, give the weekly teaching hours and the total credits*	**WEEKLY TEACHING HOURS**	**CREDITS**
Lectures	3	
Seminars / Practice exercises	1	
Laboratory		
Add rows if necessary. The organisation of teaching and the teaching methods used are described in detail at (4).	4	5

COURSE TYPE *general background, special background, specialised, general knowledge, skills development*	Special Background, Skills development
PREREQUISITE COURSES:	No. Students are advised to have already attended the courses: Procedural Programming (ECE_K260), Data Structures & Algorithms(ECE_K320),Object-Oriented Programming (ECE_K430) and Operating Systems (ECE_K540)
LANGUAGE OF INSTRUCTION and EXAMINATIONS:	Greek
IS THE COURSE OFFERED TO ERASMUS STUDENTS	Yes
COURSE WEBSITE (URL)	https://eclass.uop.gr/courses/ECE120/

(2) LEARNING OUTCOMES

Learning outcomes

The course learning outcomes, specific knowledge, skills and competences of an appropriate level, which the students will acquire with the successful completion of the course are described.

Consult Appendix A

- *Description of the level of learning outcomes for each qualifications cycle, according to the Qualifications Framework of the European Higher Education Area*
- *Descriptors for Levels 6, 7 & 8 of the European Qualifications Framework for Lifelong Learning*

and Appendix B

- *Guidelines for writing Learning Outcomes*

The course aims to provide students with knowledge in the use of concurrent and parallel processing, present basic parallel algorithms, parallel programming models and their applications on shared and distributed memory architectures.

Upon successful completion of the course the student will:

- Know the basic parallel architectures.
- Distinguish the different categories of Parallel Systems and describe their advantages and disadvantages

- Will understand the models, limitations and fundamental concepts of parallel and concurrent programming in message passing and shared memory environments and will be able to apply this knowledge to implement various algorithms and systems
- Understand the use of critical areas and mutual exclusion and when it is required in parts of a parallel application
- Distinguish different kinds of data dependencies
- Understanding synchronization problems of concurrent systems.
- Can design and customize algorithms for distributed parallel environments.
- Can carry out basic analysis of parallel algorithms for their accuracy, reliability and performance
- Will know the basic steps required to write parallel programs and will be able to apply them in real codes.
- Implements a parallel program using Message Passing (MPI) and shared memory (PThreads, OpenMP, OpenCL, CUDA) programming libraries and environments.

General Competences

Taking into consideration the general competences that the degree-holder must acquire (as these appear in the Diploma Supplement and appear below), at which of the following does the course aim?

Search for, analysis and synthesis of data and information, with the use of the necessary technology	*Project planning and management*
Adapting to new situations	*Respect for difference and multiculturalism*
Decision-making	*Respect for the natural environment*
Team work	*Showing social, professional and ethical responsibility and sensitivity to gender issues*
Working in an international environment	*Criticism and self-criticism*
Working in an interdisciplinary environment	*Production of free, creative and inductive thinking*
Production of new research ideas	*……*
	Others…
	…….

- Working independently
- Working in an international environment
- Production of free, creative and inductive thinking
- Production of new research ideas

(3) SYLLABUS

The course deals with the programming techniques of parallel systems and more specific systems that have multiple computing cores within a processing unit (multicores and many-cores). It focuses with the programming of conventional and non-conventional, homogeneous and heterogeneous parallel architectures. Students are introduced to performance measurement techniques, profiling, experimental evaluation, optimization and the interaction of software with the underlying hardware.

1. The model of concurrent execution. The problem of mutual exclusion.
2. Concurrent programming and synchronization in C/PThreads and Java.
3. Basic elements of parallel system architectures, classification of parallel systems, characteristic conventional and non-conventional multi-core architectures. Basic metrics, Amdahl and Gustafson Laws, Karp-Flatt Metric .
4. Programming using OpenMP.
5. Introduction to parallel and network algorithms.
6. Design and programming of parallel and network algorithms.
7. Programming using MPI I.
8. Load balancing problems, synchronization, communication costs and scaling.
9. Programming using MPI II.
10. Analysis, evaluation, and predicting behavior of parallel algorithms and calculations.

11. Examples of parallel algorithms for search, sorting, finding shortest paths in graphs and optimization
12. Programming models for many-core parallel systems
13. Programming using OpenCL, CUDA.

(4) TEACHING and LEARNING METHODS - EVALUATION

DELIVERY *Face-to-face, Distance learning, etc.*	Face-to-face
USE OF INFORMATION AND COMMUNICATIONS TECHNOLOGY *Use of ICT in teaching, laboratory education, communication with students*	• The e-Class platform is used for asynchronous support of the learning process (course operations regulation, distribution of slides, project related material, complementary material,announcements, links and bibliography, etc.). • During lectures a projector and presentations in electronic form are used, which are also posted in the eclass platform. • During lectures and especially at seminars, a computer is used to write and execute example code.
TEACHING METHODS *The manner and methods of teaching are described in detail.* *Lectures, seminars, laboratory practice, fieldwork, study and analysis of bibliography, tutorials, placements, clinical practice, art workshop, interactive teaching, educational visits, project, essay writing, artistic creativity, etc.* *The student's study hours for each learning activity are given as well as the hours of non-directed study according to the principles of the ECTS*	

Activity	***Semester workload***
Lectures	13*3 = 39
Seminars	13*1 = 13
Project	7*4=28
Study and analysis of bibliography	45
Course Total	***125 hours (5 ECTS)***

STUDENT PERFORMANCE EVALUATION *Description of the evaluation procedure* *Language of evaluation, methods of evaluation, summative or conclusive, multiple choice questionnaires, short-answer questions, open-ended questions, problem solving, written work, essay/report, oral examination, public presentation, laboratory work, clinical examination of patient, art interpretation, other*	Language of evaluation: Greek (English if required) Preparation of 5 programming exercises using parallel environments and delivery of reports (30% of final mark). Final examination (70% of final mark): Written exam that might include multiple choice questionnaires, short-answer questions, problem solving. Grading scale: 1 to 10

Specifically-defined evaluation criteria are given, and if and where they are accessible to students.	Minimum passing mark: 5 Grades < 5 correspond to ECTS grade F. For the rest of the grades the correspondence is as follows: >= 9: ECTS A >= 8: ECTS B >= 7: ECTS C >= 6: ECTS D >= 5: ECTS E Note: The evaluation criteria are clearly announced during the first lecture of the semester as well as the individual percentages and the rules governing the grading, both of the above evaluation methods and of the individual percentages of the programming exercises.

(5) ATTACHED BIBLIOGRAPHY

- Suggested bibliography:

- «Introduction to Parallel Programming», PETER S. PACHECO, ISBN 0-123-742-609, Morgan Kaufman, 2011
- «Programming Massively Parallel Processors 2e», D.B. Kirk – W.W. Hwu, ISBN 9-789-351-070-443, ELSEVIER , 2013
- «Parallel Systems & Programming», V. Dimakopoulos, , ISBN 978-960-603-369-8, 2015, http://hdl.handle.net/11419/3209 (In Greek)

- Related academic journals:

- IEEE Transactions on Parallel Processing
- Concurrency: Practice and Experience, Wiley
- Parallel Computing, Elsevier
- International Journal of Parallel Programming, Springer
- IEEE Transactions on Parallel and Distributed Systems
- ACM Transactions on Parallel Computing
- Journal of Parallel and Distributed Computing