COURSE OUTLINE

(1) GENERAL

SCHOOL	School of Engineering		
ACADEMIC UNIT	Informatics and Computer Engineering		
LEVEL OF STUDIES	Undergraduate		
COURSE CODE		**SEMESTER**	7th, 9th
COURSE TITLE	Parallel Systems		
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**
		4	5
Add rows if necessary. The organisation of teaching and the teaching methods used are described in detail at (d).			
COURSE TYPE general background, special background, specialised general knowledge, skills development	Specialised general knowledge, skills development		
PREREQUISITE COURSES:	Operating Systems I & II, Introduction to Parallel Computing		
LANGUAGE OF INSTRUCTION and EXAMINATIONS:	Greek		
IS THE COURSE OFFERED TO ERASMUS STUDENTS	Yes (in English)		
COURSE WEBSITE (URL)	https://eclass.uniwa.gr/courses/CS123/		

(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 present modern parallel systems and architectures and their main programming techniques and tools. It is also specifically aimed at learning integrated parallelization techniques for shared/distributed memory environments and hybrid architectures, as well as at the analysis, design and implementation of massively parallel algorithms for classical computing problems. An additional important goal is to provide students with the necessary knowledge and skills for programming and developing efficient parallel applications in modern parallel systems, and the application of all of the above in fields of real problems and further research.

Upon successful completion of the course, the student will be able to:

- To identify the most important of the modern classes/architectures of parallel systems and to describe their main characteristics.
- Distinguish between parallelization techniques in shared and distributed memory environments.
- To design efficient parallel algorithms for distributed memory environments.
- To describe the different models of parallel programming and distinguish their differences and their main advantages.
- To delve into the main programming models taught and put them into practice.
- To utilize modern parallel programming tools to develop efficient parallel applications.
- To program in modern parallel environments, such as multi-core systems, multi-

computers, GPUs/co-processors, etc).
- To measure the performance of the parallel programs and applications it implements with modern measures, techniques and tools.

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 Respect for difference and multiculturalism Respect for the natural environment
Adapting to new situations	Showing social, professional and ethical responsibility and sensitivity to gender issues
Decision-making	Criticism and self-criticism
Working independently	Production of free, creative and inductive thinking
Team work	……
Working in an international environment	Others…
Working in an interdisciplinary environment	…….
Production of new research ideas	

- Autonomous work
- Team work
- Decision-making
- Adaptation to new situations
- Project planning and management
- Work in an interdisciplinary environment
- Production of new research ideas
- Promotion of free, creative and inductive thinking

(3) SYLLABUS

Theory:

- Modern parallel systems and architectures – supercomputers, multicore systems, clusters, hybrid systems and architectures.
- Many-core architectures and modern accelerator-coprocessor technologies (CUDA GPUs, Intel Xeon Phi, Sunway, etc.).
- Parallelization techniques in distributed memory environments, parallel programming models and parallel programs design.
- Solving classical computing problems in a distributed memory environment (sorting algorithms, matrix multiplication algorithms, and algorithms for solving linear systems).
- Parallel (multithreaded) programming in shared memory environments (using the OpenMP API).
- Programming GPUs and coprocessors (the CUDA programming model, the OpenCL standard, Intel Xeon Phi programming).
- Measuring the Performance of Parallel Programs.

Lab:

Selected Exercises – Application Development using modern parallel programming tools (OpenMP, CUDA, OpenCL, etc.)

(4) TEACHING and LEARNING METHODS - EVALUATION

DELIVERY Face-to-face, Distance learning, etc.	Face-to-face. Use of distance learning (if required)
USE OF INFORMATION AND COMMUNICATIONS TECHNOLOGY Use of ICT in teaching, laboratory education, communication with students	Use of ICT in Course Teaching and Laboratory Education, Use of ICT in Communication with Students Post course material on the University's e-learning platform (e-class). Use of email and e-class in communication with students.
TEACHING METHODS The manner and methods of teaching are described in detail. Lectures, seminars, laboratory	

Activity	Semester workload
Lectures	26
Problem Solving in Class	13
Laboratory Education	13

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	<table><tr><td>Lab exercises</td><td>33</td></tr><tr><td>Non-guided study</td><td>40</td></tr><tr><td>Course total</td><td>125</td></tr><tr><td></td><td></td></tr><tr><td></td><td></td></tr></table>
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 Specifically-defined evaluation criteria are given, and if and where they are accessible to students.	Final Grade = (70% * Grade of the Theory Part) + (30% * Grade of the Laboratory Part) *Evaluation Process of Theory Part*: Final written exam at the end of the semester *Evaluation Process of Laboratory Part*: Preparation of laboratory exercises / assignments and oral or written examination ****Note: a portion of up to 35% on the Theory Part may be assessed by a research project (on-a-students-demand basis)* The evaluation process is disclosed to the students in class and online, via e-class.

(5) ATTACHED BIBLIOGRAPHY

1. Πάντζιου Γ., Μάμαλης Β., Τομαράς Α., Εισαγωγή στον Παράλληλου Υπολογισμό: Πρότυπα, Αλγόριθμοι, Προγραμματισμός, Εκδόσεις Νέων Τεχνολογιών, 2013 (in greek).
2. Παπαδάκης Σ., Διαμαντάρας Κ., *Προγραμματισμός και Αρχιτεκτονική Συστημάτων Παράλληλης Επεξεργασίας*, Εκδόσεις Κλειδάριθμος, 2012 (in greek).
3. Kirk D.B., Hwu W.M., *Programming Massively Parallel Processors*, 3rd edition, Morgan Kaufmann, 2016.
4. Grama A., Gupta A., Karypis D., Kumar V., *Introduction to Parallel Computing*, 2nd edition, Addison-Wesley, 2003.
5. Quinn M.J., *Parallel Programming in C with MPI and OpenMP*, McGraw-Hill, 2003.
6. Wilkinson B., Allen M., *Parallel Programming – Techniques and Applications Using Networked Work stations and Parallel Computers*, Pearson, Prentice Hall, 2006.
7. Andrews G.R., *Foundations of Multithreaded, Parallel and Distributed Programming*, Addison-Wesley, 2000.
8. Δημακόπουλος, Β., 2015. Παράλληλα συστήματα και προγραμματισμός [e-book] http://hdl.handle.net/11419/3209 (in greek).
9. James Jeffers, James Reinders, and Avinash Sodani, Intel Xeon Phi Processor High Performance Programming, Morgan Kaufmann, 2nd Edition, 2016.
10. LLNL OpenMP Tutorial, http://www.llnl.gov/computing/ tutorials/openMP/
11. OpenCL Specification, https://www.khronos.org/registry/OpenCL/specs/opencl-2.1.pdf
12. NVIDIA CUDA Tutorial: http://www.nvidia.com/docs/IO/116711/sc11-cuda-c-basics.pdf