ANX-PR/CL/001-01

LEARNING GUIDE

SUBJECT

615000242- Concurrent and Advanced Programming

DEGREE PROGRAMME

61IW – Degree in Software Engineering

ACADEMIC YEAR & SEMESTER

2022/23 - Semester 2

Index

Learning guide

1. Description

1.1. Subject details

Name of the subject	615000242- Concurrent and Advanced Programming
No of credits	6 ECTS
Type	Mandatory
Academic year ot the programme	Second year
Semester of tuition	Semester 4th
Tuition period	February - June
Tuition languages	English
Degree programme	61IW – Software Engineering Bachelor
Centre	61 - Escuela Tecnica Superior De Ingenieria De Sistemas Informaticos
Academic year	2022-23

2. Faculty

2.1. Faculty members with subject teaching role

Name and surname	Office/Room	Email	Tutoring hours *
Jorge Enrique Pérez Martinez (Subject coordinator)	4415	jorgeenrique.perez@upm.es	To be confirmed.
Maria Isabel Muñoz Fernandez	4412	isabel.munoz@upm.es	To be confirmed.
Pilar Manzano Garcia	4412	pilar.manzano@upm.es	To be confirmed.

* The tutoring schedule is indicative and subject to possible changes. Please check tutoring times with the faculty member in charge.

3. Prior knowledge recommended to take the subject

3.1. Recommended (passed) subjects

It is recommended some knowledge in Programming Fundamentals and Oriented Object Programming.

3.2. Other recommended learning outcomes

4. Skills and learning outcomes *

4.1. Specific outcomes of instruction

- Explain the usefulness of concurrent programming in the contexts of parallel programming and real-time systems.
- Use the different message passing models to solve concurrent programming problems, selecting the most appropriate one in each situation.
- Use monitors as a high-level tool to solve concurrent programming problems.
- Given a short-term learning objective, autonomously identifies the necessary knowledge and skills and establishes a plan to achieve it. The plan integrates the selection of information sources, methods of searching for timely information and criteria for synthesizing the information gathered.
- Apply the theories and techniques necessary to correctly develop concurrent programs.
- Detect deadlock situations and apply mechanisms to cope with such situations.
- Be able to develop applications that run concurrent threads using different tools.
- Identify the elementary algorithms to solve basic concurrent programming problems and the limitations of such solutions to solve concurrent programming problems.
- Be able to use semaphores as a low-level tool to solve concurrent programming problems.

4.2. Competencies

- CC14 - Knowledge and application of fundamental principles and basic techniques of parallel, concurrent, distributed, and real-time programming.
- CT7 - Autonomous learning: The student must take responsibility for their own learning, which leads them to use cognitive processes strategically and flexibly, depending on the learning objective.

4.3. Learning outcomes

- RA316 - Explains the usefulness of concurrent programming in the contexts of parallel programming and real-time systems.
- RA315 - Uses different message passing models to solve problems in concurrent programming, selecting the most appropriate one for each situation.
- RA188 - Uses monitors as a high-level tool to solve problems in concurrent programming.
- RA194 - Given a short-term learning objective, autonomously identifies the necessary knowledge and skills and establishes a plan to achieve it. The plan integrates the selection of information sources, methods for timely information retrieval, and criteria for synthesizing the collected information.
- RA184 - Applies the necessary theories and techniques to develop concurrent programs correctly.
- RA314 - Detects deadlock situations and applies mechanisms to address such situations.
- RA185 - Develops applications that execute concurrent threads using different tools
- RA186 - Identifies elementary algorithms to solve basic problems in concurrent programming and the limitations of such solutions in addressing larger-scale problems.
- RA187 - Uses semaphores as a low-level tool to solve problems in concurrent programming..

5. Brief description of the subject and syllabus

5.1. Brief description of the subject

A "concurrent program" can be defined as a set of sequential programs that can be executed in parallel (M. Ben-Ari). This set of sequential programs (or processes) can be executed on several processors (true parallelism) or on a single processor (pseudo- parallelism). In either case, as soon as there is cooperation between two or more processes or shared resources, the need arises to synchronize these concurrent processes in order to achieve the expected results.

Syllabus

1 Introduction to concurrent programming
 1.1. Computer SW Levels
 1.2. Programming languages
 1.3. Memory management in the operating system
 1.4. Process management in the operating system
 1.5. Threads within a process
 1.6. Concurrent applications
2. Abstraction of concurrent programming
 2.1. The Role of Abstraction
 2.2. Concurrent execution as interleaving atomic sentences
 2.3. Justification of Abstraction
 2,4. Arbitrary interleaving
 2.5. Atomic sentences
 2.6. Correctness
 2.7. Fairness
 2.8. Machine code instructions
 2.9. Volatile and Non-Atomic Variables
3. The critical section problem
 3.1. Introduction and definition of the problem
 3.2. First attempt
 3.3. Correctness test with state diagrams
 3.4. Further attempts
 3.5. Dekker's algorithm
4. Semaphores
 4.1. States of a process
 4.2. Semaphore Definition
 4.3. Critical section problem for two processes
 4.4. Invariants of the semaphores
 4.5. Critical section problem for N processes
 4.6. Execution order problems
 4.7. Producer-consumer problem
 4.8. Other semaphore definitions
 4.9. Philosophers' dinner problem
5. Monitors
 5.1. Introduction
 5.2. Declaration and use of monitors
 5.3. Condition variables
 5.4. Producer-Consumer Problem
 5.5. Immediate Resumption Problem
 5.6. Reader-writer problem
 5.7. Philosophers' Dinner Problem
 5.8. Monitors in Scala
6. Channels and messages
 6.1. Communication models

6.2. Channels
6.3. Client-Server Interaction
6.4. Channels and actors
7. Interlocks
7.1. System model
7.2. Characterization of Interlocks
7.3. Methods for handling interlocks
7.4. Interlock Prevention
7.5. How to avoid deadlocks
7.6. Detection of interlocks
7.7. Interlock recovery
8. Asynchronous programming with futures
8.1. Futures
8.2. Callbacks
8.3. Composition of futures
9. Advanced programming
9.1. Parallel programming
9.2. Distributed programming
9.3. Software transactional memory

6. Schedule

6.1. Subject schedule*

Week	Classroom activities	Laboratory activities	Distant / On-line	Assessment activities
1	**Subject presentation** Lecture **Chapter 1**	Lab 0: Introduction to the lab environment, the tools that will be used, and downloading and using them. Duration: 02:00 SL: Laboratory Practice Activity		
2	**Chapter 2.** Duración: 02:00 LM: Actividad del tipo Lección Magistral	**Lab 1: Kojo (RA184, CT7)** Duración: 02:00 PL: Actividad del tipo Prácticas de Laboratorio		
3	**Chapter 3. LM: Actividad tipo Lección Magistral**	**Lab 2 Scala :** **Duration: 02:00** PL: Actividad del tipo Prácticas de Laboratorio		
4	**Chapter 3.** Duración: 02:00 LM: Actividad del tipo Lección Magistral	**Lab 3 Scala** .Duración: 02:00 PL: Actividad del tipo Prácticas de Laboratorio		
5	**Chapter 4** Duración: 02:00 LM: Actividad del tipo Lección Magistral	**Lab 4 Scala** .Duración: 02:00 PL: Actividad del tipo Prácticas de Laboratorio		
6	**Chapter 4** Duración: 02:00 LM: Actividad del tipo Lección Magistral	**Lab 5. Traffic lights** Duración: : 02:00PL: Actividad del tipo Prácticas de Laboratorio		
7	**Practical exercises** Duración: 02:00 LM: Actividad del tipo Lección Magistral	**Lab 5: Semaphores** Duración: 02:00 PL: Actividad del tipo Prácticas de Laboratorio		
8	**Chapter 5.** Duración: 02:00 LM: Actividad del tipo Lección Magistral	**Lab E: Programming and validation of the questions included in the evaluation test on the computer.** Duration: 02:00 SL: Laboratory Practice Activity		

9				Written evaluation test on topics 1-4 (RA184, RA185, RA186, RA187, CC14) EX: Written Exam Technique Continuous Assessment In-person Duration: 01:30 Written evaluation test on lab practices 1-6 (RA184, RA185, RA186, RA187, CC14) EX: Written Exam
10	**Chapter 5.** Duración: 02:00 LM: Actividad del tipo Lección Magistral	**Lab 7 Monitors in jBACI (RA185, RA188)** Duration: 02:00 SL: Laboratory Practice Activity		
11	**Chapter 6.** Duración: 02:00 LM: Actividad del tipo Lección Magistral	**Lab 8 Monitors in jScala (RA185, RA188)** Duration: 02:00 SL: Laboratory Practice Activity		
12	**Chapter 6.** Duración: 02:00 LM: Actividad del tipo Lección Magistral	**Lab 9. Actors** Duration: 02:00 PL: Actividad del tipo Prácticas de Laboratorio		
13	**Chapter 8.** Duración: 02:00 LM: Actividad del tipo Lección Magistral	**Lab 10. Actors** Duration: 02:00 PL: Actividad del tipo Prácticas de Laboratorio		
14	**Chapter 9** Duración: 02:00 LM: Actividad del tipo Lección Magistral	**Lab 11 Parallel Programming** Duration: 02:00 PL: Actividad del tipo Prácticas de Laboratorio		
15	**Chapter 9** Duración: 02:00 LM: Actividad del tipo Lección Magistral	**Lab 11 Parallel Programming** Duration: 02:00 PL: Actividad del tipo Prácticas de Laboratorio		
16				Written evaluation test on topics 5-9 (All RA, CC14) EX: Written Exam Technique Continuous assessment and final exam only Face to face Duration: 01:30 Written evaluation test on lab practices 7-12 (All RA, CC14) EX: Written Exam Technique Continuous assessment and final exam only Face to face Duration: 01:30 Written evaluation test on topics 1-4 (RA184, RA185, RA186, RA187, CC14) EX: Written Exam Technique Final exam only assessment Face to face Duration: 01:30 Written evaluation test on lab practices 1-6 (RA184, RA185, RA186, RA187, CC14) EX: Written Exam Technique Final exam only assessment Face to face Duration: 01:30
17				

7. Activities and assessment criteria

7.1. Assessment activities

7.1.1. Assessment

Week	Description	Modality	Type	Duration	Weight	Minimum grade	Evaluated skills
9	Exam: chapters 1-4	Written exam	Face to face	01:30	25%	3/10	RA184, RA185. RA186, RA187, CC14
9	Exam: laboratorys 1-6	Written exam	Face to face	01:30	25%	3/10	RA184, RA185. RA186, RA187. CC14
16	Exam : chapters 5-9	Written exam	Face-to-face	01:30	25%	4 / 10	CC14
16	Exam: laboratorys 7-12	Written exam	Face-to-face	01:30	25%	4/ 10	CC14

7.1.2. Global examination

Week	Description	Modality	Type	Duration	Weight	Minimum grade	Evaluated skills
9	Exam: chapters 1-4	Written exam	Face to face	01:30	25%	3/10	RA184, RA185. RA186, RA187, CC14
9	Exam: laboratorys 1-6	Written exam	Face to face	01:30	25%	3/10	RA184, RA185. RA186, RA187. CC14
16	Exam : chapters 5-9	Written exam	Face-to-face	01:30	25%	4 / 10	CC14
16	Exam: laboratorys 7-12	Written exam	Face-to-face	01:30	25%	4/ 10	CC14

7.1.3. Referred (re-sit) examination

Description	Modality	Type	Duration	Weight	Minimum grade	Evaluated skills
Global theory exam (	Written exam	Face to face	01:30	50%	4/ 10	CC14
Comprehensive global practical exam	Practical exam	Face to face l	1:30	50%	4 / 10	CC14

7.2. Assessment criteria

Let be defined:
T1: theory exam of the first midterm, taken in week 8 or 9.
P1: practice test of the first part-time exam, taken in week 8 or 9.
T2: theory test of the second part-time exam, taken in week 16
P2: practice exam of the second midterm, taken in week 16.
In accordance with the assessment regulations, in week 16 there will be another exam to cover the theoretical and practical contents of the first Theoretical and practical contents of the first mid-term exam. These exams are open to any student who wishes to take them. The highest grade obtained will be maintained. Therefore,

T1?: theory exam of the first midterm, taken in week 16.
P1?: practical exam of the first midterm, taken in week 16.
The function max is defined as:

max (a, b: REAL): REAL; returns the largest value of both parameters a and b
and we define:

```
NoteTeoriaPrimerParcial:= max (T1, T1?)
PracticalPracticalFirstPreliminary:= max (P1, P1?)
```

The course calculates the half-sum of the grades obtained in the theoretical part and the half-sum of the grades obtained in the practical part in both mid-term exams. In order to make the averages, it is a prerequisite a grade of at least 3 in both parts of the first part-time exam and a grade of at least 4 in both papers of the second part-time exam. 4 in both parts of the second part-semester, i.e:

```
compensableNT:= FALSE
IF (TheoryFirstMasterTest >= 3) AND (T2 >= 4)
THEN TheoryTest:= (TheoryTestFirstPreliminaryTest + T2) / 2
 compensableNT:= TRUE
compensableNP:= FALSE
IF (PracticalFirstPreliminaryNote >= 3) AND (P2 >= 4)
THEN PracticeScore:= (FirstPracticeScore + P2) / 2
 compensableNP:= TRUE
```

The total mark is calculated as the half-sum of the theory mark and the practical mark, assuming that both are compensable. We define the function min as:

min (a, b: REAL): REAL; returns the smallest value of both parameters a and b.

```
IF compensableNT AND compensableNP
THEN TotalNote:= (TheoryNote+PracticeNote)/2
compensableNP:= TRUE
```

The total mark is calculated as the half-sum of the theory mark and the practical mark, assuming that both are compensable. We define the function min as:
min (a, b: REAL): REAL; returns the smallest value of both parameters a and b.

```
IF compensableNT AND compensableNP
THEN TotalNote:= (TheoryNote+PracticeNote)/2
```

ELSE notaTotal:= min (4.5, ((NotaTeoría+NotaPractica)/2)) CT:= 0,5 IF notaTotal >= 5 THEN notaFinal:= min (10, (notaTotal+CT)) ELSE notaFinal:= notaTotal

/* TC: Transversal Competence: passing the course means passing this TC and adds 0.5 points to the grade *//. points to the grade */

8. Teaching resources

8.1. Teaching resources for the subject

Name	Type	Notes
Moodle UPM	Web resource	The whole pack of documentation and examples used in class by the teacher. It is documentation elaborated by the teacher
Principles of Concurrent and Distributed Programming (2nd. Ed.). Ben-Ari (2006); Addison-Wesley.	Bibliography	Basic book for reference.
Programming in Scala (3rd. Ed.). Odersky, Spoon, Venners, 2016; Ed. Artima..	Bibliography	Basic book for reference.
Learning concurrent programming in Scala, second editionz. Prokopec, 2017; Ed. PACKT	Bibliography	Complementary book for the practical concepts.
The art of multiprocessor programming. Herlihy, Shavitt, 2012; Ed. Morgan-Kaufmann.	Bibliography	Complementary book.
Hagit Attiya and Jennifer Welch. 2004. Distributed Computing:Fundamentals, Simulations and Advanced Topics. John Wiley & Sons.	Bibliography	Complementary book.
Concurrent programming. Raynal, 2013; Ed. Springer.	Bibliografía	Complementary book.
Practical laboratory	Equipment	Building 4. Laboratory to be defined

9. Other information

COMMON COMPETENCES

This subject includes the following common competences which are not available for selection:

- CC11 Knowledge and application of the characteristics, functionalities and structure of Distributed Systems,
- Computer Networks and the Internet and to design and implement applications based on them.
- CC14 Knowledge and application of the fundamental principles and basic techniques of parallel, concurrent, distributed and time-controlled programming.

TRANSVERSAL COMPETENCES

This subject evaluates the transversal competence CT06 "Critical Thinking" according to the evaluation with this competence does not appear. Therefore, a new learning outcome has been added.

RA64. Makes judgements and decisions in a reasoned manner. Analyses, interprets and evaluates information and arguments from different points of view. Synthesises and relates information and draws conclusions in a reasoned way.

ACTIVITIES OF THE TRANSVERSAL COMPETENCE

The critical thinking activity consists of identifying the items that measure the quality of a scientific text. The activity is carried out in each laboratory session. In the first session, as an example, the students will analysis of a simple article, providing them with a rubric to be filled in with the items to be identified from the text. This is corrected in class. In the following session, a more complex text is provided and analysed during the rest of the lab sessions in order to design a rubric with the critical thinking items sought. This rubric is corrected in the last session.