COMP90048 Declarative Programming

Credit Points:	12.5
Level:	9 (Graduate/Postgraduate)
Dates & Locations:	2015, Parkville This subject commences in the following study period/s: Semester 2, Parkville - Taught on campus.
Time Commitment:	Contact Hours: 36 hours, comprising of two 1-hour lectures and one 1-hour workshop per week Total Time Commitment: 200 hours
Prerequisites:	**One of the following:** (see table below)
Corequisites:	None
Recommended Background Knowledge:	(see table below)
Non Allowed Subjects:	(see table below)

Prerequisites:

One of the following:

Subject	Study Period Commencement:	Credit Points:
COMP20007 Design of Algorithms	Semester 1	12.50
COMP90038 Algorithms and Complexity	Semester 1, Semester 2	12.50
COMP20003 Algorithms and Data Structures	Semester 2	12.50

Corequisites: None

Recommended Background Knowledge:

Subject	Study Period Commencement:	Credit Points:
SWEN20003 Object Oriented Software Development	Semester 2	12.50

Non Allowed Subjects:

Subject	Study Period Commencement:	Credit Points:
COMP30020 Declarative Programming	Semester 2	12.50

Core Participation Requirements:

<p>For the purposes of considering request for Reasonable Adjustments under the Disability Standards for Education (Cwth 2005), and Student Support and Engagement Policy, academic requirements for this subject are articulated in the Subject Overview, Learning Outcomes, Assessment and Generic Skills sections of this entry.</p> <p>It is University policy to take all reasonable steps to minimise the impact of disability upon academic study, and reasonable adjustments will be made to enhance a student's participation in the University's programs. Students who feel their disability may impact on meeting the requirements of this subject are encouraged to discuss this matter with a Faculty Student Adviser and Student Equity and Disability Support: <a href="http://services.unimelb.edu.au/disability">http://services.unimelb.edu.au/disability</a></p>

Coordinator: Dr Peter Schachte

Contact: email: **schachte@unimelb.edu.au (mailto:lee@unimelb.edu.au)**

Subject Overview:

AIMS

Declarative programming languages provide elegant and powerful programming paradigms which every programmer should know. This subject presents declarative programming languages and techniques.

INDICATIVE CONTENT

# The dangers of destructive update
# Functional programming

	# Recursion # Strong type systems # Parametric polymorphism # Algebraic types # Type classes # Defensive programming practice # Higher order programming # Currying and partial application # Lazy evaluation # Monads # Logic programming # Unification and resolution # Nondeterminism, search, and backtracking.
Learning Outcomes:	**INTENDED LEARNING OUTCOMES (ILO)** On completion of this subject the student is expected to: 1 Apply declarative programming techniques 2 Write medium size programs in a declarative language 3 Write programs in which different components use different languages 4 Select appropriate languages for each component task in a project
Assessment:	Project work during semester, requiring approximately 50-55 hours of work (40%) A mid-semester test (10%) One 2-hour end-of-semester examination (50%). Hurdle requirement: To pass the subject, students must obtain at least: 50% overall 20/40 in project work And 30/60 in the mid-semester test and end-of-semester written examination combined. Intended Learning Outcome (ILO) 1 is covered by all three assessment components, and ILO 2 is covered by the project work. ILO 3 and 4 are substantially less important, and are covered in lecture, but not explicitly assessed.
Prescribed Texts:	None
Breadth Options:	This subject is not available as a breadth subject.
Fees Information:	Subject EFTSL, Level, Discipline & Census Date, http://enrolment.unimelb.edu.au/fees
Generic Skills:	On completion of this subject, the student should have the following skills: # Ability to undertake problem identification, formulation and solution # Ability to utilise a systems approach to design and operational performance # Intellectual curiosity and creativity, including understanding of the philosophical and methodological bases of research activity # Openness to new ideas and unconventional critiques of received wisdom # Capacity for independent critical thought, rational inquiry and self-directed learning.
Notes:	**LEARNING AND TEACHING METHODS** This subject comprises 24 one-hour lectures plus 11 one-hour workshops combining group discussion and individual and small group programming work. Additionally, students develop two medium-size declarative programs for assessment. **INDICATIVE KEY LEARNING RESOURCES** At the beginning of the semester, the coordinator will propose a textbook on declarative programming, which will be made available through University Book Shop and library. The current suggested textbook is Bryan O'Sullivan, John Goerzen & Don Stewart: Real World Haskell, O'Reilly Media. This textbook can also be read online gratis. Lecture notes for the subject are also available online.

	CAREERS / INDUSTRY LINKS Over the last few years, the mainstream software industry has become quite interest in functional programming, as it promises more robust software by altogether avoiding many classes of problem common in non-declarative languages. Skills developed in this subject complement skills taught in other subjects, better equipping students for work in software design and implementation.
Related Course(s):	Master of Information Technology Master of Philosophy - Engineering Master of Science (Computer Science) Master of Software Systems Engineering Ph.D.- Engineering
Related Majors/Minors/ Specialisations:	Approved Masters level subjects from other departments Computer Science Computer Science MIT Computing Specialisation MIT Distributed Computing Specialisation Master of Engineering (Software with Business) Master of Engineering (Software)