## Contents

1 Overview 1
   1.1 Description ................................................................. 1
   1.2 Prerequisites ............................................................... 1
   1.3 Related modules ........................................................... 1
   1.4 Study units ................................................................. 1

2 Outcomes 2
   2.1 Career ................................................................. 2
   2.2 Course ................................................................. 2

3 Plagiarism policy 6

4 Instructors 6
   4.1 Course coordinator ....................................................... 6
   4.2 Teaching assistants ..................................................... 6
   4.3 Administrative support .................................................. 6

5 Organisation 7
   5.1 Class attendance .......................................................... 7
   5.2 Module website ........................................................... 7
   5.3 Mark Administration .................................................... 7
   5.4 Announcements ........................................................... 7
   5.5 Lectures ................................................................. 8
   5.6 Practicals ................................................................. 8
   5.7 Discussion forum ......................................................... 8
   5.8 Interaction with the Instructors ...................................... 8

6 Study Material 9
   6.1 Prescribed textbooks ..................................................... 9
   6.2 Additional references ................................................... 9
   6.3 Software ................................................................. 9

7 Assessment 9
   7.1 Semester mark ............................................................ 10
   7.2 Semester Tests ............................................................ 10
   7.3 Unannounced Class Tests ............................................... 10
   7.4 Practical Lab Experiences ............................................... 10
   7.5 Examination ............................................................. 11
   7.6 Final Mark Calculation ................................................ 11

8 Lecture Schedule 12

9 Practical Lab Experience Schedule 13
1 Overview

1.1 Description

Programming languages are the backbone for software development. Each language has its own unique syntax and semantics, but there are many common concepts that can be illustrated and studied through the languages. This module concentrates on issues of object orientation, including delegation, iteration and polymorphism. It surveys how languages provide the basic building blocks for data and control, as well as exception handling and concurrency. At the end of the module, students will be able to appreciate the rich history behind programming languages, leading to independent principles that have evolved over time. They will be skilled at using a variety of programming languages (including new paradigms such as functional, logic and scripting), and will know how to learn a new language with ease. From this experience, they will be able to apply evaluation criteria for choosing an appropriate programming language in a given scenario.

1.2 Prerequisites

COS 110 (Program Design: An Introduction) must be completed before this module may be taken.

1.3 Related modules

The following modules touch on topics that are covered in COS 333:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Related Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>COS 151</td>
<td>Introduction to Computer Science</td>
<td>Machine level representation of data</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Information storage and retrieval</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Recursion</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Object-oriented programming</td>
</tr>
<tr>
<td>COS 153</td>
<td>Introduction to Programming</td>
<td>Information storage and retrieval</td>
</tr>
<tr>
<td>COS 131</td>
<td>Introduction to Programming</td>
<td>Recursion</td>
</tr>
<tr>
<td>COS 132</td>
<td>Imperative Programming</td>
<td></td>
</tr>
<tr>
<td>COS 110</td>
<td>Program Design: Introduction</td>
<td>Object-oriented programming</td>
</tr>
<tr>
<td>COS 216</td>
<td>Netcentric Computer Systems</td>
<td>Information storage and retrieval</td>
</tr>
</tbody>
</table>

1.4 Study units

This module covers the main themes of programming languages through the following study units, taken from the ABET Computing Accreditation Commission guidelines:

<table>
<thead>
<tr>
<th>Knowledge Area</th>
<th>Study Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social and Professional Issues</td>
<td>History of computing</td>
</tr>
<tr>
<td>Architecture and Organisation</td>
<td>Machine level representation of data</td>
</tr>
<tr>
<td>Programming Fundamentals</td>
<td>Fundamental programming constructs</td>
</tr>
<tr>
<td></td>
<td>Fundamental data structures</td>
</tr>
<tr>
<td></td>
<td>Event-driven programming</td>
</tr>
<tr>
<td>Operating Systems</td>
<td>Scripting</td>
</tr>
</tbody>
</table>
2 Outcomes

2.1 Career

The aim of the course is to equip students with a broad knowledge of programming languages. This will equip students for any ICT career where new languages are used or developed on a continuous basis.

2.2 Course

The outcomes of the study units covered by this course, taken from the ABET Computing Accreditation Commission guidelines, are as follows:

History of computing

- List the contributions of several pioneers in the computing field.
- Identify significant continuing trends in the history of the computing field.

Machine level representation of data

- Explain the reason for using different formats to represent numerical data.
- Describe the internal representation of nonnumeric data.
- Describe the internal representation of characters, strings, records, and arrays.

Fundamental programming constructs

- Design, implement, test, and debug a program that uses each of the following fundamental programming constructs: basic computation, simple I/O, standard conditional and iterative structures, and the definition of functions.
- Choose appropriate conditional and iteration constructs for a given programming task.
- Describe the mechanics of parameter passing.
Fundamental data structures

- Discuss the representation and use of primitive data types and built-in data structures.
- Describe how various data structures are allocated and used in memory.
- Describe common applications for various data structures.
- Implement the user-defined data structures in various high-level languages.
- Compare alternative implementations of data structures with respect to performance.
- Write programs that use each of the following data structures: arrays, records, strings, linked lists, stacks, queues, and hash tables.
- Compare and contrast the cost and benefits of dynamic and static data structure implementations.
- Choose the appropriate data structure for modelling a given problem.

Event-driven programming

- Explain the difference between event-driven programming and command-line programming.
- Design, code, test, and debug simple event-driven programs that respond to user events.
- Develop code that responds to exception conditions raised during execution.

Scripting

- Demonstrate the typical functionality of a scripting language, and interpret the implications for programming.
- Demonstrate the mechanism for implementing scripts and the role of scripts on system implementation and integration.
- Implement a simple script that exhibits parameter passing.

Overview of programming languages

- Summarise the evolution of programming languages, illustrating how this history has led to the paradigms available today.
- Identify distinguishing characteristics for each of the programming paradigms covered in this course.
- Evaluate the tradeoffs between the different paradigms, considering such issues as space efficiency, time efficiency (of both the computer and the programmer), safety, and power of expression.
- Distinguish between programming-in-the-small and programming-in-the-large.
Virtual machines

- Describe the importance and power of abstraction in the context of virtual machines.
- Explain the benefits of intermediate languages in the compilation process.
- Evaluate the tradeoffs in performance vs. portability.

Introduction to language translation

- Compare the contrast compiled and interpreted execution models, outlining the relative merits of each.
- Describe the phases of program translation from source code to executable code and the file produced by these phases.
- Explain the difference between machine-dependent and machine independent translation and where these differences are evident in the translation process.

Declarations and types

- Explain the value of declaration models, especially with respect to programming-in-the-large.
- Identify and describe the properties of a variable such as its associated address, value, scope, persistence, and size.
- Discuss type incompatibility.
- Demonstrate different forms of binding, visibility, scoping, and lifetime management.

Abstraction mechanisms

- Explain how abstraction mechanism supports the creation of reusable software components.
- Demonstrate the difference between call-by-value and call-by-reference parameter passing.
- Defend the importance of abstractions, especially with respect to programming-in-the-large.

Object-oriented programming

- Justify the philosophy of object-oriented design and the concept of encapsulation, abstraction, inheritance, and polymorphism.
- Design, implement, test, and debug simple programs in various object-oriented programming languages.
- Describe how the class mechanism supports encapsulation and information hiding.
- Design, implement, and test the implementation of “is-a” relationships among objects using a class hierarchy and inheritance.
• Compare and contest the notions of overloading and overriding methods in an object-oriented language.

**Functional programming**

• Outline the strengths and weaknesses of the functional programming paradigm.
• Design, code, test, and debug programs using the functional paradigm.
• Explain the use of functions as data, including the concept of closures.

**Language translation systems**

• Describe the steps and algorithms used by language translators.
• Discuss the effectiveness of optimisation.
• Explain the impact of a separate compilation facility and the existence of program libraries on the compilation process.

**Type systems**

• Formalise the notion of typing.
• Describe each of the elementary data types.
• Explain the concept of an abstract data type.
• Recognise the importance of typing for abstraction and safety.
• Differentiate between static and dynamic typing.
• Differentiate between type declarations and type inference.
• Evaluate languages with regard to typing.

**Programming language design**

• Evaluate the impact of different typing regimes on language design, language usage, and the translation process.
• Explain the role of different abstraction mechanisms in the creation of user-defined facilities.

**Basic logic**

• Apply formal methods of symbolic propositional and predicate logic.
• Describe how formal tools of symbolic logic are used to model algorithms and real-life situation.
• Describe the importance and limitations of predicate logic.

**Knowledge representation and reasoning**

• Explain the operation of the resolution technique for theorem proving.
3 Plagiarism policy

The Department of Computer Science considers plagiarism to be a serious offence. Disciplinary action will be taken against students who commit plagiarism. Plagiarism includes copying someone else’s work without consent, copying a friend’s work (even with consent) and copying material (such as text or program code) from the Internet. Copying will not be tolerated in this module.

For a formal definition of plagiarism, the student is referred to http://www.ais.up.ac.za/plagiarism/index.htm (from the main page of the University of Pretoria site, follow the Library quick link, and then click the Plagiarism link). If you have any form of question regarding this, please the lecturer, to avoid any misunderstanding. Also note that the principle of code re-use does not mean that you should copy and adapt code to suit your solution.

Note that all assignments submitted for this module implicitly agree to this plagiarism policy, and declare that the submitted work is the student’s own work. Assignments will be submitted to a variety of plagiarism checks. Any typed assignment may be checked using the Turnitin system. After plagiarism checking, assignments will not be permanently stored on the Turnitin database.

4 Instructors

4.1 Course coordinator

The module coordinator takes primary responsibility for the module and its content:

<table>
<thead>
<tr>
<th>Name</th>
<th>Office</th>
<th>Telephone</th>
<th>E-mail</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mr Willem van Heerden</td>
<td>IT 5-38</td>
<td>(012) 420-5468</td>
<td><a href="mailto:wvheerden@cs.up.ac.za">wvheerden@cs.up.ac.za</a></td>
</tr>
</tbody>
</table>

4.2 Teaching assistants

There are three teaching assistants assigned to this course. They will assist students during practical sessions:

<table>
<thead>
<tr>
<th>Name</th>
<th>E-mail</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hlavutelo Maluleke</td>
<td><a href="mailto:hlavutelo.maluleke@gmail.com">hlavutelo.maluleke@gmail.com</a></td>
</tr>
<tr>
<td>Collins Mphahelele</td>
<td><a href="mailto:u12211070@tuks.co.za">u12211070@tuks.co.za</a></td>
</tr>
<tr>
<td>Sifiso Shabangu</td>
<td><a href="mailto:gammer4live@gmail.com">gammer4live@gmail.com</a></td>
</tr>
</tbody>
</table>

4.3 Administrative support

General degree and examination-related administrative queries should be directed to:

<table>
<thead>
<tr>
<th>Name</th>
<th>Office</th>
<th>Telephone</th>
<th>E-mail</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ms Elmarie Willemse</td>
<td>IT 4-18</td>
<td>(012) 420-2504</td>
<td><a href="mailto:ewillemse@cs.up.ac.za">ewillemse@cs.up.ac.za</a></td>
</tr>
</tbody>
</table>

The head of the Computer Science Department must be contacted via his secretary:

<table>
<thead>
<tr>
<th>Name</th>
<th>Office</th>
<th>Telephone</th>
<th>E-mail</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ms Angela Bekker</td>
<td>IT 4-17</td>
<td>(012) 420-2361</td>
<td><a href="mailto:abekker@cs.up.ac.za">abekker@cs.up.ac.za</a></td>
</tr>
</tbody>
</table>
5 Organisation

5.1 Class attendance

First and foremost, class attendance is vital to maintain a good academic record. Material not covered in the slides or textbook may also be discussed during lectures. Please ensure that you attend these forums so that you are aware of important announcements, additional discussions and material not covered in this study guide.

5.2 Module website

The CS module page for COS 333 is maintained on the web page of the Department of Computer Science. You can reach this page by visiting the department’s website at http://www.cs.up.ac.za/, clicking the Undergraduate Portal link, and finally clicking the COS333 link in the list. You can also reach the CS module page directly, by visiting http://www.cs.up.ac.za/courses/COS333/. The CS module page will be used to host most of the study material for this module, and is thus the primary point of contact for electronic information and resources. The CS module page will host the following content:

- All announcements.
- File downloads, links and notes.
- All practical specifications.
- Practical submission upload slots.
- The discussion forum.

You are advised to regularly monitor the COS 333 website for any updates and new announcements. Please note that this is the 26 July 2015 version of the course study guide, and refer to the website for the latest version.

5.3 Mark Administration

Marks for assessment opportunities will be posted electronically as soon as they are available. A mark query procedure will be published on the module site when marks are published. Follow these instructions carefully. Failure to do so may result in your query not being attended to. Mark queries must be sent within five (5) working days of their publication, and will be attended to as soon as possible after they have been received. Retain physical proof of all assessment opportunity marks (i.e., test scripts, and marked practicals) until after the module’s examination.

5.4 Announcements

Announcements will be made during class time, and via the module website. While every effort will be made to repeat announcements both in class and via the website, it is not guaranteed that this will be the case. It is therefore expected that students attend all classes, and check the module website on a daily basis.
5.5 Lectures

There are two (2) lectures per week for COS 333, as follows:

<table>
<thead>
<tr>
<th>Day</th>
<th>Time</th>
<th>Venue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monday</td>
<td>15:30–16:20</td>
<td>IT 4-3</td>
</tr>
<tr>
<td>Thursday</td>
<td>13:30–14:20</td>
<td>IT 4-3</td>
</tr>
</tbody>
</table>

5.6 Practicals

There are two (2) available practical lab experience time slots, of which each student must register for and attend one (1):

<table>
<thead>
<tr>
<th>Day</th>
<th>Time</th>
<th>Venue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monday</td>
<td>17:30–20:20</td>
<td>Brown Lab</td>
</tr>
<tr>
<td>Tuesday</td>
<td>17:30–20:20</td>
<td>SIT 1</td>
</tr>
</tbody>
</table>

5.7 Discussion forum

The discussion forum for the course is made available as a means for discussing course-related work. Please note that the forum is not to be used for non-academic purposes (e.g., advertising), and is not a forum for airing grievances related to the module. If you wish to make a complaint about something related to the course, please contact either the lecturer directly, or the class representative (class representative details will be posted on the module website). Failure to observe these rules may result in the forum being closed. Do not attempt to contact a lecturer via the forum, because your message may not be seen.

5.8 Interaction with the Instructors

You can speak with Mr van Heerden after classes. Make sure that you ask questions during the semester, and ensure your understanding of the work is up-to-date. It is not advisable to visit Mr van Heerden TAs without an appointment, because he has busy schedules and may be unavailable. If you require an appointment, please request one by means of an e-mail, and do so well in advance (not the day before an important deadline). When communicating via email, ALWAYS give the module code, your full name (including your first name and surname) and your student number. Your email address must have your full correct name displayed, not just your student number. Please be courteous and use clear language during all interactions.

Teaching assistants will be available during the practical lab experience sessions (see section 9). Teaching assistants can also be contacted via e-mail, should you require help outside of the practical sessions.

Depending on demand, times may be made available for teaching assistants to provide help outside of practical hours. Such consultation will take place in the CosTutorium, located in IT 4-22 (to the right of the reception desk of the Computer Science Department). In the event of such slots being made available, further details will be announced.
6 Study Material

6.1 Prescribed textbooks

Note that you must have access to a copy of the prescribed textbook for this course, since reference may be made to material contained in the book. Also note that the course slides are not a replacement for either the lectures or the textbook. The details of the prescribed textbook are as follows:

Title: Concepts of Programming Languages
Author: Robert W. Sebesta
Edition: Tenth Edition
ISBN: 978-0273769101

6.2 Additional references

The following additional texts will provide additional background information. Copies of these texts are available in the university library:

Title: Programming Language Pragmatics
Author: Michael L. Scott
Edition: Third Edition
ISBN: 978-0123745149

Title: Foundations of Programming Languages — Design and Implementation
Author: Seyed H. Roosta
ISBN: 978-0534393038

During the module, the lecturer may prescribe additional notes or articles. It is the students’ responsibility to ensure that they obtain a copy. Note that such material is examinable, unless stated otherwise. Guest lectures may also be presented during class times, and all material related to these lectures will also be examinable, unless stated otherwise. Please make sure that you obtain copies of all such material.

6.3 Software

All of the Software you are required to familiarise yourself with will be available under Linux in the Informatorium. Note that all practical lab experience submissions and demonstrations work according to the specifications. This typically means that your programs must compile and run using the software versions installed in the labs. Some programming language software may be made available on ftp://ftp.cs.up.ac.za. References to external sources, such as websites from which software must be downloaded, may also be provided. The prescribed text also lists several websites, where some of the software used in this course can be obtained.

7 Assessment

The semester mark consists of the following assessment opportunities: Two (2) semester tests, seven (7) practical lab experiences, and four (4) unannounced class tests. Each opportunity is discussed separately, below:
7.1 Semester mark

The semester mark for the module will be calculated as follows:

Semester Mark = Semester tests (50%) + Class tests (20%) + Practical experiences (30%)

7.2 Semester Tests

Both semester tests will be written tests. The scope of each will be announced closer to the date of the relevant test. Please make sure to check the course website closer to the dates of the semester tests, for any updates. The semester test details are as follows:

Semester Test 1

Date: Tuesday 22 September 2015
Time: 17:30–20:00
Venue: IT 4-1

Semester Test 2

Date: Tuesday 20 October 2015
Time: 17:30–19:30
Venue: Centenary 5

The average of the two semester test marks will count 50% of the semester mark. You are informed here, well in advance, to make appropriate arrangements with regard to clearing your schedule for these tests. Please note that it is required that students write both tests. You will not be permitted to write the examination if you fail to write both papers.

The only valid excuse for missing a semester test is due to medical reasons. In such a case, a medical certificate from a registered medical professional must be handed in at the reception desk for Mr van Heerden’s attention within three (3) working days of the test. Under exceptional circumstances (e.g., the death of a family member), a police affidavit will also serve as valid documentation for a missed test. An aegrotat test will be arranged for students submitting the appropriate documentation after missing a semester test.

7.3 Unannounced Class Tests

Four (4) unannounced class tests will be given throughout the semester, of which a minimum of two (2) must be completed to obtain examination entrance. These tests will cover any of the work covered in the lectures prior to the date of the test. The tests will count equally towards a contribution of 20% to the semester mark.

7.4 Practical Lab Experiences

During the semester there will be seven (7) practical lab experiences, of which a minimum of five (5) must be completed to obtain examination entrance. You will have to perform a practical task approximately every two weeks (see section 9 for the practical schedule). Marks will be allocated either by means of a demonstration during a practical session, or via an off-line mark that will be awarded to electronically submitted work. However the practical is assessed, each set of practical tasks must be uploaded to an assignment slot on the
course website, for the mark to be awarded. Strictly no late submissions will be accepted. Each practical will count equally towards a 30% contribution to the semester mark.

7.5 Examination

All the work covered during the semester will be examinable. Please make sure to check the course website towards the end of the semester, for any updates. The examination details are as follows:

Date: Thursday 26 November 2015
Time: 08:00–11:00
Venue: To be announced
Scope: Chapters 1–16 (excluding Chapters 3, 4, 10 and 13) of the prescribed text

7.6 Final Mark Calculation

Each student’s final mark for COS 333 will be calculated as follows:

\[
\text{Final Mark} = \text{Examination mark (50\%)} + \text{Semester Mark (50\%)}
\]

In addition, the following subject-related regulations hold for this course:

- A student will be refused examination entrance if his/her semester mark is less than 40%.
- A student will be refused examination entrance if he/she fails to complete the required number of semester tests (two), class tests (four), or practical lab experiences (five).
- To pass the course, a student must obtain at least 40% in the exam, and a final mark of at least 50%.
- A student will pass the course with distinction if he/she obtains at least 75% for the final mark.

Please also take note of the examination rules, as provided for in the yearbook for the School of Information Technology, under section IT.12 (available electronically at http://www.up.ac.za/media/shared/360/Year%20Books%202015/EBIT/ebit-part-3-it-2015.zp42451.pdf). Note that no supplementary examination will take place for COS 333.
# Lecture Schedule

The following lecture schedule (which is subject to change) will be followed:

<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Topic</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Mon 20 Jul</td>
<td>Introduction</td>
<td>Chapter 1</td>
</tr>
<tr>
<td></td>
<td>Thur 23 Jul</td>
<td>Preliminaries</td>
<td>Chapter 1</td>
</tr>
<tr>
<td>2</td>
<td>Mon 27 Jul</td>
<td>Evolution of the Major Programming Languages</td>
<td>Chapter 2</td>
</tr>
<tr>
<td></td>
<td>Thur 30 Jul</td>
<td>Evolution of the Major Programming Languages</td>
<td>Chapter 2</td>
</tr>
<tr>
<td>3</td>
<td>Mon 3 Aug</td>
<td>Evolution of the Major Programming Languages</td>
<td>Chapter 2</td>
</tr>
<tr>
<td></td>
<td>Thur 6 Aug</td>
<td>Names, Bindings, and Scopes</td>
<td>Chapter 5</td>
</tr>
<tr>
<td>4</td>
<td>Mon 10 Aug</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>Thur 13 Aug</td>
<td>Names, Bindings, and Scopes</td>
<td>Chapter 5</td>
</tr>
<tr>
<td>5</td>
<td>Mon 17 Aug</td>
<td>Data Types</td>
<td>Chapter 6</td>
</tr>
<tr>
<td></td>
<td>Thur 20 Aug</td>
<td>Data Types</td>
<td>Chapter 6</td>
</tr>
<tr>
<td>6</td>
<td>Mon 24 Aug</td>
<td>Expressions and Assignment Statements</td>
<td>Chapter 7</td>
</tr>
<tr>
<td></td>
<td>Thur 27 Aug</td>
<td>Expressions and Assignment Statements</td>
<td>Chapter 7</td>
</tr>
<tr>
<td>7</td>
<td>Mon 31 Aug</td>
<td>Statement-Level Control Structures</td>
<td>Chapter 8</td>
</tr>
<tr>
<td></td>
<td>Thur 3 Sep</td>
<td>Statement-Level Control Structures</td>
<td>Chapter 8</td>
</tr>
<tr>
<td>8</td>
<td>Mon 7 Sep</td>
<td>Subprograms</td>
<td>Chapter 9</td>
</tr>
<tr>
<td></td>
<td>Thur 10 Sep</td>
<td>Subprograms</td>
<td>Chapter 9</td>
</tr>
<tr>
<td>9</td>
<td>Mon 14 Sep</td>
<td>Abstract Data Types and Encapsulated Constructs</td>
<td>Chapter 11</td>
</tr>
<tr>
<td></td>
<td>Thur 17 Sep</td>
<td>Abstract Data Types and Encapsulated Constructs</td>
<td>Chapter 11</td>
</tr>
<tr>
<td>10</td>
<td>Mon 21 Sep</td>
<td>Support for Object-Oriented Programming</td>
<td>Chapter 12</td>
</tr>
<tr>
<td></td>
<td>Tue 22 Sep</td>
<td>Semester Test 1</td>
<td>Chapter 12</td>
</tr>
<tr>
<td></td>
<td>Thur 24 Sep</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>11</td>
<td>Mon 28 Sep</td>
<td>Support for Object-Oriented Programming</td>
<td>Chapter 12</td>
</tr>
<tr>
<td></td>
<td>Thur 1 Oct</td>
<td>Support for Object-Oriented Programming</td>
<td>Chapter 12</td>
</tr>
<tr>
<td></td>
<td>Mon 5 Oct</td>
<td>Recess</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>Thur 8 Oct</td>
<td>Recess</td>
<td>—</td>
</tr>
<tr>
<td>12</td>
<td>Mon 12 Oct</td>
<td>Exception Handling and Event Handling</td>
<td>Chapter 14</td>
</tr>
<tr>
<td></td>
<td>Thur 15 Oct</td>
<td>Functional Programming Languages</td>
<td>Chapter 15</td>
</tr>
<tr>
<td>13</td>
<td>Mon 19 Oct</td>
<td>Functional Programming Languages</td>
<td>Chapter 15</td>
</tr>
<tr>
<td></td>
<td>Tue 20 Oct</td>
<td>Semester Test 2</td>
<td>Chapter 15</td>
</tr>
<tr>
<td></td>
<td>Thur 22 Oct</td>
<td>Functional Programming Languages</td>
<td>Chapter 15</td>
</tr>
<tr>
<td>14</td>
<td>Mon 26 Oct</td>
<td>Logic Programming Languages</td>
<td>Chapter 16</td>
</tr>
<tr>
<td></td>
<td>Thur 29 Oct</td>
<td>Logic Programming Languages</td>
<td>Chapter 16</td>
</tr>
<tr>
<td>15</td>
<td>Mon 2 Nov</td>
<td>Logic Programming Languages</td>
<td>Chapter 16</td>
</tr>
<tr>
<td></td>
<td>Thur 5 Nov</td>
<td>Examination Preparation, and Questions</td>
<td>—</td>
</tr>
</tbody>
</table>

† Monday 10 August: Public holiday in terms of legislation.

‡ Monday 21 September: Heritage Day
# Practical Lab Experience Schedule

The following table lists the practical lab experience sessions for the course. During practical weeks marked with an asterisk (*), students will have to demonstrate their previous practical for marking purposes. The new practical lab experience will also be made available during the session. It is therefore compulsory for students to attend at least all of the practical lab experience sessions marked with an asterisk. It is recommended that students arrive earlier in the session, rather than later, to avoid marking bottlenecks.

In sessions that are not marked with an asterisk, teaching assistants will be available in the labs to provide assistance. If additional help is required beyond the practical sessions, students should e-mail the teaching assistants for the course directly (refer to section 4.2 for contact details). The following schedule (which is subject to change) will be followed:

<table>
<thead>
<tr>
<th>Week</th>
<th>Dates</th>
<th>New practical available</th>
<th>Assessment or submission</th>
</tr>
</thead>
<tbody>
<tr>
<td>* 2</td>
<td>Mon 27 Jul</td>
<td>Tue 28 Jul</td>
<td>Research assignment</td>
</tr>
<tr>
<td>* 3</td>
<td>Mon 3 Aug</td>
<td>Tue 4 Aug</td>
<td>Imperative 1</td>
</tr>
<tr>
<td>4</td>
<td>Mon 10 Aug</td>
<td>Tue 11 Aug</td>
<td>No practical †</td>
</tr>
<tr>
<td>* 5</td>
<td>Mon 17 Aug</td>
<td>Tue 18 Aug</td>
<td>Imperative 2</td>
</tr>
<tr>
<td>6</td>
<td>Mon 24 Aug</td>
<td>Tue 25 Aug</td>
<td></td>
</tr>
<tr>
<td>* 7</td>
<td>Mon 31 Aug</td>
<td>Tue 1 Sep</td>
<td>Imperative 3</td>
</tr>
<tr>
<td>8</td>
<td>Mon 7 Sep</td>
<td>Tue 8 Sep</td>
<td></td>
</tr>
<tr>
<td>* 9</td>
<td>Mon 14 Sep</td>
<td>Tue 15 Sep</td>
<td>Imperative 4</td>
</tr>
<tr>
<td>10</td>
<td>Mon 21 Sep</td>
<td>Tue 22 Sep</td>
<td>No practical **</td>
</tr>
<tr>
<td>* 11</td>
<td>Mon 28 Sep</td>
<td>Tue 29 Sep</td>
<td>Functional</td>
</tr>
<tr>
<td></td>
<td>Mon 5 Oct</td>
<td>Tue 6 Oct</td>
<td>Recess</td>
</tr>
<tr>
<td>12</td>
<td>Mon 12 Oct</td>
<td>Tue 13 Oct</td>
<td></td>
</tr>
<tr>
<td>* 13</td>
<td>Mon 19 Oct</td>
<td>Tue 20 Oct</td>
<td>Logic</td>
</tr>
<tr>
<td>14</td>
<td>Mon 26 Oct</td>
<td>Tue 27 Oct</td>
<td></td>
</tr>
<tr>
<td>* 15</td>
<td>Mon 2 Nov</td>
<td>Tue 3 Nov</td>
<td>Logic</td>
</tr>
</tbody>
</table>

† Monday 10 August: National Women’s Day.
‡ Electronic-only submission of research assignment on Monday 10 August.
** Tuesday 22 September: Friday timetable followed.