

# **PROPOSAL FOR A REVISION OF THE B.Sc. COMPUTER SCIENCE PROGRAMMES**

## **1.0 Introduction**

The Department of Computing and Information Technology is proposing an upgrade to its revised B.Sc. Computer Science degrees which came into effect in September 2012. The main purpose of the upgrade is to convert all the advanced courses of the Computer Science programmes from four credits to three credits to be consistent with the other departments in the Faculty of Science and Technology. The revision of the B.Sc. Computer Science degrees in 2012 only converted Level 1 courses from 6 credits to 3 credits. This current effort is a continuation of the work started in the 2012 upgrade.

The opportunity was also taken to review the curriculum so that the programmes would be even more closely aligned to the international Association for Computing Machinery / Institute for Electrical and Electronics Engineers (ACM/IEEE) Curriculum Guidelines for Computer Science. This is an important step towards international accreditation of its programmes which the department is seeking to obtain in the near future. At the same time, the curriculum was reviewed to take into consideration the needs of the business community and other stakeholders, as well as the strategic goals of the UWI with the view of creating a distinctive UWI graduate.

A near-final draft of the proposal was presented at the Quality Assurance Review of the department which took place in April, 2015. In their final report, the Review Team noted a great deal of undue strain in the department due to the teaching of two separate degree programmes (Computer Science and Information Technology) without shared courses. This severely affects the teaching loads of the full-time and part-time academic staff in the department, some of whom include graduate students. The Review Team recommended that the department should address the problem by (1) creating courses that can be shared by both the Computer Science and Information Technology degrees; and (2) reducing the number of elective offerings. These recommendations have been adopted in a significant way in the current proposal. Traditionally, applicants to any of the B.Sc. Computer Science programmes were required to have two CAPE subjects (or GCE 'A' Levels or equivalent) one of which must be Mathematics. This narrowed the pool of potential applicants to the Computer Science programmes. In this proposal, the Mathematics requirement is removed and students' mathematical background is strengthened in the first year of the programmes. In addition, applicants who hold passes in CAPE Economics or Accounting (or GCE 'A' Level equivalent) will also be considered. These changes in the admission requirements are intended to widen the pool of potential applicants to the B.Sc. Computer Science programmes.

## **1.1 Aims of the Computer Science Programmes**

Computer Science is an enormously vibrant field. From its inception just half a century ago, computing has become the defining technology of our age. Computing is integral to modern culture and is the primary engine behind much of the world's economic growth. The field, moreover, continues to evolve at an astonishing pace. New technologies are introduced continually, and existing ones become obsolete almost as soon as they appear. The rapid evolution of the discipline has a profound effect on computer science education, affecting both content and pedagogy.

## 1.2 Programme Learning Outcomes (Goals)

The B.Sc. Computer Science programmes aim to provide Computer Science graduates with the skills and knowledge to take on appropriate professional positions in computing upon graduation and to grow into leadership positions or pursue research or graduate studies in the field. On successful completion of the programme, graduates will be able to:

1. Design and implement systems and software based upon critical review of developing trends, approaches and research.
2. Devise new and innovative ways to use computers in other disciplines such as in science (e-science) and the arts.
3. Solve computing problems using algorithms and multiple programming languages.
4. Demonstrate inter-personal skills, teamwork, and efficient use of appropriate technology.
5. Develop a professional development framework for maintaining expertise in the field of computer science.
6. Adapt to new work-place challenges and environments.
7. Apply computer science theories and methods in the design and analysis of systems producing a solution to a variety of problems across multiple disciplines.
8. Manage processes, systems, and human resources in alignment with an organization's structure and/or needs.
9. Conduct research necessary to successfully complete a capstone project.
10. Design a system, component or process that illustrates the interplay between theory and practice.
11. Demonstrate effective written and oral communication techniques when completing reports and presentations.

## 1.3 Rationale for the Programmes

The B.Sc. programmes in Computer Science have been offered for many years. The most recent revision took place in 2010 when the B.Sc. Computer Science (Special) was offered for the first time. Many of the graduates of our Computer Science programmes are now employed in leadership positions in Trinidad and Tobago and elsewhere. From 1995-2005, enrollment of new students in the Computer Science programmes was about 175.

The B.Sc. programme in Information Technology has been offered since 2006 with an enrollment of 40 students and has steadily grown to about 100. With the introduction of the Information Technology programme, enrollment in the Computer Science programmes dropped as more and more applicants were accepted into the B.Sc. Information Technology programmes. The enrollment in the B.Sc. Computer Science programmes in 2014 was about 40 but climbed again to about 70 in 2015 due to the Mathematics requirement being removed via a Faculty Board paper which was approved by the Board for Undergraduate Studies.

The Computer Science programmes have traditionally been developed around the Curriculum Guidelines for Computer Science produced by the international Association for Computing Machinery / Institute for Electrical and Electronics Engineers (ACM/IEEE). Because of this, graduates in Computer Science can take up employment or pursue postgraduate studies in any part of the world. Further, our graduates are

also highly sought after locally in comparison to graduates from other tertiary-level institutions operating in Trinidad and Tobago.

A key objective of this proposal is to convert all the advanced courses from 4-credit to 3-credit. In 2012, all the first year courses in the Computer Science programmes were converted from 6-credit to 3-credit. However, based on recommendations from the Quality Assurance Review Team in 2015, we have taken the opportunity to re-engineer the B.Sc. Computer Science programmes so that there will be sharing of courses with the B.Sc. Information Technology programmes, thereby improving resource utilization. The Computer Science programmes have also been re-engineered to facilitate the new entry requirements in the hope of attracting a wider pool of students into the programmes.

Since the department is considering obtaining international accreditation of its programmes, the curriculum review exercise also re-examined the ACM/IEEE Curriculum Guidelines to ensure that the revised programmes were aligned as closely as possible to the guidelines. The Quality Assurance Review Team commented in 2015 that because of the close alignment of our programmes with the ACM/IEEE Curriculum Guidelines, it felt that obtaining accreditation would be considerably simplified.

## **2.0 Comparison with Existing Programmes**

In this section, we describe the broad structure of the existing programmes in Computer Science and explain the changes that were made to produce the structure of the revised degree programmes in Computer Science.

### **2.1 Existing Programmes**

Three variations of the Computer Science degree are currently offered:

#### **1. B.Sc. Computer Science (Special) [93 credits]**

- 24 credits of Level 1 core courses in Computer Science
- 52 credits of advanced core courses in Computer Science
- 8 credits of advanced elective courses in Computer Science or Information Technology
- 9 credits of Foundation courses

#### **2. B.Sc. Computer Science and Management [99 credits]**

- 18 credits of Level 1 core courses in Computer Science
- 12 credits of Level 1 core courses in Accounting and Economics
- 32 credits of advanced core courses in Computer Science
- 15 credits of advanced core courses in Management
- 13 credits of advanced core courses in Computer Science, Information Technology, Mathematics, Economics, or Management
- 9 credits of Foundation courses

#### **3. B.Sc. General (Major in Computer Science) [93 credits]**

- 12 credits of Level 1 core courses in Computer Science
- 12 credits of other Level 1 courses
- 24 credits of advanced core courses in Computer Science
- 8 credits of advanced elective courses in Computer Science
- 28 credits of other courses
- 9 credits of Foundation courses

In addition, a Minor in Computer Science is offered as follows:

- 12 credits of Level 1 core courses in Computer Science
- 8 credits of advanced core courses in Computer Science

8 credits of advanced elective courses in Computer Science

The Minor in Computer Science is available to any student in the Faculty of Science or Technology or other Faculties of the University of the West Indies.

## **2.2 Major Changes in the Programmes**

This proposal builds on the strengths of the existing programmes in Computer Science while making more efficient use of resources through the sharing of courses with the Information Technology programmes.

Two important changes to the existing programmes in Computer Science are as follows:

1. All the advanced courses are being changed from 4-credit to 3-credit. This requires changes to the composition of the existing degree programmes. Consequently, there is a new structure for the B.Sc. Computer Science degrees.
2. There is now a significant amount of sharing of courses between the B.Sc. Computer Science degrees and the B.Sc. Information Technology degrees. Indeed, the Level 1 courses in the B.Sc. Computer Science (Special) and the B.Sc. Information Technology are identical.

## **2.3 Structure of New Programmes**

The three variations of the Computer Science degree will now be structured as follows:

### **1. B.Sc. Computer Science (Special) [93 credits]**

24 credits of Level 1 core courses in Computer Science / Information Technology / Mathematics  
30 credits of Level 2 core courses in Computer Science / Information Technology / Mathematics  
15 credits of Level 3 core courses in Computer Science  
15 credits of advanced elective courses in Computer Science / Information Technology  
9 credits of Foundation courses

### **2. B.Sc. Computer Science with Management [93 credits]**

12 credits of Level 1 core courses in Computer Science / Information Technology / Mathematics  
12 credits of Level 1 core courses in Accounting and Economics  
18 credits of Level 2 core courses in Computer Science  
15 credits of advanced core courses in Management  
12 credits of advanced elective courses in Computer Science / Information Technology  
15 credits of additional advanced elective courses in Computer Science, Economics, Information Technology, Mathematics, or Management  
9 credits of Foundation courses

### **3. B.Sc. General (Major in Computer Science) [93 credits]**

12 credits of Level 1 core courses in Computer Science  
12 credits of other Level 1 courses  
18 credits of Level 2 core courses in Computer Science  
12 credits of advanced elective courses in Computer Science / Information Technology  
30 credits of other advanced courses  
9 credits of Foundation courses

In addition, the Minor in Computer Science will now comprise the following:

12 credits of Level 1 core courses in Computer Science  
9 credits of Level 2 core courses in Computer Science

### 3.0 Description of Programmes

#### 3.1 Structure of Degree Programmes

##### B.Sc. Computer Science (Special)

Course Code	Course Title	Semester	Credits	Prerequisite(s)
<b>Level 1</b>	<b>Introductory Core Courses (24 credits)</b>			
COMP 1600	Introduction to Computing Concepts	1	3	None
COMP 1601	Computer Programming I	1	3	None
MATH 1115	Fundamental Mathematics for the General Sciences I	1	3	None
INFO 1600	Introduction to Information Technology Concepts	1	3	None
COMP 1602	Computer Programming II	2	3	None
COMP 1603	Computer Programming III	2	3	None
COMP 1604	Mathematics for Computing	2	3	None
INFO 1601	Introduction to WWW Programming	2	3	None
<b>Level 2</b>	<b>Advanced Core Courses (30 Credits)</b>			
COMP 2600	Data Structures	1	3	COMP 1603
COMP 2601	Computer Architecture	1	3	COMP 1600
COMP 2602	Computer Networks	1	3	COMP 1600
INFO 2602	Web Programming and Technologies I	1	3	INFO 1601
MATH 2250	Industrial Statistics	1	3	None
COMP 2603	Object-Oriented Programming I	2	3	COMP 1603
COMP 2604	Operating Systems	2	3	COMP 1600
COMP 2605	Enterprise Database Systems	2	3	COMP 1602
COMP 2606	Software Engineering I	2	3	COMP 1603
INFO 2604	Information Systems Security	2	3	COMP 1602
<b>Level 3</b>	<b>Advanced Core Courses (15 credits)</b>			
COMP 3600	Software Engineering II	1	3	COMP 2606
COMP 3601	Design and Analysis of Algorithms	1	3	COMP 2600
COMP 3602	Theory of Computing	2	3	COMP 2600
COMP 3603	Human-Computer Interaction	2	3	COMP 2606
INFO 3604	Project	2	3	COMP 2606
<b>Level 2 / Level 3</b>	<b>Electives (Any 15 credits from the list below subject to pre-requisites being met)</b>			
COMP 3605	Introduction to Data Analytics	1	3	MATH 2250
COMP 3606	Wireless and Mobile Computing	1	3	COMP 2602
COMP 3607	Object-Oriented Programming II	2	3	COMP 2603

COMP 3608	Intelligent Systems	1	3	COMP 2600 <b>AND</b> MATH 2250
COMP 3609	Game Programming	1	3	COMP 2603 <b>AND</b> COMP 2606
COMP 3610	Big Data Analytics	2	3	COMP 3605
COMP 3611	Modelling and Simulation	1	3	MATH 2250
COMP 3612	Special Topics in Computer Science	1/2	3	COMP 2600 <b>AND</b> COMP 2603
INFO 2605	Professional Ethics and Law	2	3	INFO 1600 <b>OR</b> COMP 1600
INFO 3600	Business Information Systems	1	3	COMP 2605
INFO 3605	Fundamentals of LAN Technologies	1	3	INFO 2601 <b>OR</b> COMP 2604
INFO 3606	Cloud Computing	1	3	COMP 2605
INFO 3607	Fundamentals of WAN Technologies	2	3	INFO 2601 <b>OR</b> COMP 2604
INFO 3608	E-Commerce	2	3	INFO 2600 <b>OR</b> COMP 2606
INFO 3609	Internship	3	3	INFO 2600 <b>OR</b> COMP 2606

**B.Sc. Computer Science with Management**

Course Code	Course Title	Semester	Credits	Prerequisite(s)
<b>Level 1</b>	<b>Introductory Core Courses (12 credits)</b>			
COMP 1600	Introduction to Computing Concepts	1	3	None
COMP 1601	Computer Programming I	1	3	None
COMP 1602	Computer Programming II	2	3	None
COMP 1603	Computer Programming III	2	3	None
	<b>Additional Introductory Core Courses (12 credits)</b>			
ACCT 1002	Introduction to Financial Accounting	1	3	None
ECON 1001	Introduction to Economics I	1	3	None
ACCT 1003	Introduction to Cost and Managerial Accounting	2	3	None
ECON 1002	Introduction to Economics II	2	3	None
<b>Level 2</b>	<b>Advanced Core Courses (18 credits)</b>			
COMP 2600	Data Structures	1	3	COMP 1603
COMP 2601	Computer Architecture	1	3	COMP 1600
COMP 2602	Computer Networks	1	3	COMP 1600
COMP 2603	Object-Oriented Programming I	2	3	COMP 1603
COMP 2604	Operating Systems	2	3	COMP 1600
COMP 2605	Enterprise Database Systems	2	3	COMP 1602
	<b>Additional Advanced Core Courses (15 credits)</b>			
MGMT 2021	Business Law	1	3	
MKTG 2001	Principles of Marketing	2	3	
MGMT 2008	Organisational Behaviour	2	3	
MGMT 2032	Managerial Economics	2	3	
	<b>Any 3 credits from the list below:</b>			
ACCT 2017	Management Accounting	1	3	
MGMT 2012	Quantitative Methods	1	3	
MKTG 3000	Marketing Management	1	3	
MGMT 2023	Financial Management	1	3	
<b>Level 2 / Level 3</b>	<b>Electives (Any 12 credits from the list below subject to pre-requisites being met)</b>			
COMP 2606	Software Engineering I	2	3	COMP 1603
COMP 3600	Software Engineering II	1	3	COMP 2606
COMP 3601	Design and Analysis of Algorithms	1	3	COMP 2600
COMP 3602	Theory of Computing	2	3	COMP 2600
COMP 3603	Human-Computer Interaction	2	3	COMP 2606
COMP 3605	Introduction to Data Analytics	1	3	MATH 2250
COMP 3606	Wireless and Mobile Computing	1	3	COMP 2602
COMP 3607	Object-Oriented Programming II	2	3	COMP 2603
COMP 3608	Intelligent Systems	1	3	COMP 2600 <b>AND</b> MATH 2250

COMP 3609	Game Programming	1	3	COMP 2603 AND COMP 2606
COMP 3610	Big Data Analytics	2	3	MATH 2250
COMP 3611	Modelling and Simulation	1	3	MATH 2250
COMP 3612	Special Topics in Computer Science	1/2	3	COMP 2600 AND COMP 2603
INFO 2602	Web Programming and Technologies I	1	3	INFO 1601
INFO 2604	Information Systems Security	2	3	COMP 1602
INFO 2605	Professional Ethics and Law	2	3	INFO 1600 OR COMP 1600
INFO 3600	Business Information Systems	1	3	COMP 2605
INFO 3604	Project	2	3	INFO 2600 OR COMP 2606
INFO 3605	Fundamentals of LAN Technologies	1	3	INFO 2601 OR COMP 2604
INFO 3606	Cloud Computing	1	3	COMP 2605
INFO 3607	Fundamentals of WAN Technologies	2	3	INFO 2601 OR COMP 2604
INFO 3608	E-Commerce	2	3	INFO 2600 OR COMP 2606
INFO 3609	Internship	2	3	INFO 2600 OR COMP 2606
MATH 2250	Industrial Statistics	1	3	None
<b>Level 2 / Level 3</b>	<b>Additional Electives (15 credits)</b>			
	Any fifteen (15) credits from advanced Computer Science, Economics, Information Technology, Mathematics, or Management courses.			



**B.Sc. General (Major in Computer Science)**

<b>Course Code</b>	<b>Course Title</b>	<b>Semester</b>	<b>Credits</b>	<b>Prerequisite(s)</b>
<b>Level 1</b>	<b>Introductory Core Courses (12 credits)</b>			
COMP 1600	Introduction to Computing Concepts	1	3	None
COMP 1601	Computer Programming I	1	3	None
COMP 1602	Computer Programming II	2	3	None
COMP 1603	Computer Programming III	2	3	None
<b>Level 2</b>	<b>Advanced Core Courses (18 credits)</b>			
COMP 2600	Data Structures	1	3	COMP 1603
COMP 2601	Computer Architecture	1	3	COMP 1600
COMP 2602	Computer Networks	1	3	COMP 1600
COMP 2603	Object-Oriented Programming I	2	3	COMP 1603
COMP 2604	Operating Systems	2	3	COMP 1600
COMP 2605	Enterprise Database Systems	2	3	COMP 1602
<b>Level 2 / Level 3</b>	<b>Electives (Any 12 credits from the list below subject to pre-requisites being met)</b>			
COMP 2606	Software Engineering I	2	3	COMP 1603
COMP 3600	Software Engineering II	1	3	COMP 2606
COMP 3601	Design and Analysis of Algorithms	1	3	COMP 2600
COMP 3602	Theory of Computing	2	3	COMP 2600
COMP 3603	Human-Computer Interaction	2	3	COMP 2606
COMP 3605	Introduction to Data Analytics	1	3	MATH 2250
COMP 3606	Wireless and Mobile Computing	1	3	COMP 2602
COMP 3607	Object-Oriented Programming II	2	3	COMP 2603
COMP 3608	Intelligent Systems	1	3	COMP 2600 <b>AND</b> MATH 2250
COMP 3609	Game Programming	1	3	COMP 2603 <b>AND</b> COMP 2606
COMP 3610	Big Data Analytics	2	3	MATH 2250
COMP 3611	Modelling and Simulation	1	3	MATH 2250
COMP 3612	Special Topics in Computer Science	1/2	3	COMP 2600 <b>AND</b> COMP 2603
INFO 2602	Web Programming and Technologies I	1	3	INFO 1601
INFO 2604	Information Systems Security	2	3	COMP 1602
INFO 2605	Professional Ethics and Law	2	3	INFO 1600 <b>OR</b> COMP 1600
INFO 3600	Business Information Systems	1	3	COMP 2605
INFO 3604	Project	2	3	INFO 2600 <b>OR</b> COMP 2606

INFO 3605	Fundamentals of LAN Technologies	1	3	INFO 2601 <b>OR</b> COMP 2604
INFO 3606	Cloud Computing	1	3	COMP 2605
INFO 3607	Fundamentals of WAN Technologies	2	3	INFO 2601 <b>OR</b> COMP 2604
INFO 3608	E-Commerce	2	3	INFO 2600 <b>OR</b> COMP 2606
INFO 3609	Internship	2	3	INFO 2600 <b>OR</b> COMP 2606
MATH 2250	Industrial Statistics	1	3	None

### Minor in Computer Science

Course Code	Course Title	Semester	Credits	Prerequisite(s)
<b>Level 1</b>	<b>Introductory Core Courses (12 credits)</b>			
COMP 1600	Introduction to Computing Concepts	1	3	None
COMP 1601	Computer Programming I	1	3	None
COMP 1602	Computer Programming II	2	3	None
COMP 1603	Computer Programming III	2	3	None
<b>Level 2</b>	<b>Advanced Core Courses (Any 9 credits from the list below)</b>			
COMP 2600	Data Structures	1	3	COMP 1603
COMP 2601	Computer Architecture	1	3	COMP 1600
COMP 2603	Object-Oriented Programming I	2	3	COMP 1603
COMP 2604	Operating Systems	2	3	COMP 1600
<b>Level 2 / Level 3</b>	<b>Electives (Any 6 credits from the list below subject to pre-requisites being met)</b>			
COMP 2602	Computer Networks	2	3	COMP 1600
COMP 2605	Enterprise Database Systems	2	3	COMP 1602
COMP 2606	Software Engineering I	2	3	COMP 1603
COMP 3600	Software Engineering II	1	3	COMP 2606
COMP 3601	Design and Analysis of Algorithms	1	3	COMP 2600
COMP 3602	Theory of Computing	2	3	COMP 2600
COMP 3603	Human-Computer Interaction	2	3	COMP 2606
COMP 3605	Introduction to Data Analytics	1	3	MATH 2250
COMP 3606	Wireless and Mobile Computing	1	3	COMP 2602
COMP 3607	Object-Oriented Programming II	2	3	COMP 2603
COMP 3608	Intelligent Systems	1	3	COMP 2600 <b>AND</b> MATH 2250
COMP 3609	Game Programming	1	3	COMP 2603 <b>AND</b> COMP 2606
COMP 3610	Big Data Analytics	2	3	MATH 2250
COMP 3611	Modelling and Simulation	1	3	MATH 2250
COMP 3612	Special Topics in Computer Science	1/2	3	COMP 2600 <b>AND</b> COMP 2603
INFO 2602	Web Programming and Technologies I	1	3	INFO 1601
INFO 2604	Information Systems Security	2	3	COMP 1602
INFO 2605	Professional Ethics and Law	2	3	INFO 1600 <b>OR</b> COMP 1600
INFO 3600	Business Information Systems	1	3	COMP 2605
INFO 3604	Project	2	3	INFO 2600 <b>OR</b>

				COMP 2606
INFO 3605	Fundamentals of LAN Technologies	1	3	INFO 2601 <b>OR</b> COMP 2604
INFO 3606	Cloud Computing	1	3	COMP 2605
INFO 3607	Fundamentals of WAN Technologies	2	3	INFO 2601 <b>OR</b> COMP 2604
INFO 3608	E-Commerce	2	3	INFO 2600 <b>OR</b> COMP 2606
INFO 3609	Internship	3	3	INFO 2600 <b>OR</b> COMP 2606
MATH 2250	Industrial Statistics	1	3	None

## 3.2 Teaching and Learning Strategies

The B.Sc. Computer Science programmes aim to provide graduates with the knowledge, skills and attitudes to take on appropriate professional positions in computing and information technology on graduation and grow into leadership positions or pursue research or graduate studies in the field. Thus, experiential learning will permeate the curriculum. Experiential learning can be defined as a series of structured and directed learning activities whereby the educator purposefully engages the student during the learning experience to clarify knowledge, skills and attitudes followed by focused reflection by the student. Experiential learning can be provided in a variety of different ways and the Computer Science programmes will incorporate many of those listed below:

- Instructor demonstrations
- Structured and unstructured labs giving students hands-on experience with current technologies
- Multi-stage individual and group projects
- Interviews with computing and information technology professionals and/or job shadowing
- Design, implementation, and documentation projects
- Preparation and presentation of technical reports
- Internships

## 4.0 Admission

Applicants to the B.Sc. Computer Science programmes should have two CAPE subjects (or GCE 'A' Levels or equivalent) at least one of which must be a Science subject, Accounting, or Economics. Holders of Associate Degrees in the field of Computing may also be considered for entry into either B.Sc. programme. This will be assessed on a case by case basis and may lead to exemptions from various Level 1 courses.

This is an implementation of a key recommendation from the recent Quality Assurance Review of the Department of Computing and Information Technology which took place in April 2015. The change in admission requirements is intended to widen the pool of potential applicants to the B.Sc. Computer Science programmes which was previously limited to applicants with a pass in CAPE Mathematics (or equivalent).

The expected intake for the B.Sc. Computer Science programmes is about 100 per year.

## 5.0 Resources Required

The new programmes in Computer Science and Information Technology consist of 30 core courses and 16 elective courses. Therefore, 46 courses will be offered each academic year (unless some electives are offered in alternate years). The department has a budgeted staff complement of 13 members of academic staff. Assuming a workload of four courses per academic year (12 credits, which is the norm at the University of the West Indies), this means that the department is potentially capable of offering 52 courses each academic year. However, both the Head of Department and another member of staff who serves as a Deputy Dean have a reduced workload of two courses per academic year. Therefore, the real capacity of the department is 48 courses.

Some of the programming intensive courses as well as the mathematics courses tend to have high failure rates (e.g., COMP 1601, COMP 1604, COMP 2600, and COMP 3601). The department normally teaches courses such as these in both semesters to enable a student to progress fairly smoothly through their respective programmes despite failing one of these courses. If the department teaches these four courses in both semesters, it will be required to teach 50 courses each academic year. This is just over the workload of the current budgeted members of academic staff.

However, it does not allow the department to stream some of the Level 1 courses given the new common core for both the B.Sc. Computer Science and B.Sc. Information Technology students. Given the increased teaching support that will be required for the combined Level 1 courses as well as the need to offer problematic courses in both semesters, the department will need the services of two additional full-time members of staff at the Instructor level.

In terms of reading material, the library and the Internet have usually been the chief source of information for students pursuing the existing B.Sc. Computer Science and B.Sc. Information Technology degrees. The department has had no issues with these sources of information in the past and we expect this to be the case when the new programmes are launched.

## 6.0 Detailed Course Descriptions

This section gives course outlines for the courses that make up the Computer Science programmes. These include all the new Computer Science courses, the MATH courses, and the “management” courses from the Faculty of Social Sciences”. Since the management courses are already part of other programmes, only a brief description of each course is given. The detailed course outlines for the INFO courses are given in a separate document, *Proposal for a Revision of the B.Sc. Information Technology Programmes*.

It should be noted that whenever an assignment is scheduled on the Course Calendar, this is the date when the assignment is given out to the students of the course.

### 6.1 Computer Science Courses

**COURSE TITLE: Introduction to Computing Concepts**

**COURSE CODE: COMP 1600**

**TYPE: Core**

**LEVEL: 1**

**SEMESTER: 1**

**START DATE: SEPT-01-2016**

**DEPARTMENT and FACULTY: DCIT/FST**

**CREDITS: 3**

**PRE-REQUISITE(S): None**

**TEACHING STRATEGIES:**

Lectures, Class discussions, Tutorials, Problem solving sessions

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 2-hour lab, 3 hours independent study

**COURSE DESCRIPTION**

This course presents an overview of computing technology and the field of computer science. Discussion topics will include the organization of modern computers, operating systems, algorithms, programming languages and database systems.

**COURSE RATIONALE**

This course provides a fundamental understanding of the Computer Science discipline by focusing on the computer’s role in representing, storing, manipulating, organizing and communicating information. It will provide students with a foundation for future courses in Computer Science. In order to understand the capabilities of computer systems, students must recognize how the various forms of data are stored digitally, how the major hardware components store and operate on such data, and how software is developed and used to control the various subsystems. Students will therefore be introduced to many areas of Computer Science such as computer architecture, operating systems, algorithms, programming languages and database systems.

**COURSE CONTENT**

1. Data Storage
  - 1.1. Gates: AND, OR, NOT, NAND
  - 1.2. 1-bit memory, flip-flop

- 1.3. Main memory organization; storage hierarchy
- 1.4. Data representation: characters, integers, floating-point numbers
2. Data Manipulation
  - 2.1. CPU, registers, instruction execution cycle, instruction set
  - 2.2. Stored program concept, program execution
  - 2.3. Performance enhancements: cache, pipelining
  - 2.4. I/O modules
3. Operating System Fundamentals
  - 3.1. Management of resources
  - 3.2. Scheduling - process, process table, swapping
  - 3.3. Memory management - partitions, demand paging, virtual memory
4. Algorithms: Iterative and recursive structures; efficiency and correctness
5. Programming Languages
  - 5.1. Program translation; programming paradigms
  - 5.2. Procedural, Object-oriented and declarative programming
6. Database Structures
  - 6.1. Database issues
  - 6.2. The relational model, relational operations - select, join, project

## COURSE LEARNING OUTCOMES

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

1. Analyze a simple circuit involving AND, OR, and NOT gates to determine what input bit pattern will produce a specified output
2. Describe how 1 bit is stored using a flip flop circuit
3. Describe the storage hierarchy
4. Determine the internal representation of an integer, a character or a floating-point number
5. Describe the execution of an instruction
6. Trace a given simple machine language program to determine the task performed
7. Differentiate between the various ways a processor can communicate with I/O devices
8. Explain operating system concepts such as scheduling, swapping, paging and virtual memory
9. Trace through simple iterative and recursive algorithms
10. Distinguish between the procedural, object-oriented and declarative programming paradigms
11. Perform relational operations such as select, join and project on database tables
12. Describe the role of digital logic in computer design

## COURSE ASSESSMENT

Assessment	Learning Outcomes												Weighting %	Assessment Description	Duration
	1	2	3	4	5	6	7	8	9	10	11	12			
Assignment 1	X	X	X	X	X								20%	Short answer	
Assignment 2				X	X	X	X	X	X					Short answer	
Exam 1	X	X	X	X	X	X	X						30%	Problems & short answer	1.5 hour
Exam 2							X	X	X	X	X			Problems & short answer	1.5 hour
Final Examination	X	X	X	X	X	X	X	X	X	X	X	X	50%	Problems & short answer questions	2 hours
TOTAL %													100%		

## COURSE CALENDAR

Week	Topic	Readings	Activities/Assessments
1,2	Data Storage Gates: AND, OR, NOT, NAND 1-bit memory, flip-flop; Main memory organization; storage hierarchy; Data representation: characters, integers, floating-point numbers		
3,4	Data Manipulation CPU, registers, instruction execution cycle, instruction set; Stored program concept, program execution Performance enhancements: cache, pipelining; I/O modules		Assignment 1 due (Week 4)
5	Operating System Fundamentals Management of resources; Scheduling – process states, process table, swapping; Memory management - partitions, demand paging, virtual memory		Coursework Examination 1
6	Algorithms: Iterative and recursive structures; efficiency and correctness		
7,8	Algorithms continued.		Assignment 2 due (Week 8)
9,10	Database Structures Database issues The relational model, relational operations - select, join, project		
11,12	Programming Languages Program translation; programming paradigms: procedural, object-oriented and declarative programming		Coursework Examination 2 (Week 11)
13	Review		

## RESOURCES

- Lecture Notes, Hand-outs
- Brookshear, J. Glenn. Computer Science: An Overview, 11<sup>th</sup> Edition. 2011. Boston, MA: Addison Wesley.



**COURSE TITLE: Computer Programming I**

**COURSE CODE: COMP 1601**

**TYPE: Core**

**LEVEL: 1**

**SEMESTER: 1**

**START DATE: SEPT-01-2016**

**DEPARTMENT and FACULTY: DCIT/FST**

**CREDITS: 3**

**PRE-REQUISITE(S): None**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 2-hour lab, 3 hours independent study

**COURSE DESCRIPTION**

This course uses an appropriate programming language as a tool to teach fundamental programming concepts. The main concepts covered are sequence, selection and repetition logic, character and string manipulation, functions, and a basic introduction to arrays and their applications.

**COURSE RATIONALE**

This course equips students to solve problems on computer based systems. It identifies what type of problems can be solved by such systems and which cannot. It guides students on methods of developing structured algorithms. The focus on this course is problem description and presentation using either flowcharting or pseudocode tools. The selected programming language is used as a vehicle to show the basics of programming algorithms.

**COURSE CONTENT**

1. Steps in Problem Solving
  - 1.1. Problem solving steps to solve a problem that has an algorithmic solution
  - 1.2. Structured and unstructured approaches to problem solving
2. Elementary Programming Concepts
  - 2.1. Concept of a program
  - 2.2. Assembler, interpreters and compilers
  - 2.3. Number representation in computers
3. Programming Language Basics
  - 3.1. Fundamental concepts for solving problems on computers
  - 3.2. Tokens in the programming language
  - 3.3. Basic data types
4. Sequence Logic
  - 4.1. Data collection (reading from the keyboard)
  - 4.2. Processing (Assigning variables, calculation, operators and logic)
  - 4.3. Output (Displaying data to the screen)
5. Selection Logic
  - 5.1. If construct
  - 5.2. If- else construct
  - 5.3. Nested If
6. Repetition Logic
  - 6.1. While construct
  - 6.2. For construct

- 6.3. Do-while Loop
- 7. One Dimensional Arrays
  - 7.1. Declaration, Storage and Retrieval of Array Elements
  - 7.2. Do basic Problems involving 1-D arrays
- 8. Characters and Strings
  - 8.1. Character representation in computers
  - 8.2. String operations
- 9. Functions
  - 9.1. Modularization in program design
  - 9.2. Function headers
  - 9.3. How to return a value
  - 9.4. Scope of variables

## COURSE LEARNING OUTCOMES

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

1. Explain fundamental concepts for solving problems on the computer.
2. Describe how data is stored and organized.
3. Use arithmetic and logical operators in the problem solving process.
4. Create programs in the selected programming language for solving simple problems.
5. Design, implement, test, and debug a program that uses any or all of the following fundamental programming constructs: basic computation, simple I/O, standard conditional and iterative structures and functions

## COURSE ASSESSMENT

Assessment	Learning Outcomes					Weighting %	Assessment Description	Duration
	1	2	3	4	5			
Assignment 1	X	X	X			20%	Problems	
Assignment 2	X	X	X	X			Problems	
Assignment 3	X	X	X	X	X		Problems	
Exam 1	X	X	X	X	X	30%	Problems & short answer	1.5 hour
Exam 2	X	X	X	X	X		Problems & short answer	1.5 hour
Final Examination	X	X	X	X	X	50%	Problems & short answer questions	2 hours
TOTAL %						100%		

## TEACHING STRATEGIES

Interactive lectures will be used to introduce the main concepts of the course. However, discussion and brain storming methods will be used throughout the course with the goal of allowing students to engage the problem solving process. Labs will also be used to demonstrate, in a practical manner, the concepts taught in lectures.

## RESOURCES

### Lecture Notes

#### Recommended Textbooks

- Noel Kalicharan. C Programming - A Beginner's Course. 2005. CreateSpace Independent Publishing Platform.

**COURSE CALENDAR: 39 hours** (3 hours per week)

Week	Topic
Week 1	Steps in solving a problem – use HIPO diagrams, flowcharts and trace tables
Week 2, 3	Elementary Programming Concepts (Sequence/Selection Logic) Assignment 1 (Week 3)
Week 4	Repetition Logic
Week 5,6	Introduce Arrays and Continue to Look at Repetition Logic Assignment 2 (Week 6)
Week 7	CW Examination I
Week 8	Functions
Week 9	Functions – passing by reference and value Assignment 3
Week 10	Strings / Functions and Character Manipulation
Week 11	CW Examination II
Week 12	Putting it all together
Week 13	Revision

**COURSE TITLE: Computer Programming II**

**COURSE CODE: COMP 1602**

**TYPE: Core**

**LEVEL: 1**

**SEMESTER: 2**

**START DATE: JAN-15-2017**

**DEPARTMENT and FACULTY: DCIT/FST**

**CREDITS: 3**

**PRE-REQUISITE(S): None**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 2-hour lab, 3 hours independent study

**COURSE DESCRIPTION**

This course uses an appropriate programming language as a tool to teach intermediate programming concepts. The main concepts covered are structures, one and two dimensional arrays and applications involving searching, sorting and merging, random number generation, numerical methods, games and simulation.

**COURSE RATIONALE**

This course introduces new programming concepts. It focuses on the use of common searching and sorting methods and character and string manipulation which are two very important concepts that are necessary in most software systems.

**COURSE CONTENT**

1. Arrays
  - 1.1. Declaring one-dimensional arrays
  - 1.2. Finding largest and smallest values
  - 1.3. Passing arrays as arguments
  - 1.4. Array of Strings
2. Structures
  - 2.1. Declaration
  - 2.2. User Defined Types
  - 2.3. Array of Structures
  - 2.4. Nested Structures
3. Sorting, Searching and Merging
  - 3.1. Sorting arrays
  - 3.2. Inserting elements in place
  - 3.3. Sorting parallel arrays
  - 3.4. Linear Binary Search
  - 3.5. Insertion Sort, Selection Sort
  - 3.6. Merging ordered lists
4. Two dimensional arrays
5. Random Numbers
  - 5.1. Random and pseudo random numbers
  - 5.2. Random number generation and ranges
6. Games and Simulation
  - 6.1. Simulation of real-life problems

- 6.2. Simulating a queue
- 7. Numerical Methods
- 8. Files
  - 8.1. Reading data from a file
  - 8.2. Sending output to a file
  - 8.3. Binary files
- 9.
  - 9.1. Text file vs. Binary file
  - 9.2. Opening and closing files
  - 9.3. Random Access files
  - 9.4. Indexed files

## COURSE LEARNING OUTCOMES

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

1. Design programs which utilize one and two-dimensional arrays
2. Implement common searching, sorting and merging techniques.
3. Use random number generation in simulation and game development.
4. Manipulate text and binary files

## COURSE ASSESSMENT

Assessment	Learning Outcomes				Weighting %	Assessment Description	Duration
	1	2	3	4			
Assignment 1	X	X			20%	Problems	
Assignment 2	X	X	X			Problems	
Assignment 3	X	X	X	X		Problems	
Exam 1	X	X			30%	Problems & short answer	1.5 hour
Exam 2	X	X	X	X		Problems & short answer	1.5 hour
Final Examination	X	X	X	X	50%	Problems & Short Answer Questions	2 hours
TOTAL %					100%		

## TEACHING STRATEGIES

Interactive lectures will be used to introduce the main concepts of the course. However, discussion and brain storming methods will be used throughout the course with the goal of allowing students to engage the problem solving process. Labs will also be conducted to give the students an opportunity to apply what they have learnt in a hands-on manner.

## RESOURCES

### Lecture notes

### Textbooks

- Noel Kalicharan. C Programming - A Beginner's Course. 2005. CreateSpace Independent Publishing Platform.

- Noel Kalicharan. C Programming - An Advanced Course. 2006. CreateSpace Independent Publishing Platform.

**COURSE CALENDAR: 39 hours** (3 hours per week)

<b>Week</b>	<b>Topic</b>
Week 1, 2	One – dimensional arrays
Week 3, 4	Introduction to Two dimensional arrays Sorting, Searching and Merging <b>Assignment 1</b> <b>(Week 3)</b>
Week 5, 6	Two dimensional arrays
Week 7	<b>CW Examination I</b> / Random Number Generation
Week 8,9	Games and Simulation <b>Assignment 2</b> <b>(Week 8)</b>
Week 10	Numerical Methods <b>Assignment 3</b>
Week 11	Files
Week 12	Files / <b>CW Examination II</b>
Week 13	Revision

**COURSE TITLE: Computer Programming III**

**COURSE CODE: COMP 1603**

**TYPE: Core**

**LEVEL: 1**

**SEMESTER: 2**

**START DATE: JAN-15-2017**

**DEPARTMENT and FACULTY: DCIT/FST**

**CREDITS: 3**

**PRE-REQUISITE(S): None**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 2-hour lab, 3 hours independent study

**COURSE DESCRIPTION**

This course uses an appropriate programming language as a tool to teach intermediate programming concepts. The main concepts covered are pointers, linked lists, stacks and queues and their implementations using arrays and linked lists and recursion.

**COURSE RATIONALE**

This course introduces new programming concepts and abstract data types. It focuses on the use of common abstract data types such as linked lists, stacks and queues which are important concepts that are inherent in most systems. It introduces the concept of dynamic storage methods which are necessary in programming easily adaptive systems. It also covers the conditions under which dynamic storage is applicable and when it is not.

**COURSE CONTENT**

1. Structures
  - 1.1. Declaration
  - 1.2. User Defined Types
  - 1.3. Array of Structures
  - 1.4. Nested Structures
2. Pointers
  - 2.1. Concept of a pointer
  - 2.2. Pointers as arguments
  - 2.3. Pointer arithmetic
3. Linked Lists
  - 3.1. Operations on a Linked List (inserting, deleting, counting, searching)
  - 3.2. Allocating Memory (e.g., sizeof, malloc, calloc, and free in C)
  - 3.3. Sorted Lists
  - 3.4. Merging
4. Abstract Data Types (ADT)
  - 4.1. Stacks (Array and Linked List Implementation)
    - 4.1.1. Creating a Stack Header File
    - 4.1.2. How to convert from infix to postfix
  - 4.2. Queues (Array and Linked List Implementation)
5. Recursion
  - 5.1. Recursive Functions
  - 5.2. Recursive Applications

## COURSE LEARNING OUTCOMES

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

1. Design and use structures to model real world objects and concepts
2. Explain what a pointer is and how it can be used.
3. Explain what a Linked List is and how it can be used.
4. Implement the Stack ADT.
5. Implement the Queue ADT.
6. Solve programming problems that require the use of stacks and queues.
7. Use recursion as a powerful technique in problem solving.

## COURSE ASSESSMENT

Assessment	Learning Outcomes							Weighting %	Assessment Description	Duration
	1	2	3	4	5	6	7			
Assignment 1	X	X						20%	Problems	
Assignment 2	X	X	X						Problems	
Assignment 3	X	X	X	X	X	X			Problems	
Exam 1	X	X	X					30%	Problems & short answer	1.5 hour
Exam 2	X	X	X	X	X	X			Problems & short answer	1.5 hour
Final Examination	X	X	X	X	X	X	X	50%	Problems & Short Answer Questions	2 hours
TOTAL %								100%		

## TEACHING STRATEGIES

Interactive lectures will be used to introduce the main concepts of the course. However, discussion and brain storming methods will be used throughout the course with the goal of allowing students to engage the problem solving process. Labs will also be used to demonstrate, in a practical manner, the concepts taught in lectures.

## RESOURCES

### Lecture notes

### Textbooks

- Noel Kalicharan. C Programming - An Advanced Course. 2006. CreateSpace Independent Publishing Platform.

## COURSE CALENDAR:

Each week comprises of 2 lecture hours and 2 practical lab hours (2-hour lab is substituted for 1-hour tutorial)

Week	Topic
Week 1	Structures
Week 2, 3	Pointers Assignment 1



	(Week 2)
Week 4, 5, 6	Linked Lists Assignment 2 (Week 5)
Week 7, 8	Stacks, CW Examination I (CW Examination in Week 8)
Week 9, 10	Queues Assignment 3 (Week 10)
Week 11, 12	Recursion / CW Examination II (Week 11)
Week 13	Revision

**COURSE TITLE: Mathematics for Computing**

**COURSE CODE: COMP 1604**

**TYPE: Core**

**LEVEL: 1**

**SEMESTER: 2**

**START DATE: JAN-15-2017**

**DEPARTMENT and FACULTY: DCIT/FST**

**CREDITS: 3**

**PRE-REQUISITE(S): None**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 1-hour tutorial, 5 hours per week independent study

**COURSE DESCRIPTION**

This course introduces students to the basic mathematical structures and computational techniques that are considered to be the foundation for courses in computer science and information technology. Students are also taught how to reason logically and how to solve problems using various proof techniques. The main mathematical structures covered are logic, sets, relations and functions.

**COURSE RATIONALE**

The material in this course is pervasive in many areas in computing such as data structures, algorithms, security, databases and programming languages. The ability to reason logically is important in order to write correct computer programs and students are also expected to create and understand proofs in courses such as data structures, algorithms, intelligent systems and security.

**COURSE CONTENT**

1. Introduction to Propositional and Predicate Logic
  - 1.1. Propositional logic: statements; compound statements and truth tables; logical equivalence; valid and invalid arguments; rules of inference.
  - 1.2. Predicate logic: predicates and quantified statements; negation of quantified statements; arguments with quantified statements.
2. Basic Proof Techniques (with examples taken from simple number theory)
  - 2.1. Direct proofs
  - 2.2. The contrapositive argument
  - 2.3. Proof by contradiction
  - 2.4. Proof by counter-example
  - 2.5. Proofs using the first and second principles of mathematical induction
3. Sets, Relations and Functions
  - 3.1. Sets: basic definitions; subsets; set equality and operations on sets; Venn diagrams; Power sets, Cartesian products; set identities; proving set identities.
  - 3.2. Relations: relations on sets; reflexivity, symmetry and transitivity; equivalence relations; modular arithmetic.
  - 3.3. Functions: definition; injective, surjective and bijective functions; the Pigeonhole Principle.
4. Counting Techniques
  - 4.1. Sum and product rules
  - 4.2. Permutations and combinations
  - 4.3. Pascal triangle and the binomial theorem.

## COURSE LEARNING OUTCOMES

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

1. Use rules of inference to construct proofs in propositional and predicate logic
2. Identify the proof techniques used in a given proof
3. Select an appropriate proof strategy for a problem
4. Perform operations associated with sets, relations and functions
5. Apply counting arguments and compute permutations and combinations.

## COURSE ASSESSMENT

Assessment	Learning Outcomes					Weighting %	Assessment Description	Duration
	1	2	3	4	5			
Assignment 1	X	X				20%	Worksheets	
Assignment 2	X	X	X				Worksheets	
Assignment 3	X	X	X	X	X		Worksheets	
Exam 1	X	X	X			30%	Short answer	1.5 hour
Exam 2	X	X	X	X			Short answer	1.5 hour
Final Examination	X	X	X	X	X	50%	Short Answer Questions	2 hours
TOTAL %						100%		

## TEACHING STRATEGIES

Interactive lectures will be used to introduce the main concepts of the course. However, discussions and problem solving sessions will be used throughout the course with the goal of allowing students to engage in the problem solving process.

## RESOURCES

### Lecture Notes

### Recommended Textbook

- Epp, S - Discrete Mathematics with Applications, 4<sup>th</sup> edition. 2010. Pacific Grove, CA: Brooks Cole.

## COURSE CALENDAR:

Each week comprises of 2 lecture hours and a 1-hour tutorial.

Week	Topic
Week 1, 2	Propositional logic
Week 3,4	Predicate logic. Assignment 1 (Week 3)
Week 4, 5	Basic proof techniques
Week 6	Principles of mathematical induction. Assignment 2

Week 7, 8	Sets. CW Examination I (Week 8)
Week 9, 10	Relations. Modular Arithmetic and applications to cryptography. Assignment 3 (Week 10)
Week 11	Functions. Pigeonhole principle CW Examination II.
Week 12	Counting. Permutations. Combinations. Binomial theorem.
Week 13	Revision.

**COURSE TITLE: Data Structures**  
**COURSE CODE: COMP 2600**  
**TYPE: Core**  
**LEVEL: 2**  
**SEMESTER: 1**  
**START DATE: SEPT-01-2016**  
**DEPARTMENT and FACULTY: DCIT/FST**  
**CREDITS: 3**  
**PRE-REQUISITE(S): COMP 1603**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 2-hour labs, 6 hours per week independent study

**COURSE DESCRIPTION**

A data structure is a way of storing data in a computer so that it can be used efficiently. Data structures is an important part of the equation; Programs = Algorithms + Data structures. Often a carefully chosen data structure will allow the most efficient algorithm to be used. A well-designed data structure allows a variety of critical operations to be performed, minimizing the use of execution time and memory space.

This course covers some fundamental data structures—stacks, queues, linked lists, binary trees, heaps and graphs—which are required for programming the solutions to a wide variety of real-world and theoretical problems.

**COURSE RATIONALE**

A critical aspect of solving any problem on a computer is its representation. Knowledge of data structures gives one a better chance of choosing an appropriate representation for a given problem.

**COURSE CONTENT**

1. Review of stacks, queues, linked lists.
2. Methods for solving the 'search and insert' problem.
3. Hashing. Hash functions. Clustering. Methods of resolving collisions, e.g. linear, quadratic, chaining, double hashing.
4. Trees. Binary trees. Search trees. Tree traversal. Analysis of binary search tree algorithm. Build binary trees. Build the best search tree from sorted data.
5. Heaps. Priority queues.
6. Sorting. Shell sort, quicksort, heapsort, mergesort.
7. Graph concepts and terminology. Representation of graphs. Depth-first and breadth-first traversals. Topological sort. Minimal cost paths – Dijkstra's and Bellman-Ford algorithms. Minimal cost spanning trees – Prim's and Kruskal's algorithms.
8. Efficiently storing and manipulating matrices with special properties, e.g. symmetric, triangular, band, sparse and others.
9. Write programs using any or all of the above data structures/techniques.

**COURSE LEARNING OUTCOMES**

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

1. Explain what pointers and linked lists and how they can be used.
2. Identify some common, useful structures for representing data inside a computer
3. Write algorithms for manipulating these data structures
4. Analyze these data structures so that the best choices can be made for a given problem
5. Write programs to solve problems which require using any or all of these data structures

## TEACHING STRATEGIES

A combination of lectures/tutorials will be used. Feedback from assignments/problem sheets/course work tests will be used to tailor subsequent lectures/tutorials. Students will be given ample opportunity to develop correct, readable programs using structured programming techniques.

## COURSE ASSESSMENT

Assessment	Learning Outcomes					Weighting %	Assessment Description	Duration
	1	2	3	4	5			
Assignment 1	X	X	X	X		20%	Problems	
Assignment 2	X	X	X	X	X		Problems	
Assignment 3	X	X	X	X	X		Problems	
Exam 1	X	X	X	X	X	30%	Problems & short answer	1.5 hour
Exam 2	X	X	X	X	X		Problems & short answer	1.5 hour
Final Examination	X	X	X	X	X	50%	Problems & Short Answer Questions	2 hours
TOTAL %						100%		

## RESOURCES

**Lecture notes (in-class and online)**

### Textbook

Noel Kalicharan. Data Structures in C. 2008. CreateSpace Independent Publishing Platform.

## CALENDAR

Week	Content
1-2	Revision of stacks, queues, linked lists, recursion. Assignment 1 (Week 2)
3-5	Trees, binary trees, heaps. Assignment 2 (Week 4)
6-8	Graphs.

	Examination 1 in Week 6
<b>9</b>	Search and Insert. Hashing. Assignment 3
<b>10-12</b>	Sorting. Matrices with special properties. Examination 2 in Week 10
<b>13</b>	General revision

**COURSE TITLE: Computer Architecture**  
**COURSE CODE: COMP 2601**  
**TYPE: Core**  
**LEVEL: 2**  
**SEMESTER: 1**  
**START DATE: SEP-01-2016**  
**DEPARTMENT and FACULTY: DCIT/FST**  
**CREDITS: 3**  
**PRE-REQUISITE(S): COMP 1600**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 1-hour tutorial, 4 hours per week independent study

**COURSE DESCRIPTION**

This course builds upon knowledge and skills developed in COMP 1600. This course explores how computers from a programmer's view point rather than from the hardware designer's perspective. Topics include: Digital Logic and Digital Systems, Machine Level Representation of Data, Assembly Level Machine Organization, Memory System Organization and Architecture, Interfacing and Communication, Multiprocessing and Alternative Architectures, and Performance Enhancements. The overarching theme of the course is the hardware-software interface; in particular, focusing on what a programmer needs to know about the underlying hardware to achieve high performance for his or her code.

**COURSE RATIONALE**

Computing professionals should not regard the computer as a black box that executes programs by magic. This course serves students in two ways. First, for those who want to continue studying computer architecture, embedded systems, and other low-level aspects of computer systems, it lays the foundation of detailed implementation experience needed to make the quantitative tradeoffs in more advanced courses meaningful. Second, for those students interested in other areas of computer science, it solidifies an intuition about why hardware is as it is and how software interacts with hardware.

**COURSE CONTENT**

1. Digital Logic and Digital Systems
  - 1.1. Overview and history of computer architecture
  - 1.2. Combinational vs. sequential logic
  - 1.3. Multiple representations/layers of interpretation (hardware is just another layer)
  - 1.4. Physical constraints (gate delays, fan-in, fan-out, energy/power)
2. Machine Level Representation of Data
  - 2.1. Bits, bytes, and words
  - 2.2. Numeric data representation and number bases
  - 2.3. Fixed- and floating-point systems
  - 2.4. Signed and twos-complement representations
  - 2.5. Representation of non-numeric data (character codes, graphical data)
  - 2.6. Representation of records and arrays
3. Assembly Level Machine Organization
  - 3.1. Basic organization of the von Neumann machine
  - 3.2. Control unit; instruction fetch, decode, and execution
  - 3.3. Instruction sets and types (data manipulation, control, I/O)



- 3.4. Assembly/machine language programming
- 3.5. Instruction formats
- 3.6. Addressing modes
- 3.7. Subroutine call and return mechanisms (cross-reference PL/Language Translation and Execution)
- 3.8. I/O and interrupts
- 3.9. Heap vs. Static vs. Stack vs. Code segments
- 3.10. Shared memory multiprocessors/multicore organization
- 3.11. Introduction to SIMD vs. MIMD and the Flynn Taxonomy
- 4. Memory System Organization and Architecture
  - 4.1. Storage systems and their technology
  - 4.2. Memory hierarchy: importance of temporal and spatial locality
  - 4.3. Main memory organization and operations
  - 4.4. Latency, cycle time, bandwidth, and interleaving
  - 4.5. Cache memories (address mapping, block size, replacement and store policy)
  - 4.6. Multiprocessor cache consistency/Using the memory system for inter-core synchronization/atomic memory operations
  - 4.7. Virtual memory (page table, TLB)
  - 4.8. Fault handling and reliability
  - 4.9. Error coding, data compression, and data integrity (cross-reference SF/Reliability through Redundancy)
- 5. Interfacing and Communication
  - 5.1. I/O fundamentals: handshaking, buffering, programmed I/O, interrupt-driven I/O
  - 5.2. Interrupt structures: vectored and prioritized, interrupt acknowledgment
  - 5.3. External storage, physical organization, and drives
  - 5.4. Buses: bus protocols, arbitration, direct-memory access (DMA)
  - 5.5. Introduction to networks: communications networks as another layer of remote access
  - 5.6. Multimedia support
  - 5.7. RAID architectures
- 6. Functional Organization
  - 6.1. Implementation of simple datapaths, including instruction pipelining
  - 6.2. Control unit: hardwired realization vs. microprogrammed realization
  - 6.3. Instruction pipelining
  - 6.4. Introduction to instruction-level parallelism (ILP)
- 7. Multiprocessing and Alternative Architectures
  - 7.1. Example SIMD and MIMD instruction sets and architectures
  - 7.2. Multiprocessor cache coherence
- 8. Performance Enhancements
  - 8.1. Superscalar architecture
  - 8.2. Branch prediction, Speculative execution, Out-of-order execution
  - 8.3. Prefetching
  - 8.4. Hardware support for multithreading
  - 8.5. Scalability
- 9. The Six “Great” Ideas
  - 9.1. Layers of Representation/Interpretation
  - 9.2. Moore’s Law
  - 9.3. Principle of Locality/Memory Hierarchy
  - 9.4. Parallelism
  - 9.5. Performance Evaluation
  - 9.6. Dependability via Redundancy

## COURSE LEARNING OUTCOMES

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

1. Describe the progression of computer technology including the trend of modern computer architectures such as multi-core and parallel processing.
2. Articulate that there are many equivalent representations of computer functionality, including logical expressions and gates.
3. Use mathematical expressions to describe the functions of simple combinational and sequential circuits.
4. Execute instructions as in a classical von Neumann machine, with extensions for threads multiprocessor synchronization, and SIMD execution.
5. Demonstrate how to map between high-level language patterns into assembly/machine language notations.
6. Write simple assembly language program segments.
7. Show how fundamental high-level programming constructs are implemented at the machine-language level.
8. Examine the workings of a system with virtual memory management.
9. Explain how interrupts are used to implement I/O control and data transfers.
10. Characterize the costs and benefits of prefetching.

## COURSE ASSESSMENT

Assessment	Learning Outcomes									Weighting %	Assessment Description	Duration
	1	2	3	4	5	6	7	8	9			
Assignment 1	X	X								20%	Problems & short answer	
Assignment 2	X	X	X	X							Problems & short answer	
Assignment 3			X	X	X	X	X				Problems & short answer	
Exam 1	X	X	X	X	X					30%	Problems & short answer	1.5 hour
Exam 2	X	X	X	X	X	X	X	X			Problems & short answer	1.5 hour
Final Examination	X	X	X	X	X	X	X	X	X	50%	Problems & Short Answer Questions	2 hours
TOTAL %										100%		

## TEACHING STRATEGIES

The course will be taught via interactive lectures, tutorials and in-class group work. Students will be expected to work in groups as well as individually on course assignments and projects.

## RESOURCES

Lecture notes - PowerPoint handouts of the lecture notes will be made available.

Stallings William, Computer Organization and Architecture: Designing for Performance, Addison-Wesley 7th edition, 2006

Douglas E. Comer, Essentials of Computer Architecture, Prentice Hall, 2005

J. L. Hennessy and D. A. Patterson, Computer Architecture: A Quantitative Approach, 5th Edition, Morgan Kaufmann Publishing Co., Menlo Park, CA. 2012

**COURSE CALENDAR: 39 hours** (3 hours per week)

<b>Week</b>	<b>Topic</b>
<b>1</b>	Digital Logic and Digital Systems
<b>2</b>	Machine Level Representation of Data Assignment 1
<b>3</b>	Machine Level Representation of Data
<b>4</b>	Assembly Level Machine Organization Assignment 2
<b>5</b>	Assembly Level Machine Organization
<b>6</b>	Revision and <b>In-Course Exam 1</b>
<b>7</b>	Memory System Organization and Architecture
<b>8</b>	Memory System Organization and Architecture Assignment 3
<b>9</b>	Interfacing and Communication
<b>10</b>	Interfacing and Communication
<b>11</b>	Functional Organization & Multiprocessing and Alternative Architectures <b>In-Course Exam 2</b>
<b>12</b>	Performance Enhancements & The Six “Great” Ideas
<b>13</b>	Revision

**COURSE TITLE: Computer Networks**  
**COURSE CODE: COMP 2602**  
**TYPE: Core**  
**LEVEL: 2**  
**SEMESTER: 1**  
**START DATE: SEPT-01-2016**  
**DEPARTMENT and FACULTY: DCIT/FST**  
**CREDITS: 3**  
**PRE-REQUISITE(S): COMP 1600**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 1-hour tutorial, 4 hours per week independent study

**COURSE DESCRIPTION**

This course examines some of the important concepts related to computer networks, e.g., the network edge and core, routers, the ISO and TCP/IP reference models for computer communication and networking protocols. Many use the Internet and local area networks every day but are not fully aware as to what goes on “behind-the-scenes” to make this network communication possible. In COMP2604, students explore what happens to the data in the computer before it is prepared for transmission, how protocols work to transmit the data and how it is received at other computers. Error control and recovery methods for lost or corrupted data are also investigated. A layered model for computer communications is thoroughly examined. Students will write networking programs and test them on a local area network or on the Internet.

**COURSE RATIONALE**

This course is important because students need to understand the underlying principles behind the development of today’s networks, especially the Internet.

**COURSE CONTENT**

1. Computer Networks and the Internet: The Internet. Network edge and core. Network access and physical media. Protocol layers and their Service models.
2. The Application Layer: Principles of application layer protocols. FTP, Email, SMTP, DNS etc. Socket programming with TCP and UDP.
3. The Transport Layer: Transport-layer services. Multiplexing and demultiplexing. UDP and TCP. Reliability. Congestion control.
4. The Network Layer and Routing: Service models. Routing. IP. Mobility.
5. Link Layer and Local Area Networks: Services. Error detection and correction. Multiple access protocols. Ethernet. Network hardware. Wireless links. PPP. Frame Relay.
6. Introduction to Network Design: The network design and implementation process. Stages: Feasibility Study, preparing network design plan, understanding current network, defining new network requirements, identifying geographic scope, calculating circuit requirements, identifying security and control measures, designing network configurations, determining network costs, network implementation.

## COURSE LEARNING OUTCOMES

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

1. Describe the components and infrastructure that form the basis of most computer networks
2. Explain the technical aspects of data communications on the Internet
3. Write networking programs in an appropriate programming language.
4. Propose network designs based on case studies in organizations
5. Design finite-state machine (FSM) diagrams for high level network error checking and recovery protocols
6. Demonstrate inter-personal skills, teamwork, and effective use of appropriate technology associated with field of computer studies

## TEACHING STRATEGIES

The course will be taught via interactive lectures, tutorials and group work. Students will be expected to work in groups as well as individually on course assignments and projects.

## COURSE ASSESSMENT

Assessment	Learning Outcomes						Weighting %	Assessment Description	Duration
	1	2	3	4	5	6			
Assignment 1	X	X	X				20%	Problems & short answer	
Assignment 2	X	X	X	X				Problems & short answer	
Group Project	X	X	X	X	X	X		Problems & short answer	
Exam 1	X	X	X	X	X		30%	Problems & short answer	1.5 hour
Final Examination	X	X	X	X	X			Problems & Short Answer Questions	2 hours
TOTAL %							100%		

### Required Reading

- Computer Networking A Top-Down Approach featuring the Internet, J. Kurose and K. Ross. Publisher: Addison Wesley. 6th edition (2012). International edition. ISBN-10: 0273768964  
ISBN-13: 978-0273768968

Slides and other resources from the Kurose Ross Text (Access Code needed).

<http://www.pearsoninternationaleditions.com/Sitemap/kurose-ross/>

- Computer Networks 5th edition (2010). Andrew S. Tanenbaum and David J. Wetherall. Publisher: Prentice Hall. ISBN-10: 0132126958. ISBN-13: 978-0132126953

### Other Readings

- Data and Computer Communications by William Stallings. 10th edition (2013). Publisher: Prentice Hall. ISBN-13: 978-0133506488

- Fundamental Networking in Java by Esmond Pitt (2010). Publisher: Springer. ISBN-10: 1849965455. ISBN-13: 978-1849965453.

Internet RFCs and other online technical documents

## COURSE CALENDAR

Week	Content
1	Computer Networks and the Internet: The Internet. Network edge and core. Network access and physical media.
2	Protocol layers and their Service models.
3	The Application Layer: Principles of application layer protocols. FTP, Email, SMTP. DNS.
4	Socket programming with TCP and UDP. Programming labs. Assignment 1 Group Project given
5	The Transport Layer: Transport-layer services. Multiplexing and demultiplexing. UDP and TCP.
6	UDP and TCP continued. Reliability. Congestion control.
7	The Network layer and Routing: Service models. Routing. IP. Mobility. <b>In-course exam 1</b>
8	Link Layer and Local Area Networks: Services. Error detection and correction. Multiple access protocols. Assignment 2
9	Ethernet. Network hardware. Wireless links. PPP. Frame Relay.
10	Introduction to Network Design: The network design process. The network design and implementation process. Stages: Feasibility Study, preparing network design plan, understanding current network, defining new network requirements.
11	Identifying geographic scope, calculating circuit requirements, identifying security and control measures, designing network configurations.
12	Determining network costs, network implementation. <b>Group Project presentation.</b>
13	Revision.

**COURSE TITLE: Object-Oriented Programming I**

**COURSE CODE: COMP 2603**

**TYPE: Core**

**LEVEL: 2**

**SEMESTER: 2**

**START DATE: JAN-15-2017**

**DEPARTMENT and FACULTY: DCIT/FST**

**CREDITS: 3**

**PRE-REQUISITE(S): COMP 1603**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 2-hour lab, 6 hours per week independent study

**COURSE DESCRIPTION**

This course provides a comprehensive introduction to the concepts and techniques of object-oriented programming. This course introduces the concepts of object-oriented programming to students with a background in the procedural paradigm.

**RATIONALE**

The course will give the students a detailed introduction to Object-Oriented design. Students will be taught different object-oriented concepts and rules such as classes, methods, encapsulation, interfaces, inheritance and composition. Students will also learn how to develop user interfaces using an object-oriented toolkit (e.g. swing in Java). At the end of the course, students will be able to develop object-oriented programs.

**COURSE CONTENT**

1. Object-Oriented Design
2. Decomposition into objects carrying state and having behavior
3. Classes and Methods: Encapsulation, Varieties of Classes, Interface and Implementation, Classes and Method in Java, Class Variables and Class Methods.
4. Instances, Initialization and Messages: Instance Creation and Initialization, Message-Passing Syntax
5. Inheritance and Composition: Subclass, Subtype, and Substitutability, Replacement and Refinement, Assignment, Equality, and Type Conversion, Polymorphism, Abstract Classes.
6. Introduction to Developing User Interfaces
7. Introduction to a Collection Framework (e.g., LinkedList, ArrayList, HashSet, TreeSet, HashMap, TreeMap, Comparators, and Generics in Java). Choosing the Right Collection to Use
8. Object Persistence: using flat files

**COURSE LEARNING OUTCOMES**

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

1. Describe the fundamental concepts and vocabulary of the object-oriented approach
2. Analyze a real-world situation in an object-oriented way
3. Design a simple object-oriented model containing multiple classes and collaborations
4. Implement a simple object-oriented model in an appropriate object oriented programming language

5. Develop graphical user interfaces using objects from a programming toolkit

## COURSE ASSESSMENT

Assessment	Learning Outcomes					Weighting %	Assessment Description	Duration
	1	2	3	4	5			
Assignment 1	X	X	X	X		20%	Problems & short answer	
Assignment 2	X	X	X	X			Problems & short answer s	
Assignment 3	X	X	X	X	X		Problems & short answer	
Exam 1	X	X	X	X		30%	Problems & short answer	1.5 hour
Exam 2	X	X	X	X			Problems & short answer	1.5 hour
Final Examination	X	X	X	X		50%	Problems & Short Answer Questions	2 hours
TOTAL %						100%		

## TEACHING STRATEGIES

Interactive lectures will be used to introduce the main concepts of the course. However, discussion and brain storming methods will be used throughout the course with the goal of allowing students to engage the problem solving process. Labs will also be used to demonstrate, in a practical manner, the concepts taught in lectures.

## RESOURCES

### Lecture notes

### Textbooks

Mohan, Permanand. *Fundamentals of Object-Oriented Programming in Java*, CreateSpace Independent Publishing Platform. 1<sup>st</sup> Edition (February 28, 2013). ISBN-10: 1482587521, ISBN-13: 978-1482587524.

Harvey Deitel and Paul Deitel (2002). *Java: How to Program*, 5th Edition. Prentice-Hall.

Bruce Eckel (2002). *Thinking in Java*, 3rd Edition. Prentice-Hall. Free electronic copy available on the Internet from <http://www.mindview.net/Books/TIJ/>.

Timothy Budd (1999). *Understanding Object-Oriented Programming With Java*, Updated Edition. Addison Wesley.

## COURSE CALENDAR:

Week	Topic
1-2	Encapsulation, Varieties of Classes, Interface and Implementation in Java Instance Creation and Initialization, Message-Passing Syntax Week 2 - Assignment 1



3-4	Inheritance and Composition; Polymorphism, Abstract classes Week 4 - Assignment 2
5-6	Object-Oriented Software Architectures and Object-Oriented Design Examination 1
7-8	Introduction to Developing User Interfaces Assignment 3 (Week 8)
9-11	Introduction to a Collections Framework Week 10 – Examination 2
12	Object persistence
13	Revision

**COURSE TITLE: Operating Systems**  
**COURSE CODE: COMP 2604**  
**TYPE: Core**  
**LEVEL: 2**  
**SEMESTER: 2**  
**START DATE: JAN-15-2017**  
**DEPARTMENT and FACULTY: DCIT/FST**  
**CREDITS: 3**  
**PRE-REQUISITE(S): COMP 1600**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 2-hour labs, 5 hours per week independent study

**COURSE DESCRIPTION**

This course looks at the inner workings of operating systems such as Windows, Ubuntu, and Mac OS X, both from a theoretical algorithmic point of view as well as a practical system programming point of view. The student will be introduced to the fundamental algorithms that support the existence of contemporary operating systems. Topics include the important areas of processes, threads, and CPU management, main and virtual memory management, file systems, disk scheduling algorithms, protection and security.

**COURSE RATIONALE**

An operating system acts as the interface between software applications and computer hardware and therefore is an important of Computer Science. This course presents important topics of operating systems such as structures of computer hardware and OSs, processes, threads, concurrency, CPU scheduling algorithms, primary and virtual memory, page replacement algorithms, file systems, disk scheduling algorithms, protection, and security. The course prepares students for other courses such as computer networks and distributed systems.

**COURSE CONTENT**

1. Introduction to Operating Systems
2. Operating System Structures
3. Process Management
4. Thread Management
5. CPU Scheduling
6. Process Synchronization (Concurrency)
7. Main Memory Management
8. Virtual Memory Management
9. Storage Management and Disk Scheduling
10. Protection and Security

## COURSE LEARNING OUTCOMES

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

1. Describe how an operating system manages multiple processes on a single-CPU system.
2. Analyze the procedure involved in context switching.
3. Explain the benefits and disadvantages of inter-process communication (shared memory and message passing).
4. Analyze available solutions for how deadlocks can arise in a system
5. Examine virtual memory management systems using standard and inverted page tables.
6. Examine a variety of page replacement algorithms.
7. Describe the structure of a hard disk and the basic file system formats.
8. Describe how access to resources is controlled by the operating system.
9. Write programs to create, control and destroy processes and threads.
10. Write programs that use semaphores for critical region control, as well as for synchronization
11. Write programs for inter-process communication (IPC) using shared memory and message passing on UNIX systems, Win32 API, and Java API.

## COURSE ASSESSMENT

Assessment	Learning Outcomes											Weighting %	Assessment Description	Duration
	1	2	3	4	5	6	7	8	9	10	11			
Assignment 1	X	X										20%	Short answer	
Assignment 2	X	X	X	X	X	X	X	X	X	X			Short answer	
Assignment 3	X	X	X	X	X	X	X	X	X	X				
Exam 1	X	X	X	X	X	X	X	X	X	X		30%	Problems and short answer	1.5 hour
Exam 2	X	X	X	X	X	X	X	X	X	X	X		Problems and short answer	1.5 hour
Final Examination	X	X	X	X	X	X	X	X	X	X	X	50%	Short Answer and Questions	2 hours
TOTAL %												100%		

## TEACHING STRATEGIES

The course will be delivered via interactive lectures and lab demonstrations.

## RESOURCES

Lecture notes

Textbooks

Abraham Silberschatz, Peter B. Galvin, Greg Gagne. 2013. *Operating System Concepts Essentials*, 2nd Ed, Wiley. ISBN: 1118804929.

William Stallings. 2014. *Operating Systems: Internals and Design Principles*, 8th Ed, Prentice Hall. ISBN: 0133805913.

Andrew S. Tanenbaum, Herbert Bos. 2014. *Modern Operating Systems*, 4th Ed, Prentice Hall. ISBN: 013359162X.

## COURSE CALENDAR

Week	Topic
1	Introduction to Operating Systems
2	Operating System Structures
3	Processes Assignment 1
4	Threads
5	CPU Scheduling
6	Process Synchronization Assignment 2
7	Main Memory Coursework exam
8	Virtual Memory
9	File Systems
10	Mass-storage Structure (Disk Scheduling) Assignment 3
11	Protection Coursework exam
12	Security
13	Revision

**COURSE TITLE: Enterprise Database Systems**

**COURSE CODE: COMP 2605**

**TYPE: Core**

**LEVEL: 2**

**SEMESTER: 2**

**START DATE: JAN-15-2016**

**DEPARTMENT and FACULTY: DCIT/FST**

**CREDITS: 3**

**PRE-REQUISITE(S): COMP 1602**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 2-hour lab, 6 hours per week independent study

**COURSE DESCRIPTION**

This course covers the design and implementation of relational database systems. Emphasis is placed on the database design of real world business application using Entity Relationship modeling. SQL programming is covered in detail. Data Management concepts such as Transaction Management, Concurrency Control, Recovery, and backups are presented. XML-enabled databases are also studied. An overview of several specialized databases is introduced and the technical and managerial responsibilities of a database administrator are discussed.

By utilizing an abundance of real world business applications, students are introduced to database systems and designs used by organizations. Additionally students examine the characteristics of database transactions and how they affect database integrity and consistency. At the end of this course, students will be able to effectively design and implement enterprise database systems.

**COURSE RATIONALE**

Information derived from data is very important and useful to the operational, managerial and strategic aspects of an organization. The data generated by the organization must be properly organized and managed to be useful to the organization. It is the responsibility of IT professionals to develop and maintain information database systems within an organization.

**COURSE CONTENT**

1. Introduction to the Relational Database System
  - 1.1. Components of the relational model
  - 1.2. Relational Data Objects
  - 1.3. Relation Data Integrity (Null, Candidate Keys, Primary Keys, Foreign Keys)
  - 1.4. Relational Operators (Restrict, Join, Project, etc.)
2. Database Design
  - 2.1. Identification of Business Rules
  - 2.2. Entity Relationship Modeling
  - 2.3. Functional Dependencies
  - 2.4. Normalization
3. Structured Query Language (SQL)
  - 3.1. Data Definition Language (Creating and Populating Relational Databases)
  - 3.2. Data Manipulation Language
  - 3.3. Simple Queries, Null Functions, Character and Numeric Functions

- 3.4. Complex Queries, Aggregate Functions, Sub-Queries
- 3.5. ANSI Standard SQL
- 3.6. Sub Queries
- 4. XML and Databases
  - 4.1. Introduction to XML
  - 4.2. XML capabilities of Relational databases
  - 4.3. SQL features for XML
- 5. Transaction Management and Recovery
  - 5.1. Properties of Database Transactions
  - 5.2. Database Transactions Concepts
  - 5.3. Transaction Log
  - 5.4. Recovery and Transaction and System Failure
  - 5.5. Concurrency Control (Database Locks)
- 6. Special Purpose Databases
  - 6.1. Overview of Spatial, Temporal and Decision Support Databases
  - 6.2. Databases and the Internet
- 7. Database Administration
  - 7.1. Backup and Recovery Principles
  - 7.2. Technical and Managerial DBA responsibilities
  - 7.3. Introduction to Query Processing

## COURSE LEARNING OUTCOMES

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

1. Apply data modeling tools in the design of a database
2. Apply Structure Query Language (SQL) code to implement and manage a database
3. Characterize how databases can be affected by real-world transactions.
4. Compare the different types of specialized databases
5. Summarize the managerial and technical roles of a Database Administrator

## COURSE ASSESSMENT

Assessment	Learning Outcomes					Weighting %	Assessment Description	Duration
	1	2	3	4	5			
Assignment 1	X	X	X			15%	Problems & short answer	
Assignment 2	X	X	X	X	X		Problems & short answer	
Exam 1 (Theory)	X	X	X	X	X	20%	Problems & short answer	1.5 hour
Exam 2 (Practical)	X	X	X	X		15%	Problems & short answer	1.5 hour
Final Examination	X	X	X	X	X	50%	Problems & Short Answer Questions	2 hours
TOTAL %						100%		

## TEACHING STRATEGIES

Interactive lectures will be used to introduce the main theoretical concepts of the course and practical laboratory sessions will be used to introduce SQL. Case-studies will be used to introduce business scenarios allowing students to apply course content in solving database problems.

## RESOURCES

### Textbooks

Peter Rob, Carlos Coronel et. Al. Database Systems Design, Implementation and Management, 11<sup>th</sup> edition. 2014. Boston, MA: Course Technology.

### Additional

Oracle Academy Resources on Database Design and SQL Programming  
Lecturer Notes and Handouts

## COURSE CALENDAR: 39 hours

Each week comprises of 2 lecture hours and 2 practical lab hours (2 hr. lab is substituted for 1 hr. tutorial)

Week	Topic
1	Relational Model
	Lab 1: Introduction to SQL
2	Relational Operators
	Lab 2: Data Definition Language
3	Database Design, ER Modeling
	Lab 3: Simple Queries, Null, Number and Character Functions
	Assignment 1
4	ER Modeling
	Lab 4: Complex Queries
5	ER Modeling, Functional Dependencies
	Lab 5: Sub-Queries
6	Normalization
	Lab 6: Visio ER Modeling
7	CW Examination (Theory)
	Lab 7: Review Lab
8	XML and Databases
	Lab 8: XML
	Assignment 2
9	Transaction Management
	Practice Lab Examination
10	Transaction Recovery, Concurrency Control
	Practical Examination (Lab)
11	Database Connectivity and Special Purpose Databases
12	Database Administration
13	Revision

**COURSE TITLE: Software Engineering I**  
**COURSE CODE: COMP 2606**  
**TYPE: Core**  
**LEVEL: 2**  
**SEMESTER: 2**  
**START DATE: JAN-15-2017**  
**DEPARTMENT and FACULTY: DCIT/FST**  
**CREDITS: 3**  
**PRE-REQUISITE(S): COMP 1603**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 2-hour lab, 6 hours per week independent study

**COURSE DESCRIPTION**

The specification, development, management, and evolution of software systems make up the discipline of software engineering. In this course, students apply methods and tools to develop software designs and specifications. The course focuses on universal techniques for developing large-scale systems rather than individual algorithms. In order to build good business systems, it is particularly important that the student place a great deal of emphasis in exploring the different process models and the topics covering requirements analysis and system specification, system architecture and design, verification and validation and system evolution. During the course, students will participate in a real problem solving/software development project which will expose them to the processes, tools and techniques of professional product-quality software development.

**COURSE RATIONALE**

Software engineering is a practical exercise that requires an understanding of how large-scale system development differs from small-scale programming and algorithm design. The course exposes students to issues related to the design, development and management of software products. It provides the foundation for making informed decisions about what methodology and tools to use in order to create robust and well-tested systems.

**COURSE CONTENT**

1. Requirements Engineering (elicitation, analysis and specifications)
2. Software Process Models (waterfall, incremental, agile)
3. Object Oriented Software Design and UML
4. Product and Project Management (Cost Estimation, Risk Management)
5. Testing, verification and validation (Test-Driven Development (TDD), Test plans, test strategies)

**COURSE LEARNING OUTCOMES**

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

1. Illustrate how software can interact with and participate in various systems including information management, embedded, process control and communication systems.
2. Develop project proposals that include comprehensive requirements based upon organization, client and/or system needs.
3. Develop comprehensive requirements and project proposals based on organization, client and/or system needs.
4. Document the differences between software process models including the advantages and disadvantages of each one.



5. Describe how general software engineering methodologies are applied in specific domains, with a particular focus on dependable systems.
6. Derive the necessary trade-offs in developing 'fit for purpose' systems through the application of evaluation metrics.
7. Develop strategies to manage identified risks, manage and motivate teams, roles motivation and conflict.
8. Demonstrate inter-personal skills, teamwork, and effective use of appropriate technology associated with the field of computer studies

## COURSE ASSESSMENT

The assessment strategy for this course is based upon building project-management and communication skills within a team environment. The course assessment strategy also addresses the need to create critical and innovative problem-solvers.

Assessment	Learning Outcomes								Weighting %	Assessment Description	Duration
	1	2	3	4	5	6	7	8			
Assignment 1	X	X	X						20%	Problems & short answer	
Assignment 2	X	X	X	X	X	X				Problems & short answer	
Group Project	X	X	X	X	X	X	X	X	30%	Rubric Based	
Final Examination	X	X	X	X	X	X	X		50%	Problems & Short Answer Questions	2 hours
TOTAL %									100%		

## TEACHING STRATEGIES

A project-intensive methodology for teaching software engineering is applied. The class work involved in this course directly complements the project work. Theoretical concepts will be introduced through case-studies. Students will work in groups on the development of a realistic software requirement specification, system design, testing plans, project cost and risk management strategies for an identified software project.

Assignments are designed to promote the analysis of theoretical concepts discussed in class. It will further provide students the opportunity to demonstrate a theoretical understanding of concepts identified.

### Textbook

Sommerville Ian A., Software Engineering 10th edition (2015), Pearson

## COURSE CALENDAR

Week	Topic
Week 1	Introducing software engineering and processes as the realization of software engineering's systematic approach to developing software. Group Project Given
Week 2,3	Software Process Models
Week 4,5,6	Requirements Engineering Assignment 1 (Week 4)
Week 7,8	OOP and Software Design

Week 9,10	Verification and Validation: Focus on Testing, TDD Assignment 2 (Week 9)
Week 11	Introduction Project Management
Week 12	Risk Management & Cost Estimation
Week 13	Revision and Project Presentations



**COURSE TITLE: Software Engineering II**  
**COURSE CODE: COMP 3600**  
**TYPE: Core**  
**LEVEL: 3**  
**SEMESTER: 1**  
**START DATE: SEPT-01-2016**  
**DEPARTMENT and FACULTY: DCIT/FST**  
**CREDITS: 3**  
**PRE-REQUISITE(S): COMP 2606**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 1-hour tutorial, 5 hours per week independent study

**COURSE DESCRIPTION**

This course is a continuation of developing skills surrounding software engineering, its principles and practical applications within the computer science curriculum. This course will expose students to the required engineering rigors of specifying, designing, developing and maintaining product-quality code. It will prepare them for the challenge of developing software systems as part of a team through a better understanding of development process methodologies, and an appreciation of the different challenges software engineers face in domains as varied as web-based systems, mission-critical systems and safety-critical systems.

**COURSE RATIONALE**

The ability to develop software utilizing agile methodologies is an integral part of the field of computer science or software engineering. Thus, this course explores advanced topics in software engineering, while ensuring that the students participate as developers and project managers during the software development cycle.

**COURSE CONTENT**

1. Software Engineering tools (version control, testing tools, continuous integration)
2. Project Management (Approaches, Planning and Resource scheduling, Effort Estimation, tools)
3. Coding practices (Defensive and secure practices, coding standards, integration strategies, software reliability/dependability)
4. Software Evolution (Software engineering in pre-existing code bases, software uses)

**COURSE LEARNING OUTCOMES**

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

1. Utilize an agile based methodology to create a sufficiently complex software product based on developed requirements.
2. Explain the impact of identified software evolution issues associated on the software lifecycle.
3. Explain the problems that exist in achieving very high levels of reliability
4. Apply techniques, coding idioms and mechanism when implementing software designs so that reliability, efficiency and robustness is achieved.
5. Build robust code using exception handling mechanisms
6. Utilize a selected coding standard in a small software project.
7. Demonstrate how version controls can be used to manage software releases through the application of a source code control tool in a small team-based project.

8.—Develop communication skills through written material and oral presentations

## COURSE ASSESSMENT

Assessment	Learning Outcomes								Weighting %	Assessment Description	Duration
	1	2	3	4	5	6	7	8			
Assignment 1	X	X	X						10	Research	
Group Project	X	X	X	X	X	X	X		40	Documentation/Code	
Project Presentation	X	X	X	X	X	X	X	X	25	Rubric Based	
Individual Project Report								X	25	Rubric Based	
TOTAL %											

## TEACHING STRATEGIES

A project-intensive methodology for teaching software engineering is applied. The class work involved in this course directly complements the project work. Theoretical concepts are introduced through the case-studies. Students will work in groups on a software development project.

## TEXTBOOK

Sommerville Ian A., Software Engineering 10<sup>th</sup> edition (2015), Pearson

## COURSE CALENDAR

Week	Topic
1	Introduction & Review
2	Introduction to Agile Methodology (used for project)
3	Software Engineering Tools Project Given
4,5,6	Project Management Assignment 1 (Week 4)
7,8,9	Coding Practices
10,11,12	Software Evolution Project Report (Individual + Group)
13	Project Presentations

**COURSE TITLE: Design and Analysis of Algorithms**

**COURSE CODE: COMP 3601**

**TYPE: Core (Special Degree Only)**

**LEVEL: 3**

**SEMESTER: 1**

**START DATE: SEPT-01-2016**

**DEPARTMENT and FACULTY: DCIT/FST**

**CREDITS: 3**

**PRE-REQUISITE(S): COMP 2600**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 1-hour tutorial, 6 hours per week independent study

**COURSE DESCRIPTION**

This course covers specific fundamental algorithm-design techniques used to formulate the solutions to a wide variety of problems. It also covers problem-solving techniques for analyzing algorithms to determine space/time requirements.

**COURSE RATIONALE**

At a time when computers are playing an ever-increasing role in our daily lives, there is a never-ending search for better algorithms to solve problems. ‘Better’ can mean faster, use less storage, use storage more efficiently or solve a more general class of problems.

**COURSE CONTENT**

1. Analyze algorithms for time and space bounds. Growth of functions.
2. Asymptotic notation. Formulate the running time of an algorithm as a recurrence relation. Solve recurrence relations using substitution, iteration and master method.
3. Review and analysis of data structures: stacks, queues, linked lists, hash tables, binary search trees, graphs, spanning trees.
4. Review and analysis of sorting methods: insertion sort, merge sort, heapsort, quicksort.
5. Algorithm design techniques. Brute force. Dynamic programming. Greedy algorithms. Divide-and-conquer algorithms. Graph algorithms, including all-pairs shortest paths. Back-tracking algorithms. String matching algorithms. Approximation algorithms. Examples of problems which can be solved using each of these techniques.
6. Write programs which employ any or all of these techniques.

**COURSE LEARNING OUTCOMES**

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

1. Identify key issues in algorithm design
2. Differentiate between algorithms for solving the same problem
3. Write algorithms associated with advanced data structures
4. Write algorithms using several algorithm-design techniques
5. Analyze the space/time bounds of various algorithms
6. Write programs to solve problems requiring the use of all techniques presented in this course.

## TEACHING STRATEGIES

A combination of lectures/tutorials will be used. Feedback from assignments/problem sheets/course work tests will be used to tailor subsequent lectures/tutorials. Students will be given ample opportunity to develop correct, readable programs using structured programming techniques.

## COURSE ASSESSMENT

Assessment	Learning Outcomes						Weighting %	Assessment Description	Duration
	1	2	3	4	5	6			
Assignment 1	X						20%	Problems	
Assignment 2	X	X	X	X	X			Problems	
Assignment 3	X	X	X	X	X	X		Problems	
Exam 1	X	X	X	X	X		30%	Problems & short answer	1.5 hour
Exam 2	X	X	X	X	X	X		Problems & short answer	1.5 hour
Final Examination	X	X	X	X	X	X	50%	Problems & Short Answer Questions	2 hours
TOTAL %							100%		

## RESOURCES

### Lecture notes (in-class and online)

### Textbooks

Levitin, Wesley, The Design and Analysis of Algorithms, 2nd Edition, 2007. ISBN:0-321-36413-9

Cormen, Thomas H. Introduction to algorithms. MIT press, 2009.

Alsuwaiyel, M. H. "Design techniques and analysis." Algorithms 10 (1999): World Scientific Publishing.

## COURSE CALENDAR

Week	Content
1	Review of some familiar standard algorithms with the emphasis on space/time requirements. Introduction to big-O notation.
2	Review of basic data structures and associated algorithms Assignment 1
3-4	Discussion of problems that illustrate the differences between brute-force and more efficient approaches e.g., 'triangle' problem to compare brute-force (all paths), straight recursion and memorized recursion. Assignment 2 (Week 4)
5-6	Formal treatment of analysis of algorithms; definitions, recurrences, mathematical background, theorems

	Course Work Exam (Week 6)
7-8	Dynamic programming, brute-force and recursive solutions to standard problems, e.g., knapsack, coin-change, matrix chain multiplication, all-pairs shortest paths, longest increasing subsequence, longest common subsequence Assignment 3 (Week 8)
9-10	Greedy algorithms e.g., Huffman coding, Prim's, Kruskal's, Dijkstra's. Representation of disjoint subsets. Course Work Exam (Week 10)
11-12	Backtracking algorithms. String matching algorithms, e.g. Knuth-Morris-Pratt (KMP).
13	Revision



**COURSE TITLE:** Theory of Computing  
**COURSE CODE:** COMP 3602  
**TYPE:** Core (Special Degree Only)  
**LEVEL:** 3  
**SEMESTER:** 1  
**START DATE:** JAN-15-2017  
**DEPARTMENT and FACULTY:** DCIT/FST  
**CREDITS:** 3  
**PRE-REQUISITE(S):** COMP 2600

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 1-hour tutorial, 6 hours per week independent study

**COURSE DESCRIPTION**

The course introduces undergraduate computer science students to the foundations of theoretical computer science. It exposes them to abstractions which can be used to solve complex real world problems. It introduces: Regular Languages, Finite Automata, Context-free Languages, Computability; Turing machines and Complexity Classes. Finally, students gain an appreciation for theoretical aspects of computing and the basic skills required to assess the limitations of the computer.

**COURSE RATIONALE**

Part of the success of computer science and of the field of computing is due to the existence of a solid theoretical foundation. This course introduces the fundamental ideas of this theoretical foundation. Here, the computer is treated as an abstract mathematical entity, and not as a physical object.

**COURSE CONTENT**

1. Alphabet and languages
2. Finite automata and Regular languages
3. Deterministic and non-deterministic finite automata
4. Regular expressions
5. Context-free grammars
6. Context-free languages and pushdown automata
7. Proving languages non-regular
8. Turing machines
9. Computability
10. Uncomputability
11. Complexity Classes
12. Classic NP-complete problems
13. Reduction Techniques

**COURSE LEARNING OUTCOMES**

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

1. Demonstrate the concept of finite state machines.
2. Define the P and NP classes, the P-space class and its relation to the EXP class.
3. Illustrate why the halting problem has no algorithmic solution.
4. Assess the significance of: NP-completeness, the Church-Turing thesis and Rice's Theorem.
5. Provide examples of uncomputable functions and classic NP-complete problems.

6. Design a deterministic finite state machine, a context-free grammar and generate a regular expressions for a specified language.
7. Prove a problem is either NP-complete or uncomputable by reducing it to know classical problems.
8. Determine a language's place in the Chomsky hierarchy.
9. Convert between notations for a language (e.g. PDAs, CFGs, DFAs, NFAs, and regular expressions).

## COURSE ASSESSMENT

Assessment	Learning Outcomes									Weighting %	Assessment Description	Duration
	1	2	3	4	5	6	7	8	9			
Assignment 1	X	X								20%	Problems	
Assignment 2	X	X	X	X	X						Problems	
Assignment 3	X	X	X	X	X	X	X				Problems	
Exam 1	X	X	X	X	X					30%	Problems & short answer	1.5 hour
Exam 2	X	X	X	X	X	X	X				Problems & short answer	1.5 hour
Final Examination	X	X	X	X	X	X	X	X	X	50%	Problems & Short Answer Questions	2 hours
TOTAL %										100%		

## TEACHING STRATEGIES

A combination of lectures/tutorials will be used. Feedback from assignments/problem sheets/course work tests will be used to tailor subsequent lectures/tutorials. Students will be given ample opportunity to develop correct, readable programs using structured programming techniques.

## RESOURCES

### Textbooks

Webber, Adam: *Formal language: a practical introduction*. 2008. Franklin, Beedle & Associates, Incorporated, ISBN-13: 978-1-59028-197-0

Sipser, Michael. *Introduction to the theory of Computation*. 2006. Thompson Course Technology. ISBN-13: 978-1133187790

## COURSE CALENDAR:

Week	Topic
Week 1 = 3 hrs	Alphabet and languages
Week 1,2 = 3 hrs	Finite automata and Regular languages Assignment 1 (Week 2)
Week 3,4 = 6hrs	Deterministic and non-deterministic finite automata Assignment 2 (Week 4)
Week 5, 6, 7 = 9 hrs	Regular expressions, Context-free grammars, Context-free languages and pushdown automata. Proving languages non-regular.
Week 7 = 3hr	Coursework Examination 1

Week 8, 9, 10	Turing machines, Computability and Uncomputability Assignment 3 (Week 9)
Week 11, 12	Complexity Classes, Classic NP-complete problems and Reduction Techniques Coursework Examination 2 (Week 11)
Week 13	Revision

**COURSE TITLE: Human-Computer Interaction**

**COURSE CODE: COMP 3603**

**TYPE: Core (Special Degree Only)**

**LEVEL: 3**

**SEMESTER: 2**

**START DATE: JAN-15-2017**

**DEPARTMENT and FACULTY: DCIT/FST**

**CREDITS: 3**

**PRE-REQUISITE(S): COMP 2606**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 1-hour tutorial, 6 hours per week independent study

**COURSE DESCRIPTION**

Human-computer interaction is an interdisciplinary field that integrates theories and methodologies from computer science, cognitive psychology, design, and many other areas. The course is intended to introduce the student to the basic concepts of human-computer interaction. It will cover the basic theory and methods that exist in the field. The course will unfold by examining design and evaluation. Case studies are used throughout the readings to exemplify the methods presented and to lend a context to the issues discussed. The students will gain principles and skills for designing and evaluating interactive systems. The heart of the course is a semester-long group project that will help students learn in a hands-on way about the various stages of an effective design process.

The goal of this course is to help students realize that user interface development is an ongoing process throughout the full product life cycle, and developing the human-computer interface is not something to be done at the last minute, when the "rest of the system" is finished. Hence, this course concentrates on creating and testing DESIGNS of human-computer systems through low and medium fidelity prototypes and NOT with implementing a piece of software in this class.

**COURSE RATIONALE**

Human-computer interaction (HCI) has become an area of great interest and concern. HCI is concerned with the joint performance of tasks by humans and machines. It stresses the importance of good interfaces and the relationship of interface design to effective human interaction with computers. Specifically, we concentrate on so-called interactive systems. This course uses an integrative and cross-disciplinary approach to bring together a broad variety of topics together in relation to the problem of developing quality user interaction designs to provide an introduction to the field of HCI. This course is different from a majority of CS courses like software engineering that take a systems perspective. It focuses more on application (and less on theory) of user-centered design principles, guidelines, and evaluation. This course provides the concepts of HCI and user interfaces, focusing on user interface design, evaluation, and technologies. It is a fun course with interesting user interface design projects and concepts. Unlike a majority of other CS courses, this is NOT programming intensive.

**COURSE CONTENT:**

1. Introduction to human-computer interaction
2. User-centered design
  - 2.1. User-centered design development life cycle
  - 2.2. Interactive system mission and target user population

- 2.3. Techniques for data collection from users
- 2.4. Prototyping: low fidelity prototypes, medium fidelity prototypes, wizard of Oz
- 3. Needfinding
  - 3.1. Task analysis, including qualitative aspects of generating task analytic models
  - 3.2. Development of task examples
  - 3.3. Task-centered walkthroughs
- 4. Psychological foundations of HCI
  - 4.1. Physical capabilities that inform interaction design
  - 4.2. Cognitive models that inform interaction design
  - 4.3. Psychopathology of everyday things
  - 4.4. Concepts for designing everyday things
- 5. Approaches for human-computer interaction evaluation
  - 5.1. Goals of evaluation, approaches, ethics, introspection
  - 5.2. Measures for evaluation: utility, efficiency, learnability, user satisfaction
  - 5.3. Extracting the conceptual model
  - 5.4. Direct observation
  - 5.5. Constructive interaction
  - 5.6. Quantitative evaluation techniques
  - 5.7. Choosing evaluation approach
  - 5.8. Principles of usability testing
- 6. Design principles and guidelines
  - 6.1. Principles of good design
  - 6.2. Usability heuristics
  - 6.3. Golden rules
  - 6.4. Handling human/system failure
  - 6.5. Design rules
  - 6.6. HCI standards
- 7. Elements of visual design
  - 7.1. Characteristics of good visual design
  - 7.2. Information visualization
  - 7.3. Visual variables
  - 7.4. Metaphors
  - 7.5. Direct manipulation
  - 7.6. Color design
  - 7.7. Typography design
  - 7.8. Visual design guidelines
- 8. Graphical HCI design
  - 8.1. Principles of graphical user interfaces (GUIs)
  - 8.2. Components of visible language
  - 8.3. Graphical design by grids
- 9. Mobile human-computer interaction design
  - 9.1. Mobile interaction design process
  - 9.2. Principles of good mobile human-computer interaction design
  - 9.3. Mobile design rules and guidelines
- 10. User experience design (UXD)
  - 10.1. Foundations
  - 10.2. Personalized UXD
  - 10.3. Emotional UXD
- 11. HCI past and future

- 11.1. The past
- 11.2. Present
- 11.3. Future
- 11.4. New interaction techniques

## COURSE LEARNING OUTCOMES

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

1. Describe the basics of human and computational abilities and limitations.
2. Apply basic HCI theories, tools and techniques in solving authentic problems
3. Apply the fundamental aspects of designing and evaluating HCI interfaces in solving authentic problems
4. Evaluate the quality of a user interface through application of a variety of basic evaluation methods.
5. Design systems using appropriate HCI techniques
6. Determine the relationships between specific instances and broader generalizations
7. Apply concepts and principles to explain, analyze and solve specific situations implicit by the setting.
8. Design an interface for authentic tasks and users, incorporating direct manipulation in appropriate
9. Demonstrate effective written and oral communication techniques for evaluating and defending chosen HCI techniques.
10. Demonstrate strong inter-personal skills including teamwork during interactions with peers.

## COURSE ASSESSMENT:

**The assessment strategy for this course is based upon building project-management and communication skills within a team environment. The course assessment strategy also addresses the need to create critical and innovative problem-solvers.**

Assessment	Learning Outcomes										Weighting %	Assessment Description	Duration
	1	2	3	4	5	6	7	8	9	10			
Assignment 1: Project Proposal	X	X									7%	Rubric Based	
Assignment 2: Plan for Collecting Requirements	X	X									7%	Rubric Based	
Assignment 3: Requirements/low-fidelity prototype	X	X	X	X	X	X	X	X			7%	Rubric Based	
Assignment 4: High-fidelity prototype	X	X	X	X	X	X	X	X			14%	Rubric Based	
Assignment 5: Usability	X	X	X	X	X	X	X	X			14%	Rubric Based	
Assignment 6: Final Project	X	X	X	X	X	X	X	X	X		14%	Rubric Based	
Assignment 7: Presentations	X	X	X	X	X	X	X	X	X		7%	Rubric Based	
Team work/written communication and presentation									X	X	30%	Rubric Based	

TOTAL %											100%		
---------	--	--	--	--	--	--	--	--	--	--	------	--	--

## TEACHING STRATEGIES:

Lecture time will be used to introduce the main concepts of the course. Directed case studies will be used with the goal of allowing students to apply the methods taught in a scenario. Interactive discussion and blended learning methods will also be used throughout the course.

A project-intensive methodology for teaching user interface design is applied. To stress that the class work involved in this course directly complements the project work, the approach of project-intensive teaching is to go step by step through the process of preparing for the course, teaching the individual classes, and coordinating with the project pretty much as outlined in the syllabus. The activities needed at each step are described, along with support materials, assignments. Feedback, annotations and discussions of specific points are provided through each step of the project.

Students create their own teams of 2-4 for this project. The project is a considerable amount of work and requires a team to help manipulate the equipment or perform a portion of the human subject interactions.

## RESOURCES

### Essential Textbook

Dix A. et al., [Human-Computer Interaction](#). Harlow, England: Prentice Hall, 2004, ISBN-10: 0130461091

Russ Unger, Carolyn Chandler, [A Project Guide to UX Design: For user experience designers in the field or in the making](#), 2nd Edition, New Riders, 2012, ISBN-13:978-0321815385, ISBN-10:0321815386

### Recommended textbooks

Yvonne Rogers, Helen Sharp, Jenny Preece, Interaction Design: Beyond Human Computer Interaction, 3<sup>rd</sup> Edition, Wiley, 2011, ISBN-10: 0470665769

Lazar J.: Web Usability: A User-Centered Design Approach , Addison Wesley, 2005, ISBN:0321321359

## COURSE CALENDAR:

Week	Lecture Topic	Tutorial
1	Introduction to Human-Computer Interaction. Semester project and student teams	
2	User-centered design: User-centered design development life cycle; Interactive system mission and target user population; Techniques for data collection from users; Prototyping: low fidelity prototypes, medium fidelity prototypes, wizard of Oz	Hand out Assignments
3	Needfinding: Task analysis, including qualitative aspects of generating task analytic models; Development of task examples; Task-centered walkthroughs	Assist with Assignment 1
4	Psychological foundations of HCI: Physical capabilities that inform interaction design; Cognitive models that inform interaction design; Psychopathology of everyday things; Concepts for designing everyday things	<b>Assignment 1 due</b> Present and discuss Assignment 1

5	Approaches for human-computer interaction evaluation: Goals of evaluation, approaches, ethics, introspection; Measures for evaluation: utility, efficiency, learnability, user satisfaction; Extracting the conceptual model; Direct observation; Constructive interaction; Quantitative evaluation techniques; Choosing evaluation approach; Principles of usability testing	Assist with Assignment 2
6	Design principles and guidelines: Principles of good design; Usability heuristics; Golden rules; Handling human/system failure; Design rules; HCI standards	<b>Assignment 2 due</b> Present and discuss Assignment 2
7	Elements of visual design: Characteristics of good visual design; Information visualization , Visual variables; Metaphors; Direct manipulation; Color design; Typography design; Visual design guidelines	Assist with Assignment 3
8	Graphical HCI design: Principles of graphical user interfaces (GUIs); Components of visible language; Graphical design by grids	<b>Assignment 3 due</b> Present and discuss Assignment 3
9	Mobile human-computer interaction design: Mobile interaction design process; Principles of good mobile human-computer interaction design; Mobile design rules and guidelines	Assist with Assignment 4
10	User experience design (UXD): Foundations; Personalized UXD; Emotional UXD	<b>Assignment 4 due</b> Present and discuss Assignment 4
11	HCI past and future: The past; Present; Future; New interaction techniques	Assist with Assignment 5
12	Usability testing in Usability Lab	Usability testing
13	Presentation of final projects	<b>Assignment 5 due</b> Present and discuss Assignment 5 and final project



**COURSE TITLE: Introduction to Data Analytics**

**COURSE CODE: COMP 3605**

**TYPE: Elective**

**LEVEL: 3**

**SEMESTER: 1**

**START DATE: SEPT-01-2016**

**DEPARTMENT and FACULTY: DCIT/FST**

**CREDITS: 3**

**PRE-REQUISITE(S): MATH 2250**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 2-hour lab, 6 hours per week independent study

**COURSE DESCRIPTION:**

This course provides an introduction to various computational and data mining techniques that are used within the computer science discipline to facilitate intelligent decision making and analysis within systems. The course focuses on providing a practical understanding of a number of computational intelligence techniques without over-burdening the student with the theoretical foundation that many of these techniques possess. The course provides a foundational understanding of topics that will be useful for further work in data mining analysis and machine learning. The course will utilize an appropriate programming language (e.g., python) and available software tools (e.g., scikit-learn, scipy and pandas) to give students practical experience utilizing these algorithms to solve real world problems.

**COURSE RATIONALE:**

The proliferation of database systems has resulted in the generation of datasets of increasing complexity and size. To process and understand such datasets, an array of techniques and algorithms have been developed which facilitate the development of business solutions based on the data. This course aims at developing a student's ability to evaluate a dataset and use analysis techniques to develop business solutions and facilitate intelligent decision making based off data.

**COURSE CONTENT**

1. Data Preprocessing
  - 1.1. Describe the difference between Nominal, Binary, Ordinal and Numeric Attributes
  - 1.2. Describe and utilize various data cleaning techniques
  - 1.3. Introduction to Data Transformation
2. Association and Correlations
  - 2.1. Give an overview of association rules
  - 2.2. Discuss the application of market basket analysis
  - 2.3. Describe and utilize the Apriori algorithm
  - 2.4. Differentiate between Association and Correlation analysis
3. Classification
  - 3.1. Explain the fundamental concepts of supervised learning techniques
  - 3.2. Describe classification techniques and list a number of common classification techniques
  - 3.3. Describe and utilize the C4.5 decision tree classifier
  - 3.4. Describe and utilize Naive Bayesian classification algorithm
4. Cluster Analysis
  - 4.1. Explain the fundamental concepts of unsupervised learning
  - 4.2. Describe cluster analysis and list a number of common clustering algorithms

- 4.3. Describe and utilize the K-Means algorithm
- 4.4. Describe vector quantization and compare its performance and accuracy with the K-Means algorithm
- 4.5. Evaluate the performance and accuracy of clustering methods
5. Outlier Detection
  - 5.1. What are outliers
  - 5.2. List the types of outliers
  - 5.3. Discuss the difference between supervised and unsupervised methods for outlier detection
6. Support Vector Machines
  - 6.1. Explain the fundamental concepts of Vector Machines
  - 6.2. Utilize a dot kernel SVM for classification
  - 6.3. Describe using SVMs for regression analysis

## COURSE LEARNING OUTCOMES

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

1. Differentiate the different uses of data mining and computation methods in both research and application
2. Explain the value of finding association in market basket data
3. Evaluate different methodologies for effective application of data analytics/mining
4. Characterise identified sources of noise, redundancy and outliers present in data

## COURSE ASSESSMENT

Assessment	Learning Outcomes				Weighting %	Assessment Description	Duration
	1	2	3	4			
Assignment 1	X	X			10%	Problems	
Assignment 2	X	X	X	X	10%	Problems	
Lab Exam 1	X	X			20%	Problems	1.5 hour
Course work Exam	X	X	X	X	10%	Problems & short answer	1.5 hour
Final Examination	X	X	X	X	50%	Problems & Short Answer Questions	2 hours
TOTAL %					100%		

## TEACHING STRATEGIES

Lecture time will be used to introduce the main concepts of the course by using case studies and demonstrations. The focus of the course is to develop students competence utilizing the various algorithms discussed during the course. Therefore the labs sessions will be used to introduce the various techniques and will also provide case base examples for students to apply the knowledge acquired during the lectures and demonstrations. The assessment will be a balance between assessing students' competence in utilizing the specified technique and also his/her ability to evaluate which technique is most applicable for a given goal of analysis specified by a task.

## RESOURCES

**Lecture notes**

**Textbooks**

Han, Jiawei. Data Mining Concepts and Techniques (3rd Edition), Morgan Kaufmann (2011), ISBN: 978-0-12-381479-1 [Recommended Text]

Flach, Peter. Machine Learning: The Art and Science of Algorithm that make sense of data, Cambridge University Press (2012)

**COURSE CALENDAR:**

Week	Topic
1	Introduction
2	Data Preprocessing
3	Association and Correlations
4	Assignment 1
5, 6	Classification
7	Cluster Analysis
8	Assignment 2
9	Outlier Detection
10	Support Vector Machines Coursework Exam
11	Lab Exam
12	Support Vector Machines
13	Revision

**COURSE TITLE: Wireless and Mobile Computing**

**COURSE CODE: COMP 3606**

**TYPE: Elective**

**LEVEL: 3**

**SEMESTER: 1**

**START DATE: SEPT-01-2016**

**DEPARTMENT and FACULTY: DCIT/FST**

**CREDITS: 3**

**PRE-REQUISITE(S): COMP 2602**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 2-hour lab, 6 hours per week independent study

**COURSE DESCRIPTION**

This course is recommended as an essential part of the "Net-centric Computing" component of the ACM Computing curricula. It looks at the architecture of wireless networks and associated protocols. Software support for wireless and mobile computing is also examined. This includes Android Programming and SMS based applications. The course recognizes that software regimes may evolve over time and hence would examine at least one of the major relevant and applicable wireless programming languages available from time to time. Emerging technologies are also discussed.

**COURSE RATIONALE**

Wireless hosts e.g., mobile phones, tablets, laptops, palmtops, PDAs, desktop computers as well as wireless links are becoming increasingly popular, hence there is the need to investigate the principles and protocols that make wireless communication possible. Most of the major players that build devices are using wireless technology, so there is need for software support for these devices.

**COURSE CONTENT**

1. Wireless and Mobile Networks Theory
  - a. Why study wireless networks?
  - b. History and evolution of wireless standards
  - c. Special problems of wireless and mobile computing
  - d. Wireless LANs
  - e. Bluetooth
  - f. Cellular Networks
  - g. Mobile Internet Protocol
  - h. Software support for mobile and wireless computing (for mobile phones and/or other mobile devices)
2. Current Wireless Programming Language
3. Android
4. Introduction to Android
  - a. What is Android?
  - b. What devices does it run on?
  - c. Android Features.
  - d. Android Operating Systems.
  - e. Simple Android application and architecture.
  - f. Setting up the environment.

5. Creating Applications and Activities
  - a. Architecture of an application.
  - b. Activities and Life cycles.
  - c. Running an application.
  - d. Activity Practice.
6. Building User Interfaces
  - a. Widgets introduction. Popular widgets including EditText, TextView, Button, CheckBox, CompoundButton, RadioButton, RadioGroup,
    - i. Toast, AnalogClock, CalendarView, DatePicker, TextClock, ImageButton, ImageView, NumberPicker, ProgressBar, RatingBar, Spinner, ToggleButton, Layouts.
  - b. Utilizing popular widgets to develop simple to intermediate level applications.
7. Intents and Broadcast Receivers
  - a. Introduction to Intents.
  - b. Explicit Intents.
  - c. Implicit Intents
  - d. Programming practice with Explicit Intents.
  - e. Programming practice with Implicit Intents.
  - f. Programming practice with combination of Explicit and Implicit Intents.
8. Files, Saving State and Preferences
  - a. Files in Internal and External Storage.
  - b. Reading, writing and opening files from internal storage.
  - c. File applications.
9. Saving state and preferences within applications.
 

Hardware Sensors

  - a. Use of some popular sensors (subject to availability on phones).
10. Maps and Location-Based services
  - a. Maps.
  - b. Location-Based services.
11. SMS
  - a. SMS introduction.
  - b. Client/Server SMS architecture.
  - c. Client applications
  - d. Server applications.
12. Combined client/server applications.
 

Emerging Technologies

## **COURSE LEARNING OUTCOMES**

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

1. Implement Android-based applications
2. Develop applications that employ Intents and Broadcast Receivers
3. Discuss the usage of maps and location-based services for mobile software development
4. Use Maps and location-based services
5. Examine the role of SMS as a communication tool
6. Solve problems requiring the use of SMS

## TEACHING STRATEGIES

The course would be delivered using a combination of the following:  
Tutorial discussions, In-class discussions, Labs

## COURSE ASSESSMENT

Assessment	Learning Outcomes						Weighting %	Assessment Description	Duration
	1	2	3	4	5	6			
Assignment 1	X	X					10%	Problems	
Assignment 2	X	X					10%	Problems	
In-Class Test	X	X					15%	Problems	1.5 hour
In-Class Assignment	X	X	X	X	X		15%	Problems & short answer	1.5 hour
Final Examination	X	X	X	X	X	X	50%	Problems & Short Answer Questions	2 hours
TOTAL %							100%		

## RESOURCES

### Lecture Notes

### Textbooks

Professional Android 4 Application Development. Reto Meier. Edition: 3. Publisher: Wrox. Publication Date: May 1st, 2012. ISBN-10: 1118102274, ISBN-13: 978-1118102275.

Android for Programmers: An App-Driven Approach, 2nd Edition. Abbey Deitel, Harvey Deitel, Paul Deitel. Edition: 2. Publisher: Prentice Hall. Publication Date: 2014-01-06. ISBN-10: 0133570924, ISBN-13: 9780133570922.

Creating Android Applications, Development and Design, Chris Hasman. Peachpit Press. 1 edition (December 25, 2011). ISBN-10: 032178409X. ISBN-13: 978-0321784094.

## COURSE CALENDAR

Week	Topic
1, 2	Wireless and Mobile Networks Theory
3	Introduction to Android. Getting started with Android. Creating Applications and Activities Assignment 1 (take home)
4	Building User Interfaces
5	Intents and Broadcast Receivers Assignment 2 (take home)
6, 7	Files, Saving State and Preferences In-Class Test (Week 7)
8, 9	Hardware Sensors
10, 11	Maps and Location-Based Services In-Class Assignment (Week 10)
12	SMS
13	Review

**COURSE TITLE: Object-Oriented Programming II**

**COURSE CODE: COMP 3607**

**TYPE: Elective**

**LEVEL: 3**

**SEMESTER: 2**

**START DATE: JAN-15-2017**

**DEPARTMENT and FACULTY: DCIT/FST**

**CREDITS: 3**

**PRE-REQUISITE(S): COMP 2603**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 2-hour lab, 6 hours per week independent study

**COURSE DESCRIPTION**

The course looks at the main tools of modern object-oriented software development. The main tools are: design-support tools (principally design patterns) and programming-support tools (principally IDE). This course has a strong emphasis on project design and programming using design patterns

Each pattern represents a best practice solution to a software problem in a specific context. The course covers the rationale and benefits of object-oriented software design patterns. Numerous problems will be studied to investigate the implementation of good design patterns.

**COURSE RATIONALE**

Computer programmers and engineers must be able to use a variety of programs and systems for designing solutions to common information technology issues. The object-oriented programming paradigm has made it easier to handle software development involving complex tasks since it easily facilitates the decomposition of problems into modular entities. The course will allow students to practice advance concepts in Object-Oriented design. This course will help motivated students to be primary contributors to any small to mid-sized commercial or open-source software project.

**COURSE CONTENT**

1. Principles of good software design. Examine what causes software to rot.
2. UML - Class, Sequence and Use-Case Diagrams
3. Design Patterns: Strategy, Observer, Factory, Singleton, FlyWeight, Command, Adapter, Facade, Template Method, Iterator, Composite, State, Proxy and Mediator
4. Object Persistence: using relational databases
5. MVC architecture
6. Concept of Code Refactoring

**COURSE LEARNING OUTCOMES**

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

1. Analyze problems using UML tools
2. Design OO solutions using UML tools
3. Discuss the principles of good object-oriented design
4. Use design patterns to facilitate good object-oriented design
5. Explain the reasoning for each object oriented design principle.
6. Apply knowledge of design patterns to solve common programming problems
7. Draw high level class diagrams in UML for each pattern.

8. Describe the consequences of applying each pattern to the overall software quality of a system.

## COURSE ASSESSMENT

Assessment	Learning Outcomes								Weighting %	Assessment Description	Duration
	1	2	3	4	5	6	7	8			
Assignment 1	X	X	X	X					20%	Problems	
Assignment 2	X	X	X	X	X	X				Problems	
Assignment 3	X	X	X	X	X	X	X			Problems	
Exam 1	X	X	X	X	X	X			30%	Problems & short answer	1.5 hours
Exam 2	X	X	X	X	X	X	X			Problems & short answer	1.5 hours
Final Examination	X	X	X	X	X	X	X	X	50%	Problems & Short Answer Questions	2 hours
TOTAL %									100%		

## TEACHING STRATEGIES

Problem-based learning will be the main teaching strategy utilized in this course. Feedback from assignments/problem sheets/course work tests will be used to tailor subsequent lectures/tutorials. Students will be given ample opportunity to develop correct, readable programs using object-oriented programming techniques.

## RESOURCES

### Lecture notes

### Textbooks

Mohan, Permanand. *Fundamentals of Object-Oriented Programming in Java*, CreateSpace Independent Publishing Platform. 1<sup>st</sup> Edition (February 28, 2013). ISBN-10: 1482587521, ISBN-13: 978-1482587524.

Harvey Deitel and Paul Deitel (2002). *Java: How to Program*, 5th Edition. Prentice-Hall.

Bruce Eckel (2002). *Thinking in Java*, 3rd Edition. Prentice-Hall. Free electronic copy available on the Internet from <http://www.mindview.net/Books/TIJ/>.

Timothy Budd (1999). *Understanding Object-Oriented Programming With Java*, Updated Edition. Addison Wesley.

## COURSE CALENDAR:

Week	Topic
1	Principles of good software design. Examine what causes software to rot. UML - Class, Sequence and Use-Case Diagrams
2	Introduce Design Patterns: Singleton and discuss the Code of Refactoring Code
3-7	Strategy, Observer, Factory, FlyWeight, Command, Adapter, Facade, Template Method, Iterator, Composite, State, Proxy and Mediator <b>Assignment 1 (Week 3)</b>



	<b>Assignment 2 (Week 6)</b> <b>Course work Exam 1 (Week 7)</b>
8	Object Persistence: using relational databases <b>Assignment 3 given in Week 9</b>
9	MVC architecture
10	<b>Course work Exam 2</b>
11	MVC architecture
12	Putting it all together
13	Revision

**COURSE TITLE: Intelligent Systems**  
**COURSE CODE: COMP 3608**  
**TYPE: Elective**  
**LEVEL: 3**  
**SEMESTER: 1**  
**START DATE: SEPT-01-2016**  
**DEPARTMENT and FACULTY: DCIT/FST**  
**CREDITS: 3**  
**PRE-REQUISITE(S): COMP 2600 AND MATH 2250**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 2-hour lab, 6 hours per week independent study

**COURSE DESCRIPTION**

This course provides an introduction to artificial intelligence and its applications. The course concentrates on solving problems associated with artificial intelligence using data mining and knowledge representation tools. Topics covered in the course include characteristics of intelligent systems, rule-based expert Systems; production rules, reasoning with uncertainty, search strategies, artificial neural networks, genetic algorithms, knowledge engineering and data mining

**COURSE RATIONALE**

Intelligent systems continue to shape the development of modern computers e.g. robots. Whereas the social sciences study the theory of human intelligence, this course teaches students how to build intelligent entities that model human thinking. This course will provide students with the tools they need to perform effectively in the development of new artificial intelligent applications.

**COURSE CONTENT**

1. Introduction to Intelligence.
  - 1.1. Turing test
  - 1.2. Chinese room
  - 1.3. Rational vs non rational reasoning
  - 1.4. Types of problems solved by intelligent systems
  - 1.5. Intelligent agents.
2. Search Strategies
  - 2.1. Problem solving by searching.
  - 2.2. Uninformed search (breadth-first, depth-first, depth-first with iterative deepening)
  - 2.3. Heuristics and informed search (hill-climbing, generic best-first, A\*)
  - 2.4. Constructing search trees, dynamic search space, combinatorial explosion of search space.
  - 2.5. Genetic Algorithm
  - 2.6. A\* Search
  - 2.7. Minimax Search
3. Reasoning
  - 3.1. Expert and recommendation systems
  - 3.2. Forward and backward chaining
  - 3.3. Probabilistic Reasoning and inference -Bayes theorem and networks.
  - 3.4. Knowledge representation.
  - 3.5. Decision theory.
4. Machine Learning

- 4.1. Supervised and unsupervised learning.
- 4.2. Kmeans
- 4.3. Neural Networks
- 4.4. Support Vector Machines
- 4.5. Self-organizing feature maps
- 4.6. Nearest neighbor algorithms
- 4.7. Vector Quantization
- 4.8. Applications of machine learning algorithms.

## COURSE LEARNING OUTCOMES

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

1. Develop an intelligent system designed for solving specific learning problems.
2. Select an appropriate intelligent system for a given task.
3. Differentiate between the 3 main learning styles.
4. Implement a search algorithm to solve a given problem based upon selection criteria.
5. Implement a designed genetic algorithm solution to a given problem.
6. Differentiate genetic algorithms with classic search techniques.
7. Implement a designed A\*/beam search to solve a given problem.
8. Implement simple algorithms for supervised learning, reinforcement learning, and unsupervised learning.

## COURSE ASSESSMENT

Assessment	Learning Outcomes								Weighting %	Assessment Description	Duration
	1	2	3	4	5	6	7	8			
Assignment 1	X	X	X	X					20%	Short answer	
Assignment 2	X	X	X	X	X	X				Short answer	
Assignment 3					X	X	X	X			
Exam 1	X	X	X	X	X	X			30%	Problems & short answer	1.5 hour
Exam 2					X	X	X	X		Problems & short answer	1.5 hour
Final Examination	X	X	X	X	X	X	X	X	50%	Short Answer Questions	2 hours
TOTAL %									100%		

## TEACHING STRATEGIES

Problem-based learning is applied throughout the course allowing the student to examine authentic issues and design appropriate solutions. Students will discuss their chosen solutions during class sessions.

## RESOURCES

### Lecture notes

### Textbooks

Crina Grosan, Ajith Abraham, Intelligent Systems: 17 (Intelligent Systems Reference Library), Springer, 2011

Peter Flach, Machine Learning: The Art and Science of Algorithms that Make Sense of Data, Cambridge, 2012, ISBN-13:860-1200462906 ISBN-10:1107422221

### **COURSE CALENDAR**

<b>Week</b>	<b>Topic</b>
1	Definition of intelligence. Turing test and Chinese room Types of problems solved by intelligent systems.
2	Rational vs non rational reasoning, Intelligent agents, Searching
3,4,5	Search strategies. Genetic Algorithm, A* Search, Search trees Hill climbing, gradient decent. <b>Assignment 1 (Week 4)</b>
6,7,8	Reasoning, Expert systems, introduction to CLIPS <b>Assignment 2 (Week 6)</b> <b>Examination 1 (Week 8)</b>
9,10	Machine learning Techniques <b>Assignment 3 (Week 10)</b>
11,12	Speech and Image Recognition <b>Examination 2 (Week 11)</b>
13	Revision

**COURSE TITLE: Game Programming**

**COURSE CODE: COMP 3609**

**TYPE: Elective**

**LEVEL: 3**

**SEMESTER: 1**

**START DATE: SEPT-01-2016**

**DEPARTMENT and FACULTY: DCIT/FST**

**CREDITS: 3**

**PRE-REQUISITE(S): COMP 2603 AND COMP 2606**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 2-hour lab, 6 hours per week independent study

**COURSE DESCRIPTION**

The Game Programming course will allow students to combine concepts taught in order courses together with the new game programming concepts taught in this course, in order to build games. The students will be introduced to an appropriate 2D programming API (e.g., Java), the game loop, game entities, images, sound, animations, game physics and user input. At the end of the course students will have a good grasp on the concepts of game programming and will be able to produce games for multiple platforms. The course covers the fundamental aspects of images, sounds, animations and sprites and shows how to develop a two-dimensional game using these elements. Mathematics and Physics principles are discussed throughout the course whenever they are pertinent to the topics being presented (e.g. collision detection of sprites).

**COURSE RATIONALE**

Game programming involves the application of a wide range of skills and concepts from different areas of Computer Science and other disciplines. These include programming, data structures, computer architecture, operating systems, human computer interaction, and software engineering. Thus, the Game Programming course is intended to give students an introduction to the Computer Science aspects of game programming using an object-oriented programming language.

**COURSE CONTENT**

1. Review of Object Oriented Concepts
  - 1.1 Inheritance
  - 1.2 Interfaces
  - 1.3 Inner Classes
  - 1.4 Using the 2D API Documentation
2. Introduction to 2D Graphical API
  - 2.1 Creating and drawing shapes
  - 2.2 Drawing Text to screen
3. Basic Game Programming Concepts
  - 3.1 The basic game loop
  - 3.2 Drawing and Updating
  - 3.3 Game Entities
4. Movement and User Input

- 4.1 Moving a simple 2D shape
- 4.2 Getting input from the user
- 5. Simple Collision Detection
  - 5.1 Bounding rectangles
  - 5.2 Rectangle intersection
  - 5.3 Screen boundaries
- 6. Design Concepts
  - 6.1 Singleton Design Pattern
  - 6.2 Composite Design Pattern
  - 6.3 Hash tables
  - 6.4 Array Lists
- 7. Images and Sprites
  - 7.1 Loading images
  - 7.2 The Sprite class
  - 7.3 Flickering, double buffering, tearing, and page flipping
  - 7.4 Image Characteristics (Opaque, Transparent, Translucent, etc.)
  - 7.5 Asset Manager
- 8. Animation
  - 8.1 2D animation concepts
  - 8.2 Loading and manipulating Sprite sheets
  - 8.3 AnimatedSprite class
- 9. Sound
  - 9.1 A Sound API
  - 9.2 MIDI
  - 9.3 Loading and playing sound file in Java
  - 9.4 Sound Effects
  - 9.5 SoundManager class
- 10. Parallax Backgrounds
  - 10.1 Simulating movement and distance in games
  - 10.2 ParallaxBackground class
- 11. Movement and User Input
  - 11.1 Managing user input
  - 11.2 KeyboardInputService class
- 12. Design Concepts
  - 12.1 Object Pool design pattern
  - 12.2 Game States
  - 12.3 Managing game screens
- 13. 2D Platformer
  - 13.1 Side Scroller Concept
  - 13.2 Creating a game camera
  - 13.3 Tiled Map Editor
  - 13.4 Loading and displaying tile maps
  - 13.5 Map collision detection
- 14. Simple Game Physics
  - 14.1 Smooth jump motion and Gravity
  - 14.2 Projectile motion
- 15. Special Topics in Game Programming
  - 15.1 Simulating 3D Using 2D
  - 15.2 Real time strategy games

## 15.3 Multi-player games

### COURSE LEARNING OUTCOMES

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

1. Use an appropriate 2D API to draw and colour various shapes.
2. During the execution of a game manipulate identified characteristics of images and sounds
3. Implement user interaction with the keyboard and mouse
4. Create a 2D platform game
5. Implement a specific set of basic game design principles.
6. Implement some basic principles of game design
7. Develop by course end at least one game illustrating the design principles, image characteristics and sounds games outlined.

### COURSE ASSESSMENT

Assessment	Learning Outcomes							Weighting %	Assessment Description	Duration
	1	2	3	4	5	6	7			
Assignment 1	X	X	X	X				20%	Problems	
Assignment 2	X	X	X	X	X	X			Problems	
Assignment 3	X	X	X	X	X	X			Problems	
Assignment 4	X	X	X	X	X	X	X		Problems	
Presentations	X	X	X	X	X	X	X	5%	Rubric-based	
Practical Exam	X	X	X	X	X	X	X	25%	Problems	1.5 hour
Final Examination	X	X	X	X	X	X	X	50%	Problems & Short Answer Questions	2 hours
TOTAL %								100%		

### TEACHING STRATEGIES

Problem-based learning is applied throughout the course allowing the student to examine authentic issues and design appropriate solutions. Students will discuss their chosen solutions during class sessions.

### RESOURCES

#### Lecture notes

#### Textbooks

Brackeen, D., *Developing Games in Java*. Berkeley, California: New Riders Publishing. 2004.

Davison, A., *Killer Game Programming in Java*. Sebastopol, California: O'Reilly Media Inc. 2005. Book  
Web Site: <http://fivedots.coe.psu.ac.th/~ad/jg/>

### COURSE CALENDAR

Week	Topic
1	Review of Object Oriented Concepts Introduction to 2D Graphical API Basic Game Programming Concepts

2	Implementing Game Programming Concepts Movement and User Input Simple Collision Detection <b>Assignment 1</b>
3	Design Concepts Images and Sprites
4	Animation Sound
5	Parallax Backgrounds Movement and User Input <b>Assignment 2</b>
6	Design Concepts
6, 7	2D Platform Game
8	Simple Game Physics <b>Assignment 3</b>
9 - 12	Special Topics in Game Development
10	Presentation, Practical Exam
11	<b>Assignment 4</b>
13	Revision



**COURSE TITLE: Big Data Analytics**  
**COURSE CODE: COMP 3610**  
**TYPE: Elective**  
**LEVEL: 3**  
**SEMESTER: 2**  
**START DATE: JAN-15-2017**  
**DEPARTMENT and FACULTY: DCIT/FST**  
**CREDITS: 3**  
**PRE-REQUISITE(S): COMP 3605**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 2-hour lab, 6 hours per week independent study

**COURSE DESCRIPTION**

The course exposes students to the various approaches used in analyzing big data. The course focuses on the use of data warehouses and distributed database design to analyse structured data and Hadoop for unstructured data. Students are also introduced to various NoSQL databases and approaches for storing and processing large volumes of data. The MapReduce model for processing large data sets is discussed and it is compared against the parallel database approach.

**COURSE RATIONALE**

The explosion of social media and global computerization of everyday life has led to the evolution of big data. This data can be both structured and unstructured. Large companies are constantly looking for greater understanding of their data to answer their specific business questions in order to increase sales or operational efficiency. Thus, this course introduces how data warehouses are used to analyse structured data while Hadoop handles unstructured data with MapReduce. Parallel computing, distributed database and parallel database concepts are also covered to demonstrate how they can be leveraged in big data analytics.

**COURSE CONTENT**

1. Big Data
2. Parallel & Distributed Computing
  - 2.1. Goals of parallelism
  - 2.2. Throughput versus concurrency (sharing access to shared resources)
  - 2.3. Programming constructs for coordinating multiple simultaneous computations
  - 2.4. Load Balancing
3. Parallel and Distributed DBMS
  - 3.1. Distributed Database Design
  - 3.2. Distributed Query Processing
  - 3.3. Replication Management
  - 3.4. Parallel DBMS Architecture
4. Data warehouses
  - 4.1. Data Warehouse considerations
  - 4.2. Star Schema
  - 4.3. SQL Aggregation
5. NoSQL Databases
  - 5.1. MongoDB
  - 5.2. Dynamo

- 5.3. BigTable
- 6. MapReduce Processing & Hadoop
  - 6.1. Foundations of mapping and reduction for the
  - 6.2. Introduction to Hadoop
  - 6.3. Advantages and disadvantages of the HDFS
  - 6.4. Speedup and scale up within parallel systems
  - 6.5. Compare Hadoop and Parallel Databases
  - 6.6. YARN
  - 6.7. Utilize Analytical tools in Hadoop (Mahout, Solr, HBase, Storm)

## COURSE LEARNING OUTCOMES

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

1. Explain why synchronization is necessary in a specific parallel program
2. Apply task-based decomposition to parallelize an algorithm.
3. Explain the techniques used for data fragmentation, replication and allocation during the distributed database design process
4. Evaluate simple strategies for executing a distributed query to select the strategy that minimizes the amount of data transfer
5. Describe distributed concurrency control
6. Design simple data marts
7. Compare different NoSQL Databases
8. Describe major approaches to storing and processing large volumes of data
9. Utilize the MapReduce model

## COURSE ASSESSMENT

Assessment	Learning Outcomes									Weighting %	Assessment Description	Duration
	1	2	3	4	5	6	7	8	9			
Assignment 1	X	X	X							20%	Problems	
Assignment 2				X	X	X	X				Problems	
Lab Exam	X	X	X	X	X	X	X	X		15%	Rubric-based	
Coursework Exam 1	X	X	X	X	X	X				15%	Problems	1 hour
Coursework Exam 2	X	X	X	X	X	X	X	X	X		Problems & Short Answer Questions	1 hour
Final Examination	X	X	X	X	X	X	X	X	X	50%	Problems & Short Answer Questions	2 hours
TOTAL %										100%		

## TEACHING STRATEGIES

Problem-based learning is applied throughout the course allowing the student to examine the main concepts. Students will discuss their chosen solutions during class sessions. The lab exam will be used to ensure that core practical skills are developed.

## RESOURCES

Lecture notes

### Textbooks

Ozsu, M. Tamer, Valduriez, Patrick. *Principles of Distributed Database Systems (third edition)*, Springer. ISBN: 9781441988331

White, Tom. Hadoop: The Definitive Guide (4th Edition), O'Reilly Media. ISBN-10: 1491901632

### COURSE CALENDAR

Week	Topic
1	Parallel and Distributed Computing
2	Distributed Databases
3	Parallel Algorithms
4	Parallel Databases <b>Assignment 1</b>
5	Data Warehouses
6	<b>CW Examination 1</b>
7	NoSQL Databases
8	MapReduce <b>Assignment 2</b>
9	Hadoop
10	<b>Lab Exam</b>
11	Hadoop vs. Parallel Databases
12	<b>CW Examination 2</b>
13	Revision

**COURSE TITLE: Modelling and Simulation**

**COURSE CODE: COMP 3611**

**TYPE: Elective**

**LEVEL: 3**

**SEMESTER: 1**

**START DATE: SEPT-01-2016**

**DEPARTMENT and FACULTY: DCIT/FST**

**CREDITS: 3**

**PRE-REQUISITE(S): MATH 2250**

**ESTIMATED STUDY HOURS:**

2 1-hour lectures, 1 2-hour lab, 6 hours per week independent study

**COURSE DESCRIPTION**

This course covers basic to intermediate techniques for discrete event simulation of common scenarios. It draws on concepts from the theory of computation and computer programming. In many real world situations, it is infeasible to develop a precise mathematical model with a closed-form analytic solution for a given problem. Modeling and simulation is a means of coping with this type of problem.

**COURSE RATIONALE**

Systems have become so complex that it is often the case that understanding them cannot be done analytically. Therefore, their behavior can be observed by modeling them and simulating them. Thus, this course introduces the theories and applications of computer modeling and simulation, focusing on discrete event system modeling and simulation.

**COURSE CONTENT**

- Discrete and Continuous Systems
- Discrete-Event System Simulation; Dynamic Allocation and Linked Lists
- Queuing Models; Steady-State Behavior of Infinite-Population Markovian Models
- Single-Server Queues with Poisson Arrivals and Unlimited Capacity
- Analysis of Simulation Data; Goodness-of-Fit Tests, Chi-Square Test
- Non-stationary Poisson Process
- Output Analysis for Terminating Simulations
- Error Estimation for Steady-State Simulation

**COURSE LEARNING OUTCOMES**

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

1. Describe the concept of a system and its environment.
2. Define discrete event simulation and perform walk-through simulations.
3. Create a computer simulation using different variations of queuing models.
4. Develop computer simulations of such systems.
5. Analyze the results of programmed simulation runs using a variety of statistical techniques.
6. Perform output analysis for a single model.
7. Demonstrate inter-personal skills, teamwork and effective use of appropriate technology associated with modelling and simulation.
8. Develop communication skills through written material and oral presentations

**COURSE ASSESSMENT**

Assessment	Learning Outcomes								Weighting %	Assessment Description	Duration
	1	2	3	4	5	6	7	8			
Assignment 1	X	X	X						10%	Problems	
Assignment 2	X	X	X	X						Problems	
Project and Presentation	X	X	X	X	X	X	X	X	15%	Rubric-based	
Coursework Exam	X	X	X	X	X	X			25%	Problems	1 hour
Final Examination	X	X	X	X	X	X			50%	Problems & Short Answer Questions	2 hours
TOTAL %									100%		

## TEACHING STRATEGIES

Interactive lectures will be used to introduce the main concepts of the course. However, discussion and brain storming methods will be used throughout the course with the goal of allowing students to engage the problem solving process. Labs will also be used to demonstrate, in a practical manner, the concepts taught in lectures.

## RESOURCES

### Lecture notes

### Textbooks

- [Discrete-Event System Simulation \(Fourth Edition\)](#), Banks, Carson, Nelson, and Nicol, Prentice-Hall, 2005. Henceforth, referred to as [BCNN05].

## COURSE CALENDAR:

Week	Content
1	Introduction: Systems and System Environment, Components of a System, Discrete and Continuous Systems, Model of a System, Types of Models, Discrete-Event System Simulation
2	Random Distributions used in Simulations Analysis of Simulation Data: Input Modeling, Data Collection, Identifying the Distribution with Data, Histograms, Selecting the Family of Distributions and Parameter Estimation
3	Preliminary Statistics: Sample Mean and Sample Variance, Suggested Estimators, Goodness-of-Fit Tests, Chi-Square Test, Chi-Square Test with Equal Probabilities, Kolmogorov-Smirnov Goodness-of-Fit Test, p-Values and "Best Fits" <b>Assignment 1</b>
4-6	Queueing Models: Characteristics of Queueing Systems, Queue Behavior and Queue Discipline, Service Times and the Service Mechanism, Steady-State Behavior of Infinite-Population Markovian Models Single-Server Queues with Poisson Arrivals and Unlimited Capacity, Multiserver Queue, Multiserver Queues with Poisson Arrivals and Limited Capacity, Steady-State Behavior of Finite-Population Models, Networks of Queues
7	<b>Revision/Exam 1</b>
8	Writing Simulations of real business type problems in a chosen General Purpose Language <b>Assignment 2 and Project Given</b>

9	Output Analysis for a Single Model: Types of Simulations with Respect to Output Analysis, Stochastic Nature of Output Data, Measures of Performance and Their Estimation, Point Estimation, Confidence-Interval Estimation Output Analysis for Terminating Simulations, Confidence Intervals with Specified Precision, Estimating Probabilities and Quantiles from Summary Data, Output Analysis for Steady-State Simulations
10	Initialization Bias in Steady-State Simulations, Error Estimation for Steady-State Simulation, Replication Method for Steady-State Simulations, Sample Size in Steady-State Simulations
11-13	<b>Project Presentations (Week 12)</b> Revision

**COURSE TITLE: Special Topics in Computer Science**

**COURSE CODE: COMP 3612**

**TYPE: Elective**

**LEVEL: 3**

**SEMESTER: 1/2/3**

**START DATE: SEPT-01-2016**

**DEPARTMENT and FACULTY: DCIT/FST**

**CREDITS: 3**

**PRE-REQUISITE(S): COMP 2600, COMP 2603**

### **ESTIMATED WORK HOURS:**

2 1-hour lectures, 1 1-hour tutorial, 6 hours per week independent study

### **COURSE DESCRIPTION**

Each time this course is offered, it addresses a topic in computer science that is not covered as a regular course. Topics may change from year to year. Some of these topics include: Computer Graphics; Robotics, Computer Assisted Design (CAD); E-Learning; Mobile Health; Data Visualization; E-Science; Speech Synthesis; Advanced Processor Architecture; Expert Systems; Computability and Complexity; Proof of Correctness of Programs; Image Processing; any other approved topic.

### **COURSE RATIONALE**

The field of Computer Science advances rapidly. Despite frequent curriculum review, there is often a need to offer a course which familiarizes students with new or important developments in the field. This course provides a means to deliver material that is not currently being addressed in a regular course. Depending on its importance and its relevance to the Computer Science curriculum, a topic covered in this course may later be offered as a regular course.

### **COURSE LEARNING OUTCOMES**

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

1. Demonstrate a comprehensive understanding of techniques, and a knowledge of the literature, applicable to the chosen computer science specialization
2. Apply research methods, techniques, and problem solving approaches from the field of research in which they are specializing

### **COURSE ASSESSMENT**

Assessment	Learning Outcomes		Weighting %	Assessment Description
	1	2		
Course work	X	X	50%	Problems and Short Answers
Final Exam	X	X	50%	
TOTAL %			100%	

**TEACHING STRATEGIES**

Interactive lectures would be used to deliver the material to students. This will be inclusive of case analysis, problem-solving and weekly discussions.

**RESOURCES: Reading Material**

The resources for the course will be determined whenever the course is offered. They will vary from year to year depending on the topic that is being offered.

**COURSE CALENDAR**

The course calendar will be established each time the course is offered. It will depend on the special topic being offered in any given year.



## **6.2 Courses from the Department of Mathematics and Statistics**

The B.Sc. Computer Science degrees require two MATH courses from the Department of Mathematics and Statistics (Faculty of Science and Technology). These courses are:

MATH 1115: Fundamental Mathematics for the General Sciences I

MATH 2250: Industrial Statistics 1

MATH 1115 is currently a requirement for several degrees offered in the Faculty of Science and Technology. We have been informed that MATH 1115 was last reviewed in 2011. MATH 2250 is currently a requirement for several degrees offered in the Faculty of Engineering.

Detailed course outlines for these two courses as obtained from the Department of Mathematics and Statistics are given below.

**COURSE CODE: MATH 1115**

**COURSE TITLE: Fundamental Mathematics for the General Sciences I**

**NO. OF CREDITS: 3**

**LEVEL: 1**

**SEMESTER: 1, II, and III**

**PREQUISITES: None**

### **Course Rationale**

Over the years, we have observed that students entering the Faculty of Science and Technology without having done Cape Advanced level mathematics have a general deficiency in basic mathematical skills. This course has been specially crafted to address this problem. This foundation course is designed to meet the demands of several disciplines within the Faculty of Science and Technology that do not have their own courses in basic mathematics. The Faculty of Science and Technology has seen it necessary to ensure that entering students possess the necessary mathematical skills to excel in their chosen discipline within the Faculty, without having to take courses specifically designed for students of Mathematics.

Students wishing to enter the Faculty of Science and Technology without the necessary CSEC Mathematics qualifications will be given the option to take and pass the course “Fundamental Mathematics for the General Sciences I” during the summer semester in order to matriculate into the Faculty.

Students with any two units of Cape Advanced Level Mathematics or equivalent will not receive credits for this course. It should be noted that this course **cannot be considered** as a substitute or a pre-requisite for any of the courses offered by the Department of Mathematics and Statistics.

### **Course Description**

The main objective of this course is to provide entering undergraduate students with a set of mathematical tools and methods that can be applied to their scientific field of choice.

The major topics will be prefixed by typical mathematical problems that arise frequently in practical applications of the varying fields of science. As a service course in Mathematics, it should be considered as a broad introduction of the typical methods utilized in the applied sciences for solving problems. Little attention will be given to the underlying concepts of mathematical theory during lecture hours. Emphasis will be placed on the use of Microsoft Excel as a tool for the presentation of data in Laboratory Reports.

A sound knowledge of all concepts from CSEC Mathematics will be assumed. It should be stressed that it is **not meant to be** a pre-requisite for any other mathematics course offered by the Department of Mathematics and Computer Science.

### **Learning Outcomes**

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

- Solve problems that arise in their own field of study, by making use of the basic mathematical concepts outlined in the course.
- Recognize types of numbers (Naturals, Integers, Rationals, Reals, Complex).
- Determine the number of significant figures in an answer.
- Round off numbers to a given number of decimal places.

- Utilize scientific notation.
- Solve simple equations and inequalities.
- Manipulate numbers: negatives, absolutes, fractions, decimals and percentages.
- Calculate ratios and proportions.
- Evaluate factorials, reciprocals and exponentials.
- Utilize the basic rules of indices and logarithms.
- Find the inverse of a function.
- Manipulate trigonometric functions and their graphs.
- Utilize common trigonometric identities.
- Solve trigonometric equations, quadratics and polynomial equations.
- Utilize the remainder theorem.
- Simplify fractions via the rules of partial fractions.
- Calculate gradients and intercepts.
- Determine the mean and standard deviation for a set of data.
- Interpret measures of central tendency.
- State and use the basic rules of probability.
- Interpret normal and binomial distributions.
- Utilize the chi-squared test.

## **Course Content**

### **Algebra, Functions, Numerical Concepts:**

- Types of numbers (Naturals, Integers, Rationals, Reals, Complex). Significant figures, decimal places, scientific notation. Negative numbers. Absolute numbers. SI system of units, usage and prefixes  
(Examples: surface area, volume etc.)
- Manipulating numbers: fractions and decimals. Adding, subtracting, multiplying and dividing fractions.

Calculating percentages, ratios and proportions. Factorials. Solving simple inequalities.

- Rules of indices, logarithms (example: calculating pH scales) and exponentials. Functions and inverse functions. Solving simultaneous equations.
- Trigonometric functions and their graphs. Common trigonometric identities. Solution of trigonometric equations.
- Partial fractions. Solving quadratic equations.
- Remainder theorem. Solving polynomial equations.

#### **Data Analysis:**

- Calculation of gradients and intercepts. Dependent and independent variables. Extrapolation techniques.  
Line of best fit.

#### **Basic Statistics:**

- Mean, median, mode and standard deviation.
- The normal and binomial distributions. The chi-squared test.

#### **Teaching Methodology**

Lectures: Two (2) lectures each week (50 minutes each).

Tutorials: One (1) tutorial session for eleven weeks (50 minutes each) – These tutorials will serve as a forum for discussing concrete examples that students will encounter in their respective fields. The solutions to the six problem sheets will also be incorporated.

Computer Labs: Two mandatory 2- hour practical computer lab sessions in Microsoft Excel.

#### **Assessment**

<b>Method of Evaluation</b>	<b>Percentage of Grade</b>
<b>Coursework:</b> 2 coursework examinations: 25% 6 problem sheets: 10% Computer lab assessment: 5%	40 %
<b>Final Examination</b> One 2-hours written paper (Covers entire course – all lectures, readings and tutorials.)	60%

## COURSE CALENDAR

Week	Lecture subjects
1	<b>Algebra, Functions, Numerical Concepts:</b> Types of numbers. Significant figures, decimal places, scientific notation. Absolute numbers. SI system of units, usage and prefixes.
2	<b>Algebra, Functions, Numerical Concepts:</b> Manipulating numbers: fractions and decimals. Adding, subtracting, multiplying and dividing fractions. Calculating percentages, ratios and proportions. Factorials. Solving simple inequalities.
3	<b>Algebra, Functions, Numerical Concepts:</b> Rules of indices, logarithms (example: calculating pH scales) and exponentials. Functions and inverse functions. Solving simultaneous equations.
4	<b>Functions:</b> Functions, relations, and inverse functions. Composition of functions. Logarithmic and exponential functions and their graphs.
5	<b>Algebra, Functions, Numerical Concepts:</b> Trigonometric functions and their graphs. Common trigonometric identities. Solution of trigonometric equations.
6	<b>Algebra, Functions, Numerical Concepts:</b> Partial fractions. Solving quadratic equations.
7	<b>Algebra, Functions, Numerical Concepts:</b> Remainder theorem. Solving polynomial equations.
8	<b>Data Analysis:</b> Dependent and independent variables. Extrapolation techniques. Line of best fit. Error.
9	<b>Probability:</b> Sample spaces. Mutually exclusive events. Combinations. Calculation of probabilities.
10	<b>Statistics:</b> Measures of central tendency. Mean, median, mode and standard deviation.
11	<b>Statistics:</b> The binomial distribution.
12	<b>Statistics:</b> The normal distribution.
13	<b>Statistics:</b> The chi-squared test.

## Required Reading

Essential Text

- D. Jogie, D. Comissiong, **Fundamental Mathematics for the General Sciences.** (*Available in the Math Department. Cost: \$150*)
- Online notes will be provided (via myElearning).

**The University of the West Indies, St. Augustine**  
**Department of Mathematics and Statistics**

**Course Code:** MATH 2250

**Course Title:** Industrial Statistics- I

**Level:** 2

**Semester:** 1

**Number of Credits:** 3

**Prerequisites:** ENG1180

**Course Rationale**

Statistics is concerned with the collection, organizing, analyzing and interpretation of quantitative and numerical data. From comparing batting averages of cricket players to predicting the winner of parliamentary election to determining the effectiveness of a drug, statistics is commonly applied. Indeed, a working knowledge & understanding of statistics is important in virtually all forms of endeavor: business, industry, government, education, social science, engineering, applied science, behavioural science and medicine.

**Course Description**

This is an introductory course in probability & statistics that is designed for engineering students. Topics to be covered include data analysis, probability, random variables, discrete and continuous probability distributions, estimation, hypothesis testing, sampling, and introductory linear regression and ANOVA and Experimental Design. Some topics may be deleted and other added depending on time constraints.

Although this course does not involve complex mathematics, students who enrol in Industrial Statistics need a background in algebra and calculus ... the approximate equivalent of M12A or equivalent.

**Learning Objectives**

At the end of this course, students should be able to:

- Utilize data collection methods and interpretation of the same.
- Use tabular and graphical formats to display univariate and bivariate data sets.
- Explain and use the concepts of probability, random variables and their distributions, in particular the discrete and continuous distributions.
- Outline and utilize the concepts of estimation and hypothesis testing.
- Carry out correlation, regression and chi-square analyses.
- Perform the t-test with one-sample and two-samples.
- Be familiar with computer-based statistical analysis through a suitable software package like SPSS/ R.

### **Knowledge, Skills and Abilities**

- Collection, cleaning, organization of raw data
- Utilization of probability theory; Discrete and continuous probability distributions
- Application of probability theory and distributions to statistical inferences
- Preparation and graphing of univariate or bivariate data sets
- Utilization of descriptive statistics.
- Development of statistical models to engineering applications
- Application of appropriate statistical methods to solve real engineering problems

### **Assessment of Learning:**

- Representative assessment tasks:
  - Home assignments.
  - In-class activities.
  - Coursework
  - Final Examinations

### **Course Assessment**

Grades will be assigned in the overall course scores with the following weights:

• Assignments	10
• Three Course Work Exams	30
• Final Exam	60
• <b>TOTAL</b>	<b>100%</b>

### 1. Assignments

Students who do not submit an assignment will be assigned a mark of zero (0) for that assignment. It is recommended that students complete all assignments in order to achieve the learning outcomes of the course.

### 2. Course work Exams

The exams will cover material from lectures and assigned readings. They will be both conceptual and computational in nature. Exams must be taken on the scheduled day. Make-up exam will be given only in cases of submission of valid medical report.

### 3. Final Exam

## TOPIC(S) and Course Schedule (Calendar)

The following is a listing of the class meetings and the tentative topics that will be covered on those days.

#### Tentative class Calendar:

Week 1	Tuesday	Syllabus Day and Overview of programme
	Thursday	Probability
Week 2	Tuesday	Mathematical expectation
	Thursday	Mathematical expectation
Week 3	Tuesday	Discrete Random Variables and Probability Distributions
	Thursday	Discrete Random Variables and Probability Distributions
Week 4	Tuesday	Continuous Random Variables, Probability Distributions
	Thursday	Continuous Random Variables, Probability Distributions
Week 5	Tuesday	Joint Probability distribution and Random Variable
	Thursday	1 <sup>st</sup> Course Work
Week 6	Tuesday	Point estimates
	Thursday	Interval estimates
Week 7	Tuesday	Statistical Intervals Based on a Single Sample; SPSS/R
	Thursday	Tests of Hypotheses Based on a Single Mean: SPSS/R
Week 8	Tuesday	Inferences Based on Two Samples.
	Thursday	2 <sup>nd</sup> Course Work
Week 9	Tuesday	Inferences Based on Two Samples; SPSS/R
	Thursday	The Analysis of Variance
Week 10	Tuesday	Multifactor Analysis of Variance
	Thursday	3 <sup>rd</sup> Course Work Exam
Week 11	Tuesday	Randomised block design and analysis
	Thursday	Simple and Multiple Linear Regression



Week 12	Tuesday	Multiple Linear Regression SPSS/R
	Thursday	Non parametric statistical tests
Week 13	Tuesday	Review
	Thursday	Review

\* Note that this structure represents a plan and is subject to adjustment depending on session/term/holidays, etc.

### **Required Text**

<b>Textbook Title</b>	Probability and Statistics for Engineering and the Sciences, 7 <sup>th</sup> Edition
<b>Author and Publisher</b>	Jay L. Devore, Thomson, 2008
<b>ISBN</b>	ISBN-10: 0495382175; ISBN-13: 9780495382171

### **6.3 Courses from the Faculty of Social Sciences**

The B.Sc. Computer Science and Management degree comprises several courses from the Faculty of Social Sciences (FSS). These courses are offered as part of existing programmes in the FSS which are currently being revised. The new programmes have already been approved by the FSS Faculty Board and are currently awaiting approval from CETL. FSS is hoping to submit the revised curriculum documents to AQAC before the start of the 2016/2017 academic year.

The following is an outline of the FSS courses that are part of the B.Sc. Computer Science and Management degree:

**LEVEL: I**

**SEMESTER: I OR II (FT & EU)**

**COURSE CODE: ACCT 1002**

**COURSE TITLE: INTRODUCTION TO FINANCIAL ACCOUNTING**

**CREDITS: 3**

**PREREQUISITES: NONE**

**DEPARTMENT RESPONSIBLE: MANAGEMENT STUDIES**

**COURSE DESCRIPTION:** An introductory course designed for students of accounting and those in other areas of study. It aims at providing a practical and a theoretical understanding of the principles and concepts involved in the preparation of financial statements. Students are exposed to a conceptual analytical approach with the aim of improving their critical thinking and communicative skills.

**LEVEL: I**

**SEMESTER: I OR II (FT & EU)**

**COURSE CODE: ACCT 1003**

**COURSE TITLE: INTRODUCTION TO COST AND MANAGERIAL ACCOUNTING**

**CREDITS: 3**

**PREREQUISITES: NONE**

**CO-REQUISITES: ACCT 1002**

**DEPARTMENT RESPONSIBLE: MANAGEMENT STUDIES**

**COURSE DESCRIPTION:** This is an introductory course for students of accounting as well as other areas of study. It aims to acquaint them with the uses of accounting information and techniques useful to the manager in planning, decision-making and controlling organizational activities.

**LEVEL: II**

**SEMESTER: I (FT) & II (EU only)**

**COURSE CODE: ACCT 2017**

**COURSE TITLE: MANAGEMENT ACCOUNTING I**

**CREDITS: 3**

**PREREQUISITES: ACCT 1002 AND ACCT 1003**

**DEPARTMENT RESPONSIBLE: MANAGEMENT STUDIES**

**COURSE DESCRIPTION:** The course explains how managerial accounting information is used by managers in manufacturing, retail, service and not-for-profit organisations to anticipate the future and monitor the activities of the business.

**LEVEL: I**  
**SEMESTER: I**  
**COURSE CODE: ECON 1001**  
**COURSE TITLE: INTRODUCTION TO MICROECONOMICS**  
**CREDITS: 3**  
**PREREQUISITES: None**  
**DEPARTMENT RESPONSIBLE: ECONOMICS**

**COURSE DESCRIPTION:** This course provides students to the history of economic thought highlighting some of the key economic issues, which have preoccupied the discipline from its origins. The course also provides an introduction to the basic principles of micro-economic analysis together with the main perspectives on the functioning of the macro-economy. The microeconomic analysis is illustrated by reference to a key export sector in the Caribbean (e.g. oil or bananas). The implications of trends in the latter for the Balance of Payments and macro economy conclude this first semester course.

**LEVEL: I**  
**SEMESTER: II**  
**COURSE CODE: ECON 1002**  
**COURSE TITLE: INTRODUCTION TO MACROECONOMICS**  
**CREDITS: 3**  
**PREREQUISITES: None**  
**CO-REQUISITES: ECON 1001**  
**DEPARTMENT RESPONSIBLE: ECONOMICS**

**COURSE DESCRIPTION:** This course emphasizes macroeconomic theory and policy and the related national income accounting together with international trade and the balance of payments. There is a significant stress on the implications of these economic issues for the Caribbean reality.

**LEVEL: II**  
**SEMESTER: I OR II (FT/EU IN BOTH SEMESTERS)**  
**COURSE CODE: MGMT 2008**  
**COURSE TITLE: ORGANISATIONAL BEHAVIOUR**  
**CREDITS: 3**  
**PREREQUISITES: SOCI 1002 OR MGMT 1001.**  
**PREREQUISITES FOR FFA & FST: SOCI 1002 OR MGMT 1001 OR AGEX 1000 OR COMP 1100 OR (COMP 1400 & COMP1401) OR HUEC 1003 OR CHEM 1060 OR (CHEM 1065 AND CHEM 1066)**  
**DEPARTMENT RESPONSIBLE: MANAGEMENT STUDIES**  
**COURSE DESCRIPTION:** This course uses the systems approach to organisations to highlight how interrelated variables such as people, technology, task, structure and external environments impact on organizational effectiveness. Emphasis is on the nature of behavioural issues and how and why they impact on the functioning of organisations.

**LEVEL: II**  
**SEMESTER: I**  
**COURSE CODE: MGMT 2012**  
**COURSE TITLE: QUANTITATIVE METHODS**  
**CREDITS: 3**  
**PREREQUISITES: ECON 1002 AND ECON 1003**  
**PREREQUISITES FOR FFA & FST: ECON 1002 AND EITHER CHEM 1060 OR (CHEM 1065 AND CHEM 1066) OR MATH 1140 OR (MATH 1141 AND MATH 1142)**  
**DEPARTMENT RESPONSIBLE: MANAGEMENT STUDIES**  
**COURSE DESCRIPTION:** This course is an introductory level survey of quantitative techniques commonly used to provide insight into business decisions. The primary emphasis is on preparing the student to become an intelligent user of these techniques.

**LEVEL: II**

**SEMESTER: I (FT/EU) or II (FT ONLY)**

**COURSE CODE: MGMT 2021**

**COURSE TITLE: BUSINESS LAW I**

**CREDITS: 3**

**PREREQUISITES: NONE**

**DEPARTMENT RESPONSIBLE: MANAGEMENT STUDIES**

**COURSE DESCRIPTION:** The main focus of this course is the general principles of the law of contract, the law of Agency as well as other related areas of interest like the Sale of Goods Act and the Hire Purchase Act 1938 and 1954. Background material covers the role and function of the law in society, the sources of the law, the legal system etc.

**LEVEL: II**

**SEMESTER: I or II (FT/EU IN BOTH SEMESTERS)**

**COURSE CODE: MGMT 2023**

**COURSE TITLE: FINANCIAL MANAGEMENT I**

**CREDITS: 3**

**PREREQUISITES: ACCT 1002 AND ECON 1003**

**PREREQUISITES FOR FFA & FST: ACCT 1002 AND EITHER CHEM 1060 OR (CHEM 1065 AND CHEM 1066) OR AGRI 1003 OR MATH 1140 OR (MATH 1141 AND MATH 1142)**

**DEPARTMENT RESPONSIBLE: MANAGEMENT STUDIES**

**COURSE DESCRIPTION:** This course is concerned with the core concepts of financial decision-making; the timevalue of money, the cost of capital and trade-offs between risk and return. Students should develop a thorough understanding of these basic concepts and how to apply them in real-world examples.

**LEVEL: II**

**SEMESTER: II (FT/EU)**

**COURSE CODE: MGMT 2032**

**COURSE TITLE: MANAGERIAL ECONOMICS**

**CREDITS: 3**

**PREREQUISITES: ECON 1001 AND ECON 1003 OR CHEM 1060 OR MATH 1140**

**PREREQUISITES FOR FFA & FST: ECON 1001 AND EITHER CHEM 1060 OR MATH 1140**

**DEPARTMENT RESPONSIBLE: MANAGEMENT STUDIES**

**COURSE DESCRIPTION:** This course is concerned with the application of economic principles and methodologies to the decision-making process of the business firm operating under conditions of risk and uncertainty. Emphasis is also placed on the firms' competitive strategy.

**LEVEL: II**

**SEMESTER: I (FT/EU) or II (FT ONLY)**

**COURSE CODE: MKTG 2001 (MGMT 2003)**

**COURSE TITLE: PRINCIPLES OF MARKETING**

**CREDITS: 3**

**PREREQUISITES: ACCT 1002 AND ECON 1001**

**PREREQUISITES FOR FFA & FST: ACCT 1002 AND AGBU 1005**

**DEPARTMENT RESPONSIBLE: MANAGEMENT STUDIES**

**COURSE DESCRIPTION:** This course is intended to provide students with the conceptual framework and analytical skills necessary for the analysis of markets and marketing activities of firms in a dynamic environment.

**LEVEL: III**

**SEMESTER: I (FT/EU)**

**COURSE CODE: MKTG 3000**

**COURSE TITLE: MARKETING MANAGEMENT**

**CREDITS: 3**

**PREREQUISITES: MKTG 2001 (MGMT 2003)**

**DEPARTMENT RESPONSIBLE: MANAGEMENT STUDIES**

**COURSE DESCRIPTION:** This course is concerned with the development of the student's marketing decision making skills and communication effectiveness. It is case-based, and students are expected to undertake a marketing project based on fieldwork.

THE UNIVERSITY OF THE WEST INDIES  
ST. AUGUSTINE CAMPUS

AQA 1-5

LIBRARY IMPACT STATEMENT

AVAILABILITY OF LIBRARY RESOURCES

*[In consultation with the Campus Librarian, explain whether library resources are available to support the proposed award course. If new library resources are required, detail these and give an estimate of the annual cost.]*

*[At its meeting on 17 March 2006 the Academic Quality Assurance Committee acting on behalf of the Academic Board, St. Augustine Campus, agreed to advise Faculties that the Campus Library be allowed to make assessments of proposals for new and/or major changes to courses/programmes.]*

Library Impact Statement (FOR INFORMATION ONLY)

**To:** Dr. Permanand Mohan  
**Department:** Computer and Information Technology  
**Course/Programme:** Revision of the B.Sc. Computer Science and B.Sc. Information Technology Programmes  
**Level:** Undergraduate


Thank you for the information provided in relation to your revised programme proposal. The collection assessment has now been completed. Resources required to support this revised programme have been identified as:

**Immediate:**  
\$ 7,000.00

**Future Years:**  
\$ 3,500.00

*Please note that once this programme has been approved a detailed booklist should be provided so that relevant materials can be purchased and made ready for student use. Further information can be found under Faculty Services at: <http://libraries.sta.uwi.edu/ajl/index.php/services>*

Thank you.

  
For the Campus Librarian

C: Dean, Faculty of Science and Technology  
Head, Department of Computer Science and Information Technology  
Deputy Principal/Chair, Academic Quality Assurance Committee  
Asst. Registrar (Secretariat)

**Date:** February 12, 2016

THE MAIN LIBRARY  
THE UNIVERSITY OF THE WEST INDIES  
ST. AUGUSTINE.

LIBRARY PROCEDURE FOR EVALUATION OF  
NEW TEACHING PROPOSALS

*To be completed by **library staff** in respect of **all** submissions by Faculty Boards to Academic Board for course and syllabus changes, new courses, subjects and higher degree programs or other proposals with library implications.*

Faculty: Faculty of Science and Agriculture

Department: Computing and IT

Academic Staff member(s) responsible: Dr. Permanand Mohan

Programme: BSc Computer Science (Special) / BSc Computer Science and Management /  
BSc General (Major in Computer Science) / BSc Information Technology (Special) / BSc General  
(Major in Information Technology)

Course Title (and No., if applicable) 38 new/revised courses - see below

Previous Title(s) [For substitutions only]:

	Course Code	Course Title
1.	COMP 1600	Introduction to Computing Concepts
2.	COMP 1601	Programming I
3.	COMP 1602	Programming II
4.	COMP 1603	Programming III
5.	COMP 1604	Mathematics for Computing
6.	COMP 2600	Data Structures
7.	COMP 2601	Operating Systems
8.	COMP 2602	Computer Architecture
9.	COMP 2603	Object-Oriented Programming I
10.	COMP 2604	Computer Networks
11.	COMP 2605	Enterprise Database Systems
12.	COMP 2606	Software Engineering I
13.	COMP 3600	Software Engineering II
14.	COMP 3601	Design and Analysis of Algorithms
15.	COMP 3602	Theory of Computing
16.	COMP 3603	Human-Computer Interaction
17.	COMP 3605	Introduction to Data Analytics
18.	COMP 3606	Wireless and Mobile Computing
19.	COMP 3607	Object-Oriented Programming II
20.	COMP 3608	Intelligent Systems
21.	COMP 3609	Game Programming
22.	COMP 3610	Big Data Analytics
23.	COMP 3611	Modelling and Simulation
24.	INFO 1600	Introduction to Information Technology Concepts
25.	INFO 1601	Introduction to WWW Programming
26.	INFO 2600	Information Systems Development
27.	INFO 2601	Networking Technologies Fundamentals
28.	INFO 2602	Web Programming and Technologies I

29.	INFO 2603	Platform Technologies I
30.	INFO 2604	Information Systems Security
31.	INFO 2605	Professional Ethics and Law
32.	INFO 3601	Platform Technologies II
33.	INFO 3602	Web Programming and Technologies II
34.	INFO 3600	Business Information Systems
35.	INFO 3605	LAN Technologies
36.	INFO 3606	Cloud Computing
37.	INFO 3607	WAN Technologies
38.	INFO 3608	E-Commerce

## **Areas of Investigation**

### **1. Extent of 'newness' of course [Tick one]**

- ☐ Totally new subject  
☒ Area within an existing subject  
☐ Extension in depth of an existing subject

### **2. Existing holdings of library material**

#### **Print Resources**

##### **Catalogue check**

Subject(s) scanned

Approx. No. of titles held (books and serials)

a) Programming_____	>1000 Book (>95% > 3 yrs) and >20 serials
b) Web Programming_____	>300 book (>66% >5 yrs) and 2 serial
c) Computer Science_____	>3000 book (>66% >5 yrs) and 50 serial titles
d) Database management_____	>300 book (>33%>5yrs) and 10 serial titles
e) User interface design_____	~30 books (>15% >3 yrs) and 0 serials
f) Data management_____	~700 books (>95% > 3 yrs) and ~20 serials
g) Networking_____	~400 books (>60% > 3 yrs) and 6 serials
h) Critical Thinking_____	~300 books (>66% >5 yrs ) and 0 serials
i) Data management_____	~700 books (>95% > 3 yrs) and ~20 serials
j) Problem solving_____	> 400 books (> 90% >5 yrs) and 0 serials

#### **Electronic Resources**

##### **E-book Collections**

The topic of Computer science and subtopics are well covered the

Subject(s) scanned	Approx. No. of titles held
Sample from MyiLibrary	
a) Programming_____	137 e-books
b) Critical Thinking_____	230 e-books
c) Data management_____	264 e-books
d) Problem solving_____	214 e-books
e) networking_____	82 e-books
f) User interface design_____	139 e-books

In the CRCnetBase – 163 e-books in IT

##### **E-journals**

For Computer Science there are >500 e-journals available

For Information Technology there are 1,366 e-journals available



## Databases

This topic is well-subscribed to in the core databases: Compendex, Computer and Applied Sciences Complete (Ebscohost), Wiley, OUP, NTIS, Inspec, IEEE, Web of Knowledge, ACM Digital Library, Computing (Proquest)

### 3. Publishing in Subject

Indicate approx. No. of titles currently in print in the field 400,000  
Indicate Approx. No. of titles published in the last year in the field 75,000  
Approx. No. of journal titles 10,000

### 4. List of Required Information Resources submitted by Faculty

List was received ☒ yes ☐ no

No of titles on List 64

No. of titles on List in library 50

### 5. Additional needs for material

(To be completed when Academic Staff member(s) indicate that existing holdings are limited/inadequate, Very Poor or Non-existent or when it is necessary to update the collection)

	Titles	Copies	Cost Estimate TTS	Total TTS
<u>Immediate:</u>				
Books	10	X 1	X 500.00	= 7,000.00
Journal subs.		X		=
E-books		X	X	=
Databases		X		=
Total estimated cost				7,000.00
<u>Future Years:</u>				
Books	7	X 1	X 500.00	= 3,500.00
Journal subs.		X		=
E-books		X	X	=
Databases		X		=
Total estimated cost				3,500.00

### 6. Remarks:

(Consult with faculty on desirable extent of additions (immediate and future) in relation to findings above, levels of course and projections of program.)

The materials in the library was considered adequate but there were 14 items on the reading list which we did not have or were not up to date.

Signed: [Signature]  
(Librarian Submitting Evaluation)

Date Submitted to Head, User Services: 2016 - Feb - 12

### 7. Additional Needs (Staff /Space/Equipment)

Based on the extent of new ordering and processing of material indicated above and the size of the additional student intake, the Head, User services will determine the need for additional library staff, space and equipment needs to be detailed below.

Signed: [Signature]  
(Head, User Services)

Date Submitted to Campus Librarian: 2016 Feb - 12

# Resources check for the DCIT proposal

## If one form (duplicate items excluded)

Items on list	64
Accessible items	50
Inaccessible items	14

## If separate forms

No.	Course	No. of items on list	No. of accessible items on list	No. of inaccessible items on list
1	COMP 1600 Introduction to Computing Concepts	1	1	0
2	COMP 1601 Computer Programming I	1	1	0
3	COMP 1602 Programming II	2	2	0
4	COMP 1603 Programming III	1	1	0
5	COMP 1604 Mathematics for Computing	1	1	0
6	COMP 2600 Data Structures	1	1	0
7	COMP 2601 Operating Systems	3	2	1
8	COMP 2602 Computer Architecture	3	3	0
9	COMP 2603 Object-Oriented Programming I	4	2	2
10	COMP 2604 Computer Networks	5 (count includes item with access code)	4	1
11	COMP 2605 Enterprise Database Systems	1	1	0
12	COMP 2606 Software Engineering I	1	1	0
13	COMP 3600 Software Engineering II	1	1	0
14	COMP 3601 Design and Analysis of Algorithms	3	3	0
15	COMP 3602 Theory of Computing	2	1	1
16	COMP 3603 Human-Computer Interaction	4	3	1
17	COMP 3605 Introduction to Data Analytics	2	2	0
18	COMP 3606 Wireless and Mobile Computing	3	0	3

19	COMP 3607 Object-Oriented Programming II	4	2	2
20	COMP 3608 Intelligent Systems	2	1	1
21	COMP 3609 Game Programming	2	1	1
22	COMP 3610 Big Data Analytics	2	1	1
23	COMP 3611 Modelling and Simulation	1	1	0
24	INFO 1600 Introduction to Information Technology Concepts	1	1	0
25	INFO 1601 Introduction to WWW Programming	2	2	0
26	INFO 2600 Information Systems Development	2	2	0
27	INFO 2601 Networking Technologies Fundamentals	4	3	1
28	INFO 2602 Web Programming and Technologies I	1	0	1
29	INFO 2603 Platform Technologies 1	0	-	-
30	INFO 2604 Information Systems Security	1	1	0
31	INFO 2605 Professional Ethics and Law	3	3	0
32	INFO 3600 Business Information Systems	2	2	0
33	INFO 3601 Platform Technologies II	3	3	0
34	INFO 3604 Information Technology Project	0	-	
35	INFO 3605 Fundamentals of LAN Technologies	2	2	0
36	INFO 3606 Cloud Computing	0	-	-
37	INFO 3607 Fundamentals of WAN Technologies	2	2	0
38	INFO 3608 E-Commerce	1	1	0
39	MATH 1115 Fundamental Mathematics for the General Sciences I	1	1	0
40	MATH 2250 Industrial Statistics I	1	1	0

#### COMP 1600 Introduction to Computing Concepts

No.	Course	Citation provided by Department	Call number and latest edition in The	Bibliographic details for order suggestion
-----	--------	---------------------------------	---------------------------------------	--

			Alma Jordan Library	
1	COMP 1600 Introduction to Computing Concepts	Brookshear, J Glen. 2014. Computer Science: An Overview. Addison Wesley. Most recent edition.	QA76.B743 2012, edition 11	Brookshear, J Glen. 2014. <i>Computer science: An overview</i> . Pearson. Edition 12. ISBN 9780133760064. US\$159.80. QA76.B743 2012

#### COMP 1601 Computer Programming I

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library
1	COMP 1601 Computer Programming I	Kalicharan, Noel. 2005. C Programming - A Beginner's Course. Sept. Amazon/CreateSpace.	QA76.73.C15 K355 2008

#### COMP 1602 Programming II

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library
1	COMP 1602 Programming II	Kalicharan, Noel. 2005. C Programming - A Beginner's Course. Sept. Amazon/CreateSpace.	QA76.73.C15 K355 2008
2	COMP 1602 Programming II	Kalicharan, Noel. 2006. C Programming - An Advanced Course. Amazon/CreateSpace.	QA76.73.C15 K35 C66 2006

#### COMP 1603 Programming III

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library
1	COMP 1603 Programming III	Kalicharan, Noel. 2006. C Programming - An Advanced Course. Amazon/CreateSpace.	QA76.73.C15 K35 C66 2006

### COMP 1604 Mathematics for Computing

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library
1	COMP 1604 Mathematics for Computing	Lpp, S. 2010. Discrete Mathematics with Applications. Brooks Cole, 4th edition.	QA39.3.L65 2011, edition 4

### COMP 2600 Data Structures

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library
1	COMP 2600 Data Structures	Kalicharan, Noel. Data Structures in C. CreateSpace/Amazon	QA76.73 C15 K35 D3 2008

### COMP 2601 Operating Systems

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library	Bibliographic details for order suggestion
1	COMP 2601 Operating Systems	Silberschatz, Abraham, Peter B. Galvin, Greg Gagne. 2013. Operating System Concepts Essentials, 2nd Ed, Wiley. ISBN: 1118804929.	not held	Silberschatz, Abraham et al. 2013. <i>Operating System Concepts Essentials</i> . 2nd edition. Wiley. ISBN: 9781118804926. US\$137.75.
2	COMP 2601 Operating Systems	Stallings, William 2014. Operating Systems: Internals and Design Principles, 8th Ed, Prentice Hall. ISBN: 0133805913.	QA76.77.S73 2015, edition 8	no later edition
3	COMP 2601 Operating Systems	Tanenbaum, Andrew S., Herbert Bos. 2014. Modern Operating Systems, 4th Ed, Prentice Hall. ISBN: 013359162X.	QA76.76.O63 T359 2015, edition 4	no later edition

### COMP 2602 Computer Architecture

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library	Bibliographic details for order suggestion
1	COMP 2602 Computer Architecture	Stallings, William. 2006. Computer Organization and Architecture: Designing for Performance, Addison-Wesley 7th edition.	QA76.9.C643 S73 2016, edition 10	no later edition
2	COMP 2602 Computer Architecture	Comer, Douglas E. 2005. Essentials of Computer Architecture, Prentice Hall.	QA76.9.A73 C595 2005, [edition 1]	no later edition
3	COMP 2602 Computer Architecture	Hennessy, J. L. and D. A. Patterson. 2012. Computer Architecture: A Quantitative Approach, 5th Edition, Morgan Kaufmann Publishing, Menlo Park, CA.	QA76.9.A73 P377 2003, edition 3	Hennessy, John L. and David A. Patterson. 2011. <i>Computer architecture: A quantitative approach</i> . 5th edition. Morgan Kaufmann. ISBN 9780123838728. US\$89.95. QA76.9.A73 P377 2003

### COMP 2603 Object-Oriented Programming I

No.	Course	Citation provided by Department (corrected where necessary)	Call number and latest edition in The Alma Jordan Library	Bibliographic details for order suggestion
1	COMP 2603 Object Oriented Programming I	Mohan, Permanand. 2013. Fundamentals of Object-Oriented Programming in Java, CreateSpace Independent Publishing Platform. 1st edition. ISBN-10: 1482587521, ISBN-13: 978 1482587524.	QA76.73 J38 M64 2013	no later edition
2	COMP 2603 Object-Oriented Programming I	Deitel, Harvey and Paul Deitel. 2002. Java: How to Program. 5th edition. Prentice-Hall.	QA76.73 J38 D45 1999, edition 3	Deitel, Paul and Harvey Deitel. 2014. <i>Java: How to program: Early objects</i> . 10th edition. Prentice Hall. ISBN 9780133807806. US\$166.40.

				Deitel, Paul and Harvey Deltel. 2014. <i>Java: How to program: Late objects version</i> . 10th edition. Prentice Hall. ISBN 9780132575652. US\$171.40.  to be confirmed by Department
3	COMP 2603 Object-Oriented Programming I	Eckel, Bruce. 2002. <i>Thinking in Java</i> , 3rd Edition. Prentice-Hall. Free electronic copy available on the Internet from <a href="http://www.mindview.net/Books/TIJ/">http://www.mindview.net/Books/TIJ/</a> .	not held	Eckel, Bruce. 2006. <i>Thinking in Java</i> . 4th edition. Prentice Hall. ISBN 9780131872486. US\$74.99.
4	COMP 2603 Object-Oriented Programming I	Budd, Timothy. 1999. <i>Understanding Object-Oriented Programming With Java</i> , Updated Edition. Addison Wesley.	not held	Budd, Timothy. 2000. <i>Understanding Object-Oriented Programming With Java</i> . Updated edition. Addison-Wesley. ISBN 9780201612738. US\$77.00.

#### COMP 2604 Computer Networks

No.	Course	Citation provided by Department (corrected where necessary)	Call number and latest edition in The Alma Jordan Library	Comments
1	COMP 2604 Computer Networks	Kurose, J. and K. Ross. 2012. <i>Computer Networking A Top-Down Approach</i> featuring the Internet. Addison Wesley. 6th edition. International edition. ISBN-10: 0273768964 ISBN-13: 978-0273768968	TK5105.875.I57 K88 2013, edition 6	no later edition
2	COMP 2604 Computer Networks	Slides and other resources from the Kurose Ross Text (Access Code needed). <a href="http://www.pearsoninternationaleditions.com/Sitemap/kurose-ross/">http://www.pearsoninternationaleditions.com/Sitemap/kurose-ross/</a>	not accessible	access code not retained in Library's copies
3	COMP 2604 Computer Networks	Tanenbaum, Andrew S. and David J. Wetherall. 2010. <i>Computer Networks</i> . 5th edition. Prentice Hall. ISBN-10: 0132126958. ISBN-13: 978-0132126953.	TK5105.5.T36 2011, edition 5	no later edition
4	COMP 2604 Computer Networks	Stallings, William. 2013. <i>Data and Computer Communications</i> . 10th edition. Prentice Hall. ISBN-13: 978-0133506488	TK5105 .S73 2014, 10th edition	no later edition

5	COMP 2604 Computer Networks	Pitt, Esmond. 2010. Fundamental Networking in Java. Springer. ISBN-10: 1849965455. ISBN-13: 978-1849965453.	QA76.73.J38 P578 2006 and QA76.73.J38 P578 2006eb	no later edition
---	-----------------------------	---	---	------------------

#### COMP 2605 Enterprise Database Systems

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library
1	COMP 2605 Enterprise Database Systems	Rob, Peter et. al. Database Systems Design, Implementation and Management (10th edition or international edition).	QA76.9.D26 C67 2015, 11th edition

#### COMP 2606 Software Engineering I

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library	Bibliographic details for order suggestion
1	COMP 2606 Software Engineering I	Sommerville Ian A. 2015. Software engineering 10th edition, Pearson	QA76.758.S657 2011, edition 9	Sommerville, Ian A. 2015. <i>Software engineering</i> . 10th edition. Pearson. ISBN 9780133943030. US\$186.80. QA76.758.S657 2011

#### COMP 3600 Software Engineering II

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library	Bibliographic details for order suggestion
1	COMP 3600 Software Engineering II	Sommerville Ian A. 2015. Software engineering 10th edition. Pearson	QA76.758.S657 2011, edition 9	Sommerville, Ian A. 2015. <i>Software engineering</i> . 10th edition. Pearson. ISBN 9780133943030. US\$186.80. QA76.758.S657 2011

#### COMP 3601 Design and Analysis of Algorithms



No.	Course	Citation provided by Department (corrected where necessary)	Call number and latest edition in The Alma Jordan Library
1	COMP 3601 Design and Analysis of Algorithms	Levitin, [Anany]. 2007. [Introduction to] The Design and Analysis of Algorithms. 2nd edition. Addison Wesley. ISBN:0-321- 36413-9	QA76.9.A43 L48 2012, edition 3
2	COMP 3601 Design and Analysis of Algorithms	Cormen, Thomas H. 2009. Introduction to algorithms. MIT Press.	QA76.6.C662 2009, edition 3
3	COMP 3601 Design and Analysis of Algorithms	Alsuwaiyel, M. H. 1999. Algorithms: Design techniques and analysis. World Scientific Publishing.	QA76.9.A43 A47 1999

### COMP 3602 Theory of Computing

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library	Bibliographic details for order suggestion
1	COMP 3602 Theory of Computing	Webber, Adam. 2008. Formal language: a practical introduction. Franklin, Beedle & Associates. ISBN-13: 978-1- 59028-197-0	not held	Webber, Adam Brooks. 2008. <i>Formal language: A practical introduction</i> . Franklin, Beedle & Associates. ISBN: 9781590281970. US\$70.00.
2	COMP 3602 Theory of Computing	Sipser, Michael. 2006. Introduction to the theory of Computation. Thompson Course Technology. ISBN-13: 978-1133187790	QA267.S56 2006, edition 2	Sipser, Michael. 2012. <i>Introduction to the theory of computation</i> . 3rd edition. Cengage Learning. ISBN 9781133187790. US\$261.95. QA267.S56 2006

### COMP 3603 Human-Computer Interaction

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library	Bibliographic details for order suggestion
1	COMP 3603 Human-	Dix, A. et al. 2004. Human- Computer Interaction. Harlow,	QA76.9.H85 H856 2004, edition 3	no later edition

	Computer Interaction	England: Prentice Hall. ISBN-10: 0130461091		
2	COMP 3603 Human-Computer Interaction	Rogers, Yvonne, Helen Sharp, Jenny Preece. 2011. <i>Interaction Design: Beyond Human Computer Interaction</i> , 3rd Edition, Wiley. ISBN-10: 0470665769	QA76.9.H85 P72 2011, edition 3	no later edition
3	COMP 3603 Human-Computer Interaction	Lazar, J. 2005. <i>Web Usability: A User-Centered Design Approach</i> , Addison Wesley. ISBN:0321321359	TK5105.H8.L3975 2006, [edition 1]	no later edition
4	COMP 3603 Human-Computer Interaction	Unger, Russ, Carolyn Chandler. 2012. <i>A Project Guide to UX Design: For user experience designers in the field or in the making</i> . 2nd edition, New Riders. ISBN-13:978-0321815385, ISBN-10:0321815386	not held	Unger, Russ and Carolyn Chandler. 2012. <i>A project guide to UX Design: For user experience designers in the field or in the making</i> . 2nd edition, New Riders. ISBN 9780321815385. US\$44.99.

#### COMP 3605 Introduction to Data Analytics

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library
1	COMP 3605 Introduction to Data Analytics	Han, Jiawei. 2011. <i>Data Mining Concepts and Techniques</i> (3rd Edition), Morgan Kaufmann. ISBN: 978-0-12-381479-1 [Recommended Text]	QA76.9 .D343 H36 2012eb, edition 3
2	COMP 3605 Introduction to Data Analytics	Flach, Peter. 2012. <i>Machine Learning: The Art and Science of Algorithm that make sense of data</i> , Cambridge University Press.	Q325.5.F5 2012, [edition 1]

#### COMP 3606 Wireless and Mobile Computing

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library	Bibliographic details for order suggestion
1	COMP 3606	Meier, Reto. 2012. <i>Professional Android 4 Application</i>	not held	Meier, Reto. 2012. <i>Professional Android 4</i>

	Wireless and Mobile Computing	Development. Edition: 3. Publisher: Wrox. ISBN-10: 1118102274, ISBN-13: 978-118102275.		<i>Application Development.</i> Edition: 3. John Wiley. ISBN 9781118102275. US\$44.99.
2	COMP 3606 Wireless and Mobile Computing	Deitel, Paul, Harvey Deitel and Abbey Deitel. 2014. <i>Android for Programmers: An App-Driven Approach.</i> Edition: 2. Prentice Hall. 0133570924, ISBN-13: 9780133570922.	not held	Deitel, Paul et al. 2015. <i>Android 6 for Programmers: An App-Driven Approach.</i> Edition: 3. Prentice Hall. ISBN 9780134289366. US\$44.99.
3	COMP 3606 Wireless and Mobile Computing	Haseman, Chris. 2011. <i>Creating Android Applications, Develop and Design.</i> Peachpit Press. 1 edition. ISBN-10: 032178409X. ISBN-13: 978-0321784094.	not held	Haseman, Chris. 2011. <i>Creating Android Applications, Develop and Design.</i> Peachpit Press. ISBN 9780321784094. US\$39.99.

#### COMP 3607 Object-Oriented Programming II

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library	Bibliographic details for order suggestion
1	COMP 3607 Object-Oriented Programming II	Mohan, Permanand. 2013. <i>Fundamentals of Object Oriented Programming in Java.</i> CreateSpace Independent Publishing Platform. 1st Edition. ISBN 10: 1482587521, ISBN 13: 978-1482587524.	QA76.73 .J38 M54 2013	no later edition
2	COMP 3607 Object-Oriented Programming II	Deitel, Harvey and Paul Deitel. 2002. <i>Java: How to Program</i> , 5th Edition. Prentice-Hall.	QA76.73.J38 D45 1999, edition 3	Deitel, Paul and Harvey Deitel. 2014. <i>Java: How to program: Early objects.</i> 10th edition. Prentice Hall. ISBN 9780133807806. US\$166.40.  Deitel, Paul and Harvey Deitel. 2014. <i>Java: How to program: Late</i>

				<i>objects version</i> . 10th edition. Prentice Hall. ISBN 9780132575652. US\$171.40.  to be confirmed by Department
3	COMP 3607 Object- Oriented Programming II	Eckel, Bruce. 2002. <i>Thinking in Java</i> , 3rd edition. Prentice-Hall. Free electronic copy available on the Internet from <a href="http://www.mindview.net/Books/TIJ/">http://www.mindview.net/Books/TIJ/</a>	not held	Eckel, Bruce. 2006. <i>Thinking in Java</i> . 4th edition. Prentice Hall. ISBN 9780131872486. US\$74.99.
4	COMP 3607 Object- Oriented Programming II	Budd, Timothy. 1999. <i>Understanding Object-Oriented Programming With Java</i> , Updated Edition. Addison Wesley.	not held	Budd, Timothy. 2000. <i>Understanding Object-Oriented Programming With Java</i> . Updated edition. Addison-Wesley. ISBN 9780201612738. US\$77.00.

#### COMP 3608 Intelligent Systems

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library	Bibliographic details for order suggestion
1	COMP 3608 Intelligent Systems	Grosan, Crina and Ajith Abraham. 2011. <i>Intelligent Systems: 17 (Intelligent Systems Reference Library)</i> , Springer.	not held	Grosan, Crina and Ajith Abraham. 2011. <i>Intelligent systems: A modern approach</i> . Springer. ISBN 9783642210037. US\$179.00.
2	COMP 3608 Intelligent Systems	Flach, Peter. 2012. <i>Machine Learning: The Art and Science of Algorithms that Make Sense of Data</i> . Cambridge, ISBN-13:960-1200462906 ISBN-10:1107422221	Q325.5.F5 2012, [edition 1]	no later edition

### COMP 3609 Game Programming

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library	Bibliographic details for order suggestion
1	COMP 3609 Game Programming	Brackeen, D. 2004. <i>Developing Games in Java</i> . Berkeley, California: New Riders Publishing.	QA76.76.C672 B74 2004	no later edition
2	COMP 3609 Game Programming	Davison, A. 2005. <i>Killer Game Programming in Java</i> . Sebastopol, California: O'Reilly Media. Book Web Site: <a href="http://five.dots.coe.psu.ac.th/~ad/jg/">http://five.dots.coe.psu.ac.th/~ad/jg/</a>	not held	Davison, Andrew. 2005. <i>Killer game programming in Java</i> . O'Reilly Media. US\$59.99. ISBN 9780596007300.

### COMP 3610 Big Data Analytics

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library	Bibliographic details for order suggestion
1	COMP 3610 Big Data Analytics	Ozsu, M. Tamer, Valduriez, Patrick. <i>Principles of Distributed Database Systems</i> (third edition), Springer. ISBN: 9781441988331	QA76.9.D3 O956 1991, [edition 1]	Özsu, M. Tamer and Patrick Valduriez. 2011. <i>Principles of distributed database systems</i> . Third edition. Springer. ISBN 9781441988331. US\$119.00. QA76.9.D3 O956 1991
2	COMP 3610 Big Data Analytics	White, Tom. <i>Hadoop: The Definitive Guide</i> (4th Edition), O'Reilly Media. ISBN-10: 1491901632	not held	White, Tom. 2015. <i>Hadoop: The definitive guide</i> . 4th edition, O'Reilly Media. ISBN 9781491901632. US\$49.99.

### COMP 3611 Modelling and Simulation

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library
1	COMP 3611 Modelling and Simulation	Ranks, Carson, Nelson, and Nicol. 2005. Discrete-Event System Simulation (Fourth Edition), Prentice-Hall.	T57.62.D53 2010, edition 5

### INFO 1600 Introduction to Information Technology Concepts

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library	Bibliographic details for order suggestion
1	INFO 1600 Introduction to Information Technology Concepts	Laudon, Kenneth C. and Jane P. Laudon. Management Information Systems (11th edition). Management Information Systems, Effy Oz. Thomson.	T58.6.L376 2012, edition 12	Laudon, Kenneth C. and Jane P. Laudon 2015. <i>Management information systems: Managing the digital firm</i> . 14th edition. Pearson. ISBN 9780133898163. US\$276.20. T58.6.L376 2012

### INFO 1601 Introduction to WWW Programming

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library	Bibliographic details for order suggestion
1	INFO 1601 Introduction to WWW Programming	Deitel, Paul et al. 2011. Internet and World Wide Web How To Program (5th edition)	QA76.625.D47 2008, edition 4	Deitel, Paul et al. 2011. <i>Internet and World Wide Web How To Program</i> . Pearson. 5th edition. ISBN 9780132151009. US\$154.40. QA76.625.D47 2008
2	INFO 1601 Introduction to WWW Programming	Reed, D.2005. A Balanced Introduction to Computer Science. Upper Saddle River, NJ: Pearson.	QA76 .R395 2011, edition 3	no later edition

### INFO 3607 Fundamentals of WAN Technologies

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library
1	INFO 3607 Fundamentals of WAN Technologies	Dye, Mark A. and Antoon W. Ruff. Network Fundamentals: CCNA Exploration Companion Guide.	QA76.3 .D94 2007
2	INFO 3607 Fundamentals of WAN Technologies	Vachon, Bob. Accessing the WAN: CCNA Exploration Companion Guide.	QA76.3.V334 2007

### INFO 3608 E-Commerce

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library	Bibliographic details for order suggestion
1	INFO 3608 E-Commerce	Laudon K., C. G. Traver, E-Commerce 2012, 8/E, ISBN-10:0136100570, ISBN-13:978-0136100577, Prentice Hall	HF5548.32.L38 2011, edition 7	Laudon Kenneth and Carol G. Traver. 2015. <i>E-commerce 2016: Business, technology, society</i> . 12th edition. ISBN 9780133938951. US\$245.80. ISBN 9780133938951. HF5548.32.L38 2011

### MATH 1115 Fundamental Mathematics for the General Sciences I

No.	Course	Citation provided by Department (corrected where necessary)	Call number and latest edition in The Alma Jordan Library
1	MATH 1115 Fundamental Mathematics for the General Sciences I	Jogie, D. and D. Comissiong. [n.d.] Fundamental Mathematics for the General Sciences. (Available in the Math Department. Cost: \$150)	QA37.3.D39

### MATH 2250 Industrial Statistics I

No.	Course	Citation provided by Department	Call number and latest edition in The Alma Jordan Library
1	MATH 2250 Industrial Statistics I	Devore, Jay L. Probability and statistics for Engineering and the Sciences.	edition 9 (2015) in Cataloguing

Feb 11, 2016