Philadelphia University

Faculty of Information technology

Department of Computer Science

Course Description

January, 2018

CONTENT

CHAPTER 1

INTRODUCTION TO CURRICULUM DESIGN

This Handbook contains a set of module descriptions and some information on the curriculum design and organisation that mostly follow the report of the Higher Education Accreditation Commission in Jordan (HEAC) and the Computing Curricula 2013 (CC2013). The CC2013 is a joint undertaking of the Computer Society of the Institute for Electrical and Electronic Engineers (IEEE-CS) and the Association for Computing Machinery (ACM) that developed curricular guidelines for undergraduate programs in computing.

These modules are offered at the Department of Computer Science, Faculty of Information Technology/ Philadelphia University, to obtain the four years B.Sc. (honour) degree in Computer Science (CS).

The information given in this Handbook is extracted for the Program Specifications for the Degree programme. These specifications are published separately.

1.1 Fundamental Concepts

The most important concepts for understanding the module descriptions are as follows:

- **The CS Body of Knowledge.** The modules described in this Handbook are defined in relation to a general taxonomy of that portion of Computer Science appropriate for an undergraduate curriculum. That taxonomy represents the body of knowledge for Computer Science. The body of knowledge is organised hierarchically into three levels. The highest level of the hierarchy is the **area**, which represents a particular disciplinary sub-field. The areas are broken down into smaller divisions called **units**, which represent individual thematic modules within an area. Each unit is further subdivided into a set of **topics**, which are the lowest level of the hierarchy.
  For coding the modules, the Department of CS has applied the following scheme of coding. Each area is identified by a one-digit number, such as 1 for Programming or 3 for Architecture and Organisation. Each unit is identified by adding a numeric suffix to the area number; as an example, 31 is a unit Logic Circuits Design of the area "Architecture and Organisation".
  The whole computer science areas are listed in Table (1-1).

Table (1-1) The Areas of Computer Science

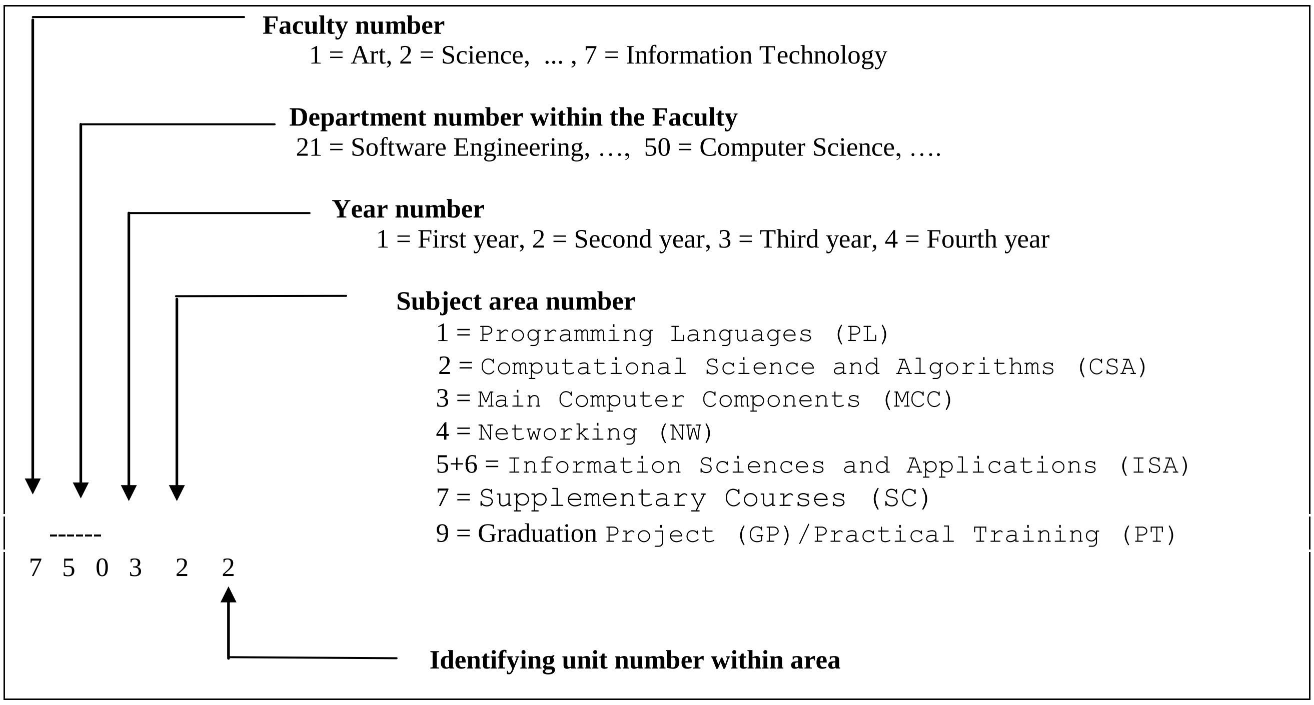
No. Of KA	Name of Knowledge Areas
1.	Programming Languages (PL)
2.	Computational Science and Algorithms (CSA)
3.	Main Computer Components (MCO)
4.	Networking (NW)
5 + 6.	Information Sciences and Applications (ISA)
7.	Supplementary Courses (SC)
9.	Graduation Project (GP) / Practical Training (PT)

- **Core and Elective Units**. Given the expanding scope of the computing discipline, it is impossible to insist that every undergraduate learn all the topics that were at one time considered fundamental to the field. The CC2013 report defines a minimal set of core units for which there is a broad consensus that the material is essential to anyone obtaining an undergraduate degree in computer science. Because the core is defined as minimal, the core alone cannot constitute a complete undergraduate curriculum. The undergraduate program must include additional elective units from the body of knowledge. These elective units could be chosen according to the needs of the individual student. Note that, occasionally, timetabling difficulties restricts elective units.

- **Credit Hours.** To give a sense of the time required to cover a particular unit, a time metric should be chosen. The system of study at Philadelphia University is based on the credit hours. The basic measure unit of the curriculum is 3 credit hours module (or course unit). A module, which delivers at least 3 hours per week of lectures or tutorial time, is worth 3 credit hours. Some modules may also provide an extra 1-hour per week for laboratory, but the module is still classified as 3 credit hours. In general, over a 16 weeks semester, a typical module provides minimum 45 hours of contact time. The final week of the semester is used for the examinations. The contact time corresponds to the in-class time required to present the material in a traditional lecture oriented format. Note that this time does not include the instructor's preparation time or the time students spend outside of class. As a general guideline, the time required outside of class is twice the time of the in-class time. Thus, a unit that is listed as requiring 3 credit hours will typically entails a total of 9 hours (3 in class and 6 outside). It is also important to keep in mind that the time associated with each unit represents the minimum number of hours required for adequate coverage, and that it is always appropriate to spend more time than the listed minimum.

1.2 Format of the Module Coding Adopted

Each module in the CS programme is identified by a code and a title. For example, **"750322 Design and Analysis of Algorithms"** represents a module offered by Faculty of Information Technology, Department of Computer Science in the third year, in the area of Algorithms and Computation Theory, and the module title is Design and Analysis of Algorithms. Figure (1-1) illustrates the scheme of module coding and numbering, where the Design and Analysis of Algorithms module is presented as an example.



Faculty number
1 = Art, 2 = Science, ... , 7 = Information Technology
Department number within the Faculty
21 = Software Engineering, …, 50 = Computer Science, ….
Year number
1 = First year, 2 = Second year, 3 = Third year, 4 = Fourth year
Subject area number
1 = Programming Languages (PL)
2 = Computational Science and Algorithms (CSA)
3 = Main Computer Components (MCC)
4 = Networking (NW)
5+6 = Information Sciences and Applications (ISA)
7 = Supplementary Courses (SC)
9 = Graduation Project (GP)/Practical Training (PT)
------
7 5 0 3 2 2
Identifying unit number within area

Figure (1-1) Module Coding and Numbering Scheme

CHAPTER 2

CURRICULUM DESIGN, ORGANISATION, AND CONTENT

2.1 Outlines of the Degree Programme

Within the general area of Computer Science (CS), the modules recognise several major subject themes. This represents fundamental material on programming, algorithms and software engineering, the structure and operation of computer systems including a high-level view of processing, memory, data communication and input/output devices, plus operating systems and compilers, graphics and user interfaces. This includes the theoretical foundations of computing, including programming languages and formal analysis of algorithms and machines.
Details of each module are set out in Chapter (3).

2.2 Requirements for the Degree Programme

The CS programme is covered with different requirements. For obtaining the full award, students should complete 46 modules, each of 3 credit hours (except the Practical Training with zero hour, the Research Project (1) with one hour, and the Research Project (2) with two hours), i.e. a total of 132 credit hours summarised as follows:

-	9 modules (University requirements)	(27 credit hours)	(20.45 %)
-	8 modules (Faculty requirements)	(24 credit hours)	(18.18 %)
-	17 modules (Departmental Compulsories)	(48 credit hours)	(36.36%)
-	3 modules (Departmental Electives)	(9 credit hours)	(6.81 %)
-	8 modules (Supportive Compulsory modules)	(24 credit hours)	(18.18 %)

The Faculty requirements and University requirements include some computer-oriented modules that account to the Department requirements. (See Chapter (3), Table (3-1) for the titles of these modules).

2.3 Design, Organisation, and Content of Curriculum

- ***Organisation of Modules:*** The modules are organised into three levels according to the year at which they occur in the curriculum:
  1- Level 1: **Introductory** modules,
  2- Level 2: **Intermediate** modules,
  3- Level 3: **Advanced** modules.

  Modules designated as **Introductory** are offered in the first and second years of the Department curriculum. Modules listed as **Intermediate** are usually offered in the second or third year and build a foundation for further study in the field. Modules designated as **Advanced** tend to be taken in later years (third and fourth) and focus on those topics that require significant preparation in the earlier coursework. For these modules, the Department wishes to orient such modules to its own areas of expertise**.**

  While these distinctions are easy to understand in their own right, it is important to recognise that there is no necessary relationship between the notions of core and elective - which apply to units in the body of knowledge - and the level of the module. The introductory and intermediate modules concentrate on core material, and the advanced modules include some core material and elective modules.

  The point of organising the modules into three levels: **Introductory**, **Intermediate**, and **Advanced** is to provide natural boundaries for defining implementation strategies. The CC2013 report defined many strategies. Figure (2-1) shows these strategies and their relationship in the curriculum.

Figure (2-1) Module Levels and Implementation Strategies

Introductory Modules	Imperative First	Object First	Functional First	Breadth First	Algorithmic First	Hardware First
Intermediate Modules	Topic-Based Approach	Compressed Approach	Systems-Based Approach	Web-Based Approach		
Advanced Modules	Additional modules used to complete the undergraduate program					

- For Introductory Modules, the Department adopted the **Imperative-First (or Procedural-First) strategy**. The imperative language is C++. Then C# is adopted to introduce Object Oriented concepts.
- For Intermediate Modules, the Department adopted **Topic-Based strategy** to preparing for specific areas.
- Some Advanced Modules are selected to attend the departmental objectives and the areas of expertise.

The CS programme is organised to cover some specified areas selected from the general areas listed in Table (1-1). Table (2-1) shows the areas covered by the specialisation Modules (including those computer-oriented modules taken from the Faculty and University requirements) and the number of modules in each of them. Note that the ratios in Table (2-1) are calculated according to the total number of modules (i.e. 46).

Table (2-1) Specialisation Areas

Area		Compulsory Modules		Elective Modules		Total No. of Modules
		No.	**(No./45) %**	**No.**	**(No./45) %**	
1	Programming Languages (PL)	**7**	15.55%	**1**	2.22%	8
2	Computational Science and Algorithms (CSA)	**5**	11.11%	**1**	2.22%	6
3	Main Computer Components (MCC)	**5**	11.11%	**0**	0%	5
4	Networking (NW)	**1**	2.22%	**1**	2.22%	2
5 + 6	Information Sciences and Applications (ISA)	**8**	17.77%	**3**	6.66%	11
7	Supplementary Courses (SC)	**4**	8.88%	**0**	0%	4
9	Graduation Project (GP) / Practical Training (PT)	**2**	4.44%	**0**	0%	2
	Total	**32**	**71.08%**	**6**	**13.32%**	**38**

- ***The Study Plan.*** The whole modules of the curriculum offered by the CS Department are shown in Appendix A of this Handbook.

- ***The Guidance Plan***. The Department guides students in their registration and selection of modules during the four years. The Department organizes a guidance plan that is shown in Table (2-2), where UR, FR, DR, and SR indicate University Requirements, Faculty Requirements, Department Requirements, and Supportive Requirements, respectively.

Philadelphia University

Guidance Plan for the Computer Science Department / Computer Science 2018 / 2019

Faculty of Information Technology

Year	Semester	Module Number	Module Title	Prerequisi	Type of Requirements
First	First (18 Credit Hours)	0114101	Arabic Language Skills (1)	0114099	(UR)
		0130101	English Language Skills (1)	0130099	(UR)
			University Elective (1)	------	(UR)
		0750113	Programming Fundamentals (1)	------	(FR)
		0250101	Differentiation and integration (1)	------	(SR)
		0731110	Introduction to Information Systems and Technology	------	(FR)
	Second (18 Credit Hours)	0111101	National Education	------	(UR)
		0780110	Introduction to Internet and Web Technology	------	(FR)
		0750114	Programming Fundamentals (2)	0750113	(FR)
		0750120	Discrete Mathematics	0750099	(DR)
		0721111	Software Engineering Fundamentals	0731110	(SR)
		0130102	English Language Skills (2)	0130101	(UR)
Second	First (18 Credit Hours)	0721240	Computing Ethics	0731110	(FR)
		3072122	Object-Oriented Programming	0750114	(FR)
		0731213	Introduction to Web Programming	0750114	(FR)
		0750230	Digital Logic Design	0731110	(DR)
		0750224	Theory of Computation	0250104	(DR)
		0750272	Numerical Analysis	0750114	(DR)
	Second (18 Credit Hours)	0721224	Data Structures	0721223	(SR)
		0731221	Database Fundamentals	0721223	(SR)
		0750233	Computer Organization and Design	0750230	(DR)
		0250241	Linear Algebra (1)	0250101	(SR)
		0750215	Visual Programming	0721223	(FR)
		0250231	Introduction to Statistics and Probabilities		(SR)
Third	First (18 Credit Hours)	0731321	Systems Analysis and Design	0721111	(SR)
		0750321	Concepts of Programming Languages	0721224	(DR)
		0750323	Algorithms	0721224	(DR)
		0750332	Computer Architecture	0750233	(DR)
		----	University Elective (2)	-----	(UR)
		0750350	Intelligent Systems	0250231	(DR)
	Second (15 Credit Hours)	0731340	Computer Networks Fundamentals	0721224	(SR)
		0750335	Operating Systems	0750332	(DR)
		0750399	Practical Training	90h	(DR)
		0750362	Database Applications Programming	0731221	(DR)
		0750324	Compiler Construction	0750324	(DR)
Fourth	First (13 Credit Hours)	0750472	Modeling and Simulation	075272	(DR)
		0750497	Research Project 1	-----	(DR)
		----	Department Elective (1)	-----	(DR)
		-----	University Elective (3)	-----	(UR)
		-----	Department Elective (2)		(DR)
	Second (14 Credit Hours)	-----	University Elective (4)	-----	(UR)
		0111100	Military Sciences (Or UE Non-Jordanians Students)	----	(UR)
		----	Department Elective (3)	----	(DR)
		0750446	Information Security	0731340	(DR)
		0750498	Research project 2	0750497	(DR)

(UR) University Req. **(UE) University Elective** **(FR) Faculty Req.**
(DR) Dept. Req. **(DE) Department Elective** **(SR) Supplementary Req.**

CHAPTER 3

FULL DESCRIPTION OF MODULES

This chapter presents the full description of the Department modules and those modules from the Faculty and University requirements that are computer-oriented modules.

3.1 Module Descriptor

The Department organised a format for the module descriptor that includes much information on the module. This sub-section presents the components of the adopted module descriptor that are shown in Figure (3-1). The University Quality Assurance Handbook explains in details the components of the module descriptor.

Figure (3-1) Components of the Module Description

Module Number, Module Title

- **Providing Department:**
- **Module Coordinator(s):**
- **Year:**
- **Credit:**
- **Prerequisites:** Required modules or background
- **Aims:**
- **Teaching Methods:**
- **Learning Outcomes:**
- **Assessment of Learning Outcomes:**
- **Contribution to Programme Learning Outcome:**
- **Syllabus:** Bulleted list providing an outline of the topics covered.
- **Modes of Assessment:**
- **Textbook and Supporting Materials:**
- **Instructor:**

3.2 Introductory Modules

Table (3-1) presents the Introductory (Level 1) modules whose full descriptions are given below.

Table (3-1) Introductory Modules in Computer Science Department

Module Number	**Module Title**
0250101	Differentiation and integration (1)
250241	Linear Algebra (1)
250231	Introduction to Statistics and Probabilities
0731110	Introduction to Information Systems and Technology
0750113	Programming Fundamentals (1)
0750114	Programming Fundamentals (2)
0750120	Discrete Mathematics
0721111	Software Engineering Fundamentals
0750230	Digital Logic Design
0780110	Introduction to Internet and Web Technology

0250101 Differentiation and Integration (1)

Level: 1

Prerequisite: **None**

Aims:

This course deals with the following main topics: differentiation of algebraic and transcendental functions, an introduction to analytic geometry, applications of differentiation, and a brief introduction to integration.

Teaching Methods: 38 hours Lectures (2-3 per week) + 10 hours Tutorial

Synopsis:

Functions: Representations of Functions. The Vertical Line Test. Symmetry. Linear Function. Polynomials. Piecewise Defined Functions. Rational Functions. Root Function. Trigonometric Functions. Combinations of Functions: Sum, Difference, Product, Quotient, Composition. Inverse Functions: Functions. Horizontal Line Test. Inverse Trigonometric Functions. Exponential and Logarithmic Functions. Hyperbolic Functions. Limits and Continuity: An Introduction to Limits. Calculating Limits using the Limit Laws. Limits at Infinity and Infinite Limits. Limits Involving ($\sin\theta/\theta$). Continuous Functions. The Derivative: The Derivative as a Function. Differentiation Rules and Higher Derivatives. The Chain Rule. Implicit Differentiation. Tangent Line. Applications of Differentiation: L'Hospital's Rule. Rolle's Theorem; Mean-Value Theorem. Analysis of Functions: Increase, Decrease, and Concavity. Relative Extrema; Graphing Polynomials. Absolute Maxima and Minima. Integration: Anti-derivatives. Indefinite Integrals. Integration by Substitution. The Definite Integral. The Fundamen.

Modes of Assessment:
Two 1-hour midterm exams (20% each); Assignments (20%); One 2-hours Final Examination (40%)

Textbook
Howard Anton, Irl C. Bivens and Stephen Davis, Calculus: Early Transcendentals, 10th Edition, John Wiley & Sons, Inc. 2013. − James Stewart, Calculus: Early Transcendentals, Brooks/ Cole.

250231, Introduction to Probability and Statistics

3 hours per week, 3 credit hours, prerequisite: **none**

Teaching Method: 30 hours Lectures (2 per week) + 15 hours Tutorials (1 per week)

Aims: This module aims to help students grasp basic statistical techniques and concepts, and to present real-life opportunities for applying them.

Textbooks:
1- D.C. Montgomery and .G.C. Runger, Applied Statistics and Probability For Engineers, 2nd Edition, Wiley, 2002
2- William, Probability and Statistics in Engineering and Management, Wiley, 2002

Synopsis: Descriptive statistics and probability distribution; Sampling distribution Estimation for the mean, variance and proportions; Testing for the mean, variance and proportions; Regression and correlation; One-way analysis of variance.

Assessment: Two 1-hour midterm exams (15% each); Assignments/Quizzes (10%); Tutorial Contribution (10%); 2-hours Final Exam (50%).
………………………………………………………………………………………………………. .

250241, Linear Algebra (1)

Course Hours: 4 hours per week , 3 credit hours

Prerequisite: 250101

Aims: It includes the study of System of Linear Equations, Gaussian Elimination, Methods to Find A-1, Matrices, Determinants, Euclidean Vector spaces, General Vector spaces, Subspaces, Linear Independence and Dependent Basis, Dimension, Row Space, Column Space, Null Space, Theory and Applications.

- To enable the students to carry on Matrix Operations.
- To enable students to solve Systems of Linear Equations using Matrices, and Gaussian Elimination.
- To understand the concepts of Vector Spaces.
- To understand Subspaces, and Basis.
- To carry on Row Space, Column Space, and Null Space.

Synopsis: Introduction to Systems of Linear Equations, Gaussian Elimination, Matrices and Matrix Operations, Inverses; Algebraic Properties of Matrices, Elementary Matrices and a Method for Finding A-1, More on Linear Systems and Invertible Matrices, Diagonal, Triangular, and Symmetric Matrices, Determinants by Cofactor Expansion, Evaluating Determinants by Row Reduction, Properties of the Determinants; Cramer's Rule, Vectors in 2-Space, 3-Space, and n-Space, Norm, Dot Product, and Distance in Rn, Orthogonality, Real Vector Spaces, Subspaces, Linear Independence, Coordinates and Basis, Row Space, Column Space, and Null Space, Rank, Nullity, and the Fundamental Matrix Spaces

Assessment: Two midterm exams (20% each); Laboratory (20%); Tutorial contribution (20%); Final exam (40%).

Textbooks:

- Linear algebra with applications by Leon, Steven J., 9th ed. Boston: Pearson Education Limited, 2015.
- Linear Algebra by L.W. Jhonson & R.D. Riess & J.T. Arnold- Addisson Wesely 2007.
- Linear Algebra by Eric Carlen_ Freeman 2007
- Linear Algebra and its applications by Gilbert Srang_Belmont, CA 2006
- Linear Algebra and its applications by David C. Lay_ pearson/addisson wesly2006.

0750120 Discrete Mathematics

Level: 1

Prerequisite: **None**

Aims:

This course is an introduction to Discrete Mathematics for students from the IT majors, covering main topics in number theory, propositional logic, proof techniques, sets and relations, counting techniques, and graph theory, together with selected applications in computer algorithms.

Teaching Methods: 38 hours Lectures (2-3 per week) + 10 hours Tutorial

Synopsis:

Logic: logic operators AND, OR, IFF, XOR, truth table, tautology, equivalence. Normal forms, predicates and quantifiers. Sets: set operations, set identities, power set, cardinality, cross product, power set. Modulo operation, divisibility, GCD and LCM, the Euclidean algorithm. Combinatorics: the addition and multiplication principles, the Pigeonhole principle. Inclusion-exclusion principles, permutations and combinations, permutations on multisets. Recurrence relation: solving first order homogeneous sequences. Methods of proof: mathematical induction. Relations: properties of relations, representation by digraphs, zero-one matrices, transitive closures. Equivalence relations, partial order relations, total order, Hasse diagrams. Graph Theory: complete graphs, complete bipartite, representations by adjacency matrix, incidence matrix, distance matrix. Trees, minimal spanning trees, Euler circuit, the Chinese postman problem. Coloring algorithms, planar graphs, maps and dual graphs.

Modes of Assessment:
Two 1-hour midterm exams (20% each); Assignments (20%); One 2-hours Final Examination (40%)

Textbook
Amin Witno, Discrete Structures in Five Chapters, CreateSpace 2010

750113 Programming Fundamentals (1)

Course Hours: 3 hours per week, 3 credit hours (total of 48 hours)

Level: 1

Prerequisite: **None**

Aims:
This module aims to introduce computer programming and emphasis in problem solving on the fundamentals of structured design using the principles of Top Down problem solving strategy (divide and conquer). This includes development, testing, implementation, documentation.
The module also aims to explore the logic of programming via the algorithm concepts and implement them in programming structures including functions, arrays, strings, and pointers.
.
Teaching Methods *Duration*: 16 weeks, 80 hours in total
Lectures: 32 hours (2 hours per week), *Tutorials*: 16 hours (1 per week),
Laboratories: 32 hours, 2 per week

Synopsis: problem solving strategies, algorithmic language to describe such problem solving, introduces the principles of procedural programming, data types, control structures, data structures and functions, data representation on the machine level. Various problems are considered to be solved using C-like procedural programming language.

Assessment: Two 1-hour midterm exams (15% each); lab (30%); one 2-hours Final Examination (40%)

Textbook:

- P. Deitel & H. Deitel, **C++ How to program,** Pearson Education Limited, 2013.
- Guttag, John. **Introduction to Computation and Programming Using Python**. Spring 2013 edition. MIT Press, 2013. ISBN: 9780262519632. - MIT

750114 Programming Fundamentals (2)

Course Hours: 3 hours per week, 3 credit hours (total of 48 hours)

Level: 1

Prerequisite: **750113**

Aims:
This module aims to introduce computer programming and emphasis in problem solving on the fundamentals of structured design using the principles of Top Down problem solving strategy (divide and conquer). This includes development, testing, implementation, documentation.
The module also aims to explore the logic of programming via the algorithm concepts and implement them in programming structures including functions, arrays, strings, and pointers
.
Teaching Methods *Duration*: 16 weeks, 80 hours in total
Lectures: 32 hours (2 hours per week), *Tutorials*: 16 hours (1 per week),
Laboratories: 32 hours, 2 per week

Synopsis: Functions definition, Parameters definition and passing , One dimensional array, Two dimensional array, use of main operations of a sequential file: open, reset, rewrite, read, write, eof, Introduction to Class and object, Generics, components reuse, component programming
Various problems are considered to be solved using C-like procedural programming language.
.

Assessment: Two 1-hour midterm exams (15% each); lab (30%); One 2-hours Final Examination (40%)

Textbook
D.S. Malik, Thomson, C++ Programming: From Problem Analysis to Program Design, Sixth Edition, Course Technology, 2011

……………………………………………………………………………………………………

0731110, Introduction to Information Systems and Technology

3 hours per week, 3 credit hours, prerequisite: None

Course (module) description:
This course provides an introduction to information systems and information technology, information systems development concepts, and application software. It identifies the basic types of business information systems, the major steps of the systems development process and some of the strategies employed to lower costs or improve service. It explains how information is used in organizations and how IT enables

improvement in quality, timeliness, and competitive advantage. It also defines the competitive advantages, types of roles, functions, and careers available in IS.

Course (module) objectives:
Students should be acquainted with handling and managing data and information in business organizations and to understand the meaning of “Information Systems and technology and their effects on organizations and the different types of business information systems and the development life cycle.
Students must learn about different Computer Hardware and Software and different types of computer networks. Students should know how to deal with e-commerce

Course/ module components:
1.Books (title , author (s), publisher, year of publication)
Information Systems Essentials, Editors: Stephen Haag, Maeve Cumming; Published: McGraw-Hill/Irwin, Inc, 2009, Third edition.

- **Support material (s):** slides
- **Study guide (s) (if applicable).**
- **Homework and laboratory guide (s) if (applicable).**

Learning outcomes:

- Knowledge and understanding

1. Know and understand a wide range of principles and fundamentals of Information Systems and Information Technology.
2. The application of IS and IT.

- Cognitive skills (thinking and analysis).

Basic analytical steps of Information Systems and defining the specifications of the IT required in business contexts**.**

- Communication skills (personal and academic).

1. Plan and undertake a small individual project in IS and IT fields.
2. Use the scientific literature effectively and make discriminating use of Web resources.
3. Present seminars in IS and IT fields.

- Practical and subject specific skills (Transferable Skills).

1. Use appropriate computer-based tools.
2. Work effectively with and for others.
3. Strike the balance between self-reliance and seeking help when necessary in new situations.
4. Get knowledge about self learning on the long run.

Assessment instruments:

- Short reports, presentations, Short research projects, Quizzes, and/or Home works (20%)
- First exam (20%)
- Second exam (20%)
- Final exam (40%)

……………………………………………………………………………………………..

750230, Digital Logic Design

Providing Department: Computer Science, Faculty of IT
Module Coordinator(s):
Year: 2

Credit: 3 credit hours
Prerequisite: 0731110

Aims: This module introduces you to the design and implementation of digital circuits. Topics include: combinational and sequential circuit analysis and design, digital circuit design optimization methods using random logic gates, multiplexers, decoders, registers, counters and programmable logic arrays. Laboratory experiments will be used to reinforce the theoretical concepts discussed in lectures. The lab experiments will involve the design and implementation of digital circuits. Emphasis is on the use computer aided tools in the design, simulation, and testing of digital circuits.

Teaching Methods: 41 hours Lectures (2-3 per week) + 4 hours Tutorials (1 per 3 weeks) + 3 hours Laboratory (1 per 4 weeks)

Learning Outcomes:
A student completing this module should be able to:
1. Define the problem (Inputs and Outputs), write its functions. (A, B, C)
2. Minimize functions using any type of minimizing algorithms (Boolean Algebra, Karnaugh map or Tabulation Method). (A, B)
3. Implement functions using digital circuit (Combinational or Sequential). (A, B)
4. Have knowledge in analyzing and designing procedures of Combinational and Sequential circuits. (B, C)
5. Have knowledge in designing and analyzing circuits with Flip-Flops, Counters and Registers. (B, C)
6. Work effectively with others. (D)
7. Use simulation software, for testing the designed circuit. (C, D)

Assessment of Learning Outcomes
Learning outcomes (1), (2), and (3) are assessed by examinations, tutorial and in the laboratory. Learning outcomes (4), (5), and (6) is assessed by course work/workshops. Learning outcomes (7) is assessed in the laboratory.

Contribution to Programme Learning Outcomes:
A1, A3, A5, B1, B3, C6, D3, D6

Synopsis: Introduction to Digital logic Design; Binary Systems and Codes: Binary Numbers, Octal and Hexadecimal Numbers, Number Base Conversions, Arithmetic Operation with different Bases, Complements, Signed Binary Numbers, Binary Codes: BCD, Gray, ASCII and EBCDIC; Binary Logic and Logic Gates: AND, OR and NOT; Boolean Algebra and Logic Gates: Basic Definition, Basic Theorems, Boolean Functions; Standard Forms: Minterm and Maxterm, Simplification of Boolean Functions using SOP and POS; Logic Operations: NAND, NOR, Exclusive-OR and Equivalence, Integrated Circuits; Gate-Level Minimization: The Map Method, Two- and Three-Variable Map, Four-Variable Map, Product of Sums Simplification, Don't-Care Conditions, NAND and NOR Implementation, The Tabulation Method, Simplification of Boolean Functions using Tabulation Method; Analysis and Synthesis of Combinational Circuits: Combinational Circuits, Analysis and Design Procedure, Binary Adders-Subtractor, Decoders and Multiplexers; Analysis and Synthesis of Sequential Circuits: Sequential Circuits, Latches, Flip-Flops: RS, D, JK and T, Analysis of Clocked Sequential Circuits, Design Procedure; Registers and Counters: Registers, Shift Registers, Synchronous Counters, Ripple Counters; Sequential Circuits with Programmable Logic Devices: Introduction, Random-Access Memory, Memory Decoding, Read-Only Memory, Programmable Logic Array.

Modes of Assessment:
Two 1-hour midterm exams (20% each); Assignments (20%); one 2-hours Final Examination (40%)

Textbook and supporting material:
1- Morris Mano, Digital Logic, Prentice-Hall, 2012

2- Morris Mano, Charles R. Kime, Logic and computer design fundamentals, Pearson Prentice Hall, 2004
3- Basavaraj,B., Digital fundamentals, New Delhi: Vikas Publishing House, 1999.
4- Kandel Langholz, Digital Logic Design, Prentice Hall, 1988.
5- Rafiquzzaman & Chandra, Modern Computer Architecture, West Pub. Comp., 1988.

……………………………………………………………………………………………………….

721111, Software Engineering Fundamentals

Providing Department: Software Engineering, Faculty of IT
Module Coordinator:
Year: 2
Credit: 3 credit hours
Prerequisite: 750113+731110

Aims:
This module aims to provide students a comprehensive introduction to software engineering. It gives an introduction to basic concepts, principles and techniques used in software engineering. This module gives an introduction to methods for analysis, design, testing, and implementation of medium size software systems. Simple and realistic case studies will be used along all the software process steps.

Teaching Methods: 38 hours Lectures (2-3 per week) + 10 hours Tutorial

Learning Outcomes:
A student completing this module should be able to:
1- Understand a wide range of principles and tools available to the software engineer such as specification, design, coding and testing methodologies, and user interface techniques. (A)
2- Design software systems of small size through academic and realistic case studies (tutorials). (B, C, D)

Contribution to Programme Learning Outcomes:
A2, B2, C5, D4
Synopsis: Basic Concepts: Software product, Software crisis, software engineering, software process, software process model, methodologies, methods, tools, artefacts; **Software Process (I):** process models, iterative process; **Software Process (II):** software process activities (specification, design and implementation, validation/verification, evolution); **Software Requirement Engineering (I):** Functional/Non Functional requirements, user requirements, system requirement, requirement document; **Software Requirement Engineering (II**): Software requirement, elicitation and analysis, basics on Use cases, UML notation; **Software Prototyping; System Models (I):**Context models, Behavioural models**; System Models (II):** Data Models, Objects Models**; Architectural Design:** system structuring, control models, modular decomposition; **Object Oriented Design, UML notation; User interface design:** user interface design principles, user interaction, information presentation**; Verification and Validation:** planning, software inspections, automated static analysis; **Software Testing :** defect testing, integration testing; **Software Change:** program evolution dynamics, software maintenance; **Software Cost estimation**

Modes of Assessment:
Two 1-hour midterm exams (20% each); Assignments (20%); One 2-hours Final Examination (40%)

Textbook and supporting material:
1- Ian Sommerville, Software Engineering 7/e, Addison Wesley, 2004
2- R. S. Pressman Software Engineering: A Practitioner's Approach, 5th Edition, McGraw Hill; 2001
Website(s): www.software-engin.com

3.3 Intermediate Modules

The Intermediate (Level 2) modules are listed in Table (3-2) and their full descriptions are given below.

Table (3-2) Intermediate Modules in Computer Science Department

Module Number	Module Title
0721240	Computing Ethics
0721224	Object-Oriented Programming
0250231	Introduction to Statistics and Probabilities
0750224	Theory of Computation
0750272	Numerical Analysis
0721224	Data Structures
0731221	Database Fundamentals
0750233	Computer Organization and Design
0750215	Visual Programming
0731213	Introduction to web programming

721240, Computing Ethics

Providing Department: Software Engineering, Faculty of IT
Module Coordinator(s)**:**
Year: 2
Credit: 3 credit hours
Prerequisite: 731110

Aims:

This module aims to give students an informed awareness of the principal issues of professional ethics and responsibility (ergonomics and ethics) in the analysis, design, implementation and use of computers, information systems and Information Technology (IT) products. This will help students in recognition of ethical problems when they occur. Also it will enable students to deal effectively with ethical, social and professional issues now and in their future careers.

Teaching Methods: 36 hours Lectures (2-3 per week) + 9 hours Projects (class work) (average 1 per week) + 3 hours Seminars (1 per month)

Learning Outcomes:
On completing this module, students will:
1. Understand the basic concepts of ethics, moral, law, ergonomics and profession.
2. Be aware of the requirements for accreditation in respect of Professional Issues.
3. Have a basic knowledge of Intellectual Property Rights (IPR) in relation to Copyright and Patents.
4. Be aware of some of the potential problems of managing large IT projects in accordance with professional and ethical issues.
5. Be aware of the requirements for professionalism in respect of the work of the professional societies and their codes of conduct and practice.
6. Have acquired basic knowledge of the Data Protection Act and its implications.
7. Be able to assess and evaluate the legal aspect of workplace practices.
8. Be able to asses and evaluate the impacts of IT technology on society and culture.
9. Be aware of Jordanian Professional Issues.

Assessment of Learning Outcomes:
Learning outcomes are assessed through examination and individual and group case studies, which require demonstration of the use of a combination of the learning outcomes to be employed in producing the essays and presentations.

Contribution to Programme Learning Outcomes:
A2, A5, B1, C1, C3, C4, C6, D3, D4.

Synopsis*:* Introduction to Ethics; Professional and Professionalism; Code of Ethics and Social Issues; Computer/IT professionals; Computer Security; Privacy and Internet Issues; Information Systems and Ethics; Associations of IT professionals; Ethics and the Internet; Ethical Challenges of e-Business; Ethical Challenges of e-Business; Continuous Professional Development; Intellectual Property Rights; Jordanian Codes for Intellectual Property Rights; Seminars and Project Discussion.

Modes of Assessment:
Two 1-hour midterm exams (20% each); Assignments (20%); One 2-hours Final Examination (40%)

Textbooks and Supporting Material:
1. Deborah G. Johnson, Computer Ethics. 3ed Edition, Englewood Cliffs, N.J., Prentice Hall, 2001.
2. Gorge Reynoids, Ethics in Information Technology, Thomason, 2003.
3. Sara Baase, A Gift of Fire: Social, Legal and Ethical Issues for Computer and the Internet, 2nd ed., 2003.
4. Tavani H. T. and Hoboken N. J., Ethics and Technology, John Wiley, 3rd ed, 2004.
5. مجموعة تشريعات الملكية الفكرية الأردنية

Website(s):
ACM, IEEE and BCS Web Sites.
www.cyberethics.cbi.msstste.edu
www.aitp.org
www.acm.org
www.prenhall.com

……………………………………………………………………………………………………..

721223, Object-Oriented Programming

Providing Department: Software Engineering, Faculty of IT
Module Coordinator(s)**:**
Year: 1
Credit: 3 credit hours
Prerequisite: 750114

Aims:
This module introduces the concepts of object-oriented programming for students after having a background in the procedural paradigm. It aims to develop an understanding of the principles of the object-oriented paradigm, provide familiarity with approaches to object-oriented modelling and design, provide a familiarity with the syntax, class hierarchy, environment and simple application construction for an object-oriented programming language. The module emphasizes modern software engineering principles and developing fundamental programming skills in the context of a language that supports the object-oriented paradigm (Java for instance).

Teaching Methods: 32 hours Lectures (2 per week) + 16 hours Tutorial (1 per week) + 16 hours Laboratory (1 per week)

Learning Outcomes:
A student completing this module should:
1. Acquire a full Object Oriented Thinking (A)
2. Have a clear understanding of the object-oriented concepts such as objects, classes, inheritance, and polymorphism. (A)
3. Have an informal understanding of the operational semantics of object-oriented programs in terms of creation of objects and messages passing between them. (A)
4. Be able to design small object oriented programs which meet requirements expressed in English, with a strong software engineering foundation (B)
5. Have knowledge of Object Oriented Design guidelines. (A, B)
6. Be able to code small software systems in Java language. (C).
7. Be able to maintain large, high-quality software systems (C)

Assessment of Learning Outcomes:
Learning outcomes (1), (6), and (7) are assessed by examination and laboratory. Learning outcomes (2), (3), and (7) are assessed by tutorial and examination. Learning outcomes (4) and (5) are assessed in the laboratory.

Contribution to Programme Learning Outcomes:
A2, A3, B1, B2, C5, D2, D4

Synopsis: Introduction to Object Oriented Thinking: Object Modeling; Objects and Classes; Understanding Class Definition; Object Interaction (1): Overloading; Object Interaction (2): Composition; Grouping Objects; Using Library Classes; More Sophisticated Behavior: Information Hiding; Inheritance (1): Reuse, Inheritance (2): Sub-typing; Inheritance (3): Polymorphism, Overriding; Abstract Classes, Abstract Methods, Interfaces, Multiple inheritance; Exception Handling; Designing Applications

Modes of Assessment:
Two 1-hour midterm exams (20% each); Assignments (20%); One 2-hours Final Examination (40%)

Textbooks and Supporting Material:

1- David j. Barnes And Michael Kolling, Objects First with Java: A Practical Introduction using BlueJ, Prentice Hall, Pearson Education, 2nd Edition, 2005

……………………………………………………………………………………………………..

750224, Theory of Computation

Providing Department: Computer Science, Faculty of IT
Module Coordinator(s):
Year: 2
Credit: 3 credit hours
Prerequisite: 250104

Aims:
This module introduces the theory of computation through a set of abstract machines that serve as models for computation - finite automata, pushdown automata, and Turing machines - and examines the relationship between these automata and formal languages. Additional topics beyond the automata classes themselves include deterministic and nondeterministic machines, regular expressions, context-free grammars, undecidability, and the P = NP question.
Finite automata are a useful model for many important kinds of hardware and software. Here are the most important kinds: Software for designing and checking the behaviour of digital circuits; The "lexical analyzer" of a typical complier, that is, the compiler component that breaks the input text into logical units, such as identifiers, keywords, and punctuation; Software for scanning large bodies of text, such as collections of Web pages, to find occurrences of words, phrases, or other patterns; Software for verifying systems of all types that have a finite number of distinct states, such as communication protocols or protocols for secure exchange of information.

Teaching Methods: 38 hours Lectures (2-3 hours per week) + 10 hours Tutorials (average 1 per week)

Learning Outcomes:
A student completing this module should be able to:
1. Acquire a full understanding and mentality of Automata Theory as the basis of all computer science languages design (A)
2. Have a clear understanding of the Automata theory concepts such as RE's, DFA's, NFA's, Stack's, Turing machines, and Grammars (A, B).
3. Design FAs, NFAs, Grammars, languages modelling, small compilers basics (B).
4. Minimize FA's and Grammars of Context Free Languages (C).
5. Design sample automata (B)

Assessment of Learning Outcomes
Learning outcome (1) and (2) are assessed by tutorials and examinations. Learning outcomes (4) is assessed by tutorials, homework, and examinations. Learning outcomes (3) and (5) are assessed by tutorials.

Contribution to Programme Learning Outcomes:
A1, A2, B1, B2, C2, C5

Synopsis: Basic concepts and definitions; Set operations; partition of a set; Equivalence relations; Properties on relation on set; Proving Equivalences about Sets; Central concepts of Automata Theory; Regular Expressions; Operations on Regular expressions; Finite Automata and Regular Expressions; Recursive definitions; Conversion from FA and regular expressions; Kleen's Theory; Mealy Moore Machines; Conversion from Mealy to Moore and vice versa; Deterministic Finite Automata (DFA); Equivalence Classes; Minimization of DFA; Non-Deterministic Finite Automata (NDFA); Equivalence of Deterministic and Non- Deterministic Finite Automata; Finite Automata with Epsilon-Transition; Equivalence between DFA, NFA, NFA-Λ; Pumping Lemma for Regular Languages; Closure Properties of Regular Languages; Context Free languages; Context-Free Grammars; Regular Grammars; Parse Trees; Ambiguity in Grammars and Languages; Simplified Forms; Standard Forms; Chomsky Normal Forms; Greibach normal Forms; Pumping Lemma for Context-Free Languages; Closure Properties of Context-Free Languages; Minimization of CFGs; Pushdown Automata (PDA); Deterministic and Non-Deterministic (PDA); Formal definition of NPDA; Transition functions of NPDA; NPDA Execution; Accepting Strings with NPDA; Equivalence of PDAs and CFG; The Turing Machine; Programming Techniques for Turing Machines; Formal definition of TMs; TMs as acceptors; TMs as transducers; Recognizing Languages with TMs; Sorting with TMs; Programming in TMs; Multiple Tracks, Subroutines; Complexity issues and analysis.

Modes of Assessment:
Two 1-hour midterm exams (20% each); Assignments (20%); One 2-hours Final Examination (40%)

Textbooks and Supporting Material:

1. Michael Sipser, Introduction to the Theory of Computation, 3rd edition, Published by Cengage Learning, 2013.
2. Daniel I. A. Cohen, "Introduction to computer theory", Second Edition, Prentice-Hall, 1997.
3. Papadimitriou, Elements of the Theory of Computation, Prentice-Hall, 1998
4. John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman, "Introduction to Automata Theory, Languages, and Computation", Second Edition, Prentice-Hall, 2001
5. Peter Dehning, Jack B. Dennis, “Machines, Languages and Computation”, Second Edition, Prentice-Hall, 1978
6. Harry R. Lewis, Christos H. Papadimitriou, "Elements of the theory of computation", Second Edition, Prentice-Hall, 1998

Simulators:
In order to improve the pedagogy of this course, interactive animations of the various automata using available simulators are recommended.
……………………………………………………………………………………………………. .

750272, Numerical analysis

Course Hours: 3 hours per week, 3 credit hours (total of 48 hours)

Level: 1

Prerequisite: **250101 + 750114**

Aims:
The aim of the module is to give students a clear understanding and deep knowledge how the typical of "real life" mathematical, physical, or engineering problems are to be solved in the modern setting. As opposed to tendency in lower-level mathematical courses to teach recipes for "exact" solving particular problems fitting into very special form, this module provides the idea of approximate solving wide variety of applied standard problems on a computer by numerical methods.

Teaching Methods: Lectures: 36 hours, Tutorials: 12 hours

Synopsis: Mathematical Preliminaries: Computer arithmetic, round-off error, source of errors, Solution of equations in one variable: Bisection method, fixed point method, false position method, Secant method, Newton-Raphson method, Interpolation and polynomial approximation, Introduction to interpolation, Direct methods for solving linear systems of equations, Iterative methods for solving linear systems, Iterative methods for solving nonlinear systems,Curve fitting techniques.

Assessment: Two 1-hour midterm exams (20% each); Assignments (20%); One 2-hours Final Examination (40%)

Text book:

Richard L. Johnson and Douglas J. Faires, Numerical Analysis, 9th Edition, Brooks/
Cole 2010.

………………………………………………………………………………………………………. .

721224, Data Structures

Providing Department: Software Engineering, Faculty of IT
Module Coordinator(s)**:**
Year: **2**
Credit: 3 credit hours
Prerequisite: 721223+250104

Aims:
This is a **programming-intensive** module where students learn the fundamentals of designing data structures for use in complex programs. Data structures course is an essential area of study for computer scientists and for anyone who will ever undertake any serious programming task. This course deals with the fundamentals of organizing and manipulating data efficiently using clean conceptual models. Students study many of the important conceptual data types, their realization through implementation, and analysis of their efficiency. Implementations in this course are carried out in the Java programming language, but the principles are more generally applicable to most modern programming environments.
Topics include recursion, the underlying philosophy of object-oriented programming, fundamental data structures (including stacks, queues, linked lists, hash tables, trees, and graphs), and the basics of algorithmic analysis.

Teaching Methods: 32 hours Lectures (2 per week) + 16 hours Tutorial (1 per week)

Learning Outcomes:
On successful completion of this module, student will:

1. build on understanding of basic ideas about data structures given in the prerequisite module (A)
2. understand the basic concepts of time and space complexity (A)
3. be able to manipulate recursive algorithms (B)
4. be able to develop efficient algorithms for manipulating data structures (B)
5. know a range of algorithm structures and how to implement them (A, B, C)
6. know and understand a wide range of searching and sorting algorithms (A, B)
7. understand how the Abstract Data Type (ADT) is used (A)
8. understand several representations of trees, and their applications (A, C)
9. understand several representations of graphs, and their applications, together with a selection of important algorithms on graphs (A, C)
10. be able to construct and use the data structures mentioned above. (A, B, C)

Assessment of Learning Outcomes:
Outcomes 1 to 10 are assessed by coursework and examinations
Contribution to Programme Learning Outcomes:
A1, B1, B2, C5, D6

Synopsis: Introduction to Software Engineering, Introduction to data structures: data structures and algorithms; Data Design and Implementation; Algorithm complexity; List ADT: static implementation, single linked list; List ADT: dynamic implementation, single linked list; Lists: doubly linked list and circular linked list; Stacks: Static implementation and dynamic implementation; Queues: Static implementation and dynamic implementation, circular queue; Programming with Recursion; Trees: Binary search tree; Trees : binary expression tree, and heap tree; Priority Queues and Heaps; Graph ADT; Sorting: Bubble sort, selection sort, insertion sort, Quick sort, Heap sort; Searching: Sequential search, Binary Search; Hashing: hash function, Separate chaining, open addressing

Modes of Assessment:
Two 1-hour midterm exams (15% each); Coursework (15%); Tutorial Contribution (5%); Final (unseen) Exam (50%)

Textbooks and Supporting Material:
1- Nell Dale, Daniel T. Joyce and Chip Weems, Object-Oriented Data Structures using Java, Jones and Bartlett Publishers, 2001
2- Goodrich and Tamassia, Data Structures and Algorithms in Java, 2nd edition, John Wiley and Sons, 2000, ISBN 0471383678.
3- Arnold, Gosling, and Holmes, The Java Programming Language, 3rd edition, Addison-Wesley, 2000, ISBN 0201704331.
4- Mark Allen Weiss, Data Structures and Algorithm Analysis in C++, Addison-Wesley, 1999

……………………………………………………………………………………………….

731221, Database Fundamentals

Providing Department: Computer Information Systems, Faculty of IT
Module Coordinator(s):
Year: 2
Credit: 3 credit hours
Prerequisite: 721223

Aims:
This module aims to give the students the main concepts of database, design the database, database models, normalization techniques, query languages, object oriented database, query optimization and database and the web. Further the students have to practice and write some applications regarding the database.

***Teaching Methods*:** 26 hours Lectures (average 2 per week) + 16 hours Laboratory (1 per week) + 6 hours Tutorials (1 each fortnight)

Learning Outcomes:
When completing this module, a student should be able to:
1. Discuss/explain the importance of data, and the difference between file management and databases. (A)
2. Discuss/explain the design of database management system architectures and environments. (A)
3. Discuss/explain the principals of database design. (A)
4. Discuss, explain and apply conceptual design methodologies, in particular conceptual design using Extended Entity Relationship modelling. (A, B, C, D)
5. Discuss, explain and apply the relational model and mappings from conceptual designs, in particular normalizations. (A, B, C, D)
6. Discuss/explain physical and performance related design considerations. (A)
7. Discuss/explain transaction processing. (A)
8. Discuss, explain and apply SQL and the Oracle DBMS. (A, C, D)

Assessment of Learning Outcomes:

Learning outcomes (1) through (7) are assessed by examinations. Learning outcomes (3), (4), and (8) are assessed by projects design and implementation.

Contribution to Programme Learning Outcomes:
A2, A3, A4, A5, B1, B2, B3, C1, C2, C6, D1, D3

Synopsis: Introduction to Data base and DBMS; Database Models; Database Design; Relational Algebra and Relational Calculus; Query Languages (SQL); DB normalization; Database Integrity and Security; Indexing Techniques; Query Optimization; Distributed Data Base; Object-Oriented Database

Modes of Assessment:
Two 1-hour midterm exams (20% each); Assignments (20%); One 2-hours Final Examination (40%)

Textbooks and Supporting Material:
1- A. Silberschatz, H.F. Korth & S. Sudarshan, Database System Concepts, McGraw Hill, 2002
2- El-Masri & Navathe, Fundamentals of Database Systems, Prentice Hall, 2002.
3- Jeffry Ullman, Principles of Database Systems, SU Publishers, 1999
4- C.J. Date, An Introduction to Database Systems, Addison Wesley, 1995

……….

750233 Computer Organization and design

Course Hours: 3 hours per week, 3 credit hours (total of 48 hours)

Level: 3

Prerequisite: **750230**

Aims:
The aim of this course is to introduce organization of the computer components. The module emphasizes on the following knowledge areas: Digital components used in the organization and design of digital computer, serial and parallel transfer, Flow of information and timing signals, design an elementary basic computer, organization and architecture of the central processing unit.

Teaching Methods *Duration*: 16 weeks, 80 hours in total
Lectures: 32 hours (2 hours per week), *Tutorials*: 16 hours (1 per week),
Laboratories: 32 hours, 2 per week

Synopsis: Data Representation, Register Transfer and Micro-operations, Basic Computer Organization and Design, Instruction Types, Central Processing Unit, Design of Basic Computer.

Assessment: Two 1-hour midterm exams (20% each); Assignments (20%); One 2-hours Final Examination (40%).

Text Book:

The Essentials of Computer Organization and Architecture, Linda Null, Julia Lobur, Jones and Bartlett Publishers, 2012

………

750215, Visual Programming

Providing Department: Computer Information Systems, Faculty of IT
Module Coordinator(s):

Year: 2
Credit: 3 credit hours
Prerequisite: **721223**

Aims: This module aims to provide students capabilities to design and implement the applications using visual programming through Microsoft Visual Studio .Net and VC# to develop different types of applications using .Net platform.

Teaching Methods: 32 hours Lectures (2 per week) + 12 hours Tutorials (on average 1 per week) + 16 hours Laboratory (1 per week) + 4 hours Seminar

Learning Outcomes:
On completing this module you should:

2. Have a clear understanding of what comprises a correct program in C# through .Net frame components (A)
3. Have a clear understanding of the object-oriented terminology used to describe features of C# and VC# project with their visual components. (A, C)
4. Have an informal understanding of the operational semantics of object-oriented programs in terms of creation of objects and messages passing between difference interfaces. (A)
5. Be able to design, code, and test C# project, which meet requirements expressed in English. (B, C, D)
6. Be able to understand the documentation for, and make use of, the MSDN library. (A, C)
7. Have a good understanding of the different focus at various stages of the development process. (A, C, D)
8. Have knowledge of design GUI with visual components guidelines. (A, B)
9. Be able to apply the guidelines in learning outcome (7) to a real design problem and justify how they have been used. (A, B)
10. Be able to write a project in C# and VC#, which implements the design in learning outcome (8). (C).
11. Be able to work effectively alone or as a member of a small group working on some programming tasks. (C, D)

Assessment of Learning Outcomes:
Learning outcomes (1), (6), and (8) are assessed by examination and laboratory; learning outcomes (2), (3), and (7) are assessed by tutorial and examination; learning outcomes (4), (5), (9) and (10) are assessed in the laboratory.

Contribution to Programme Learning Outcomes:
A2, A3, A4, B3, C5, C6, D1, D2, D4, D5

Synopsis: **Introducing the Microsoft .NET Platform:** .NET Platform, .NET and Windows DNA, .NET Architecture Hierarchy, .NET Platform features, Multilanguage Development, Platform and Processor Independence, .NET Components, Common Type System CTS, Common Language Specification CLS , .NET Base Class Library (BCL); **Visual Studio.NET IDE:** Visual Studio.NET, Components of VS.NET, Design Window, Code Window, Server Explorer, Toolbox, Docking Windows, Properties Explorer, Solution Explorer, Object Browser, Dynamic Help, Task List Explorer, Features of VS.NET, XML Editor, Creating a Project, Add Reference, Build the Project, Debugging a Project; **Introducing C# Programming:** Data Types, Value Types, Reference Types, Control Structures (if, if-else, switch, for, while, do while, break, continue, return, goto), Understanding Properties and Indexers Accessing Lists (Array) with Indexers, Events, Exception Handling, Using OOP (Object, Class, Constructor/destructor, Inheritance, Polymorphism, Encapsulation); **Windows Forms:** Windows Forms, Adding Controls, Adding an Event Handler, Adding Controls at Runtime, Attaching an Event Handler at Runtime, Writing a Simple Text Editor, Creating a Menu, Adding a New Form, Creating a Multiple Document Interface, Creating a Dialog Form, Using Form Inheritance, Adding a *TabControl,*, Anchoring Controls, Changing the Startup Form, Connecting the Dialog, Using the ListView and TreeView, Controls, Building an *ImageList,* Adding a *ListView,* Using the Details View, Attaching a Context Menu, Adding a *TreeView,* Implementing Drag and Drop, Creating Controls, Creating a User Control, Adding a Property, Adding Functionality, Writing a Custom Control, Testing the Control, Enhancing the Control, Sub classing Controls; **Graphics and Multimedia:** Graphics Contexts and

Graphics Objects, Color Control, Font Control, Drawing Lines, Rectangles and Ovals, Drawing Arcs, Drawing Polygons and Polylines, Advanced Graphics Capabilities, Introduction to Multimedia, Loading Displaying and Scaling Images, Animating a Series of Images, Windows Media Player, Microsoft Agent; **ADO.NET:** ADO.NET Architecture, Understanding the *Connection*Object, Building the *Connection* String, Understanding the *Command*Object, Understanding *DataReaders*, Understanding *DataSets* and *DataAdapters,* DataTable, *DataColumn, DataRow,* Differences between *DataReader* Model and *DataSet* Model, Understanding the *DataView*Object ,Working with System.Data.OleDb, Using *DataReaders,* Using *DataSets,* Working with SQL.NET, Using Stored Procedures, Working with Odbc.NET, Using DSN Connection; **Multithreading:** Thread States: Life Cycle of a Thread, Thread Priorities and Thread Scheduling, Thread Synchronization and Class Monitor, Producer/Consumer Relationship without Thread, Synchronization, Producer/Consumer Relationship with Thread Synchronization, Producer/Consumer Relationship: Circular Buffer; **Networking:** Introduction, Establishing a Simple Server (Using Stream Sockets), Establishing a Simple Client (Using Stream Sockets), Client/Server Interaction with Stream-Socket Connections, Connectionless Client/Server Interaction with Datagrams, one Server multi-Clients system; **ASP.NET:** Introducing the ASP.NET Architecture, ASP.NET Server Controls, Working with User, Controls, Custom Controls, Understanding the Web.config File, Using the Global.asax Page,

Modes of Assessment: Two 1-hour midterm exams (20% each); Assignments (20%); One 2-hours Final Examination (40%)
)

Textbooks and reference books:

1- H. M. Deitel & J. Deitel, " C# How to Program", Prentice Hall, 2014 fifth edition
2- A.Turtschi et.al. " Mastering Visual C# .Net", Sybex 2002
3- Eric Gunnerson, "A Programmer's Introduction to C#", Apress 2000
4- Anders Hejlsberg et.al. " C# Language Reference", Microsoft Corporation 2000
5- Erric Buttow et al. "C#, your visual blueprint for building .Net application", Hungry Minds 2002
6- Charles Carroll, "Programming C#", O'Reily & Associates 2000
Karh Watson "Beginning C#" Wrox Press 2001.

……………………………………………………………………………………………………

0731213 Introduction to Web Programming

3 hours per week, 3 credit hours, prerequisite: 0750114

Course description

This course is intended to give the student advanced issues in website design and implementation. At the course completion, students will have the know-how of designing and implementing web-based applications, completely database-driven web sites.

The course involves two main parts:

- Advanced client-side programming.
- Advanced server-side programming.

Course objective

On successfully completing the module, the students are expected to have gained good knowledge of:

- Implementing advanced server-side programming
- Improving personal productivity concepts through web authoring.

Course components:

- Books (title, author (s), publisher, year of publication)
Gosselin, don, PHP Programming with MySQL. Course Technology Incorporated, 2005, ISBN 0-619-21687-5
- Support material (s)
The world's largest web development site: www.w3schools.com
- Study guide (s) (if applicable).
- Homework and laboratory guide (s) if (applicable).

Teaching methods

Lectures, discussion, groups, tutorials, problem solving, etc.

Learning outcomes:

- Knowledge and understanding.
- Cognitive skills (thinking and analysis).
- Communication skills (personal and academic).
- Practical and subject specific skills (Transferable Skills).

Assessment instruments:

- Short reports, Assignments, Projects, Quizzes, and/or Home works (20%)
- First exam (20%)
- Second exam (20%)
- Final exam (40%)

3.4 Advanced Modules

In this sub-section, the full descriptions of Level 3 modules are presented. Table (3-3) shows these modules and their descriptions are given below.

Table (3-3) Advanced Modules in Computers and Information Systems Department

Module Number	Module Title
0731321	Systems Analysis and Design
0750321	Concepts of programming languages
0750323	Algorithms
0750332	Computer Architecture

0750350	Intelligent Systems
0731340	Fundamentals of Computer Networks
0750335	Operating Systems
0750399	Practical Training
0750362	Database applications programming
0750324	Compiler construction
0750472	Modelling and Simulation
0750497	Research Project 1
0750445	Wireless and Mobile Computing
0750413	Concurrent and Distributed Programming
0750446	Information Security
0750498	Research project 2

731321, Systems Analysis and Design

Providing Department: Management Information Systems, Faculty of IT
Module Coordinator(s):
Year: 3
Credit: 3 credit hours
Prerequisite: 721111
Aims:
This module introduces the students to the concepts and skill of system analysis and design. It includes expanded coverage of data flow diagrams, data dictionary, and process specifications, as it introduces examples of new software used by analysts, designers to manage projects, analyze and document systems, design new systems and implement their plans. It introduces also a recent coverage of UML, wireless technologies and ERP; web based systems for e-commerce and expanded coverage on RAD and GUI design.

Teaching Methods: 32 hours Lectures (2 per week) + 8 hours Tutorials (1 per fortnight) + 8 hours Seminars (in the last 3 weeks)

Learning Outcomes:
At the end of this module, student will be able to:
1- Understand the principles and tools of systems analysis and design (A).
2- Solve a wide range of problems related to the analysis, design and construction of information systems (A, B, C).
3- Understand the application of computing in different context (A).
4- Understand the professional and ethical responsibilities of practicing the computer professional including understanding the need for quality. (A)
5- Plan and undertake a major individual project, prepare and deliver coherent and structured verbal and written technical reports (B, C, D).
6- Analysis and Design of systems of small sizes. (B, C)

Assessment of Learning Outcomes
Learning outcomes (1) – (4) are assessed by examinations, tutorial and in the assignments. Learning outcomes (5) and (6) are assessed by seminars and projects

Contribution to Programme Learning Outcomes:
A2, A3, A4, A5, B1, B3, C1, C2, C5, D2, D4, D5.

Synopsis: Systems Analysis Fundamentals: Introducing Systems Analysis and Design Concepts, roles of systems analysts, system development life cycle, using CASE Tools, depicting system graphically, determine feasibility, activity planning and control; Information requirements analysis: Sampling and investigating data, interviewing, using questionnaires, prototyping; The analysis process: Using data flow diagram, using data dictionaries, describing process specifications and structured decisions, the system proposal; The essential of design: Designing output, designing the files or database, designing the user interface, designing data entry forms, documenting the design phase; Software engineering and implementation: Quality assurance through software engineering, implementing the information system, Object oriented analysis and design; Different software tools will be used in this course.

Modes of Assessment:
Two 1-hour midterm exams (15% each); Assignments (15%); Tutorial Contribution (5%); Final Examination: written (unseen) exam (40%) + final project (10%)

Textbook and Supporting Material:
1. Kenneth E. Kendall and Julie E. Kendall, Systems Analysis and Design 5th Edition, Prentice Hall PTR, 2001
2- Silver and Silver, System Analysis and Design, Addison Wesley, Last Edition
……………………………………………………………………………………………………

750321, Concepts of Programming Languages

Providing Department: Computer Science, Faculty of IT
Module Coordinator(s):
Year: 3
Credit: 3 credit hours
Prerequisite: 721224

Aims:
This module aims to provide the student with a framework for thinking about programming languages. There are always new languages being devised, of which a very few actually become widely used; typically, these are specialized languages for particular applications (e.g., Java). As a computer scientist, the student must be able to learn new languages as necessary, and the background he/she gets from this

module should make this easier. Finally, students will almost certainly have to choose which programming language to use for a particular project. A final goal of this module is to give students enough background in the study of programming languages that they can argue persuasively why a particular language is appropriate (or inappropriate) for a particular problem.

Teaching Methods: 35 hours Lectures (2 per week) + 13 hours Tutorials (average 1 per week)

Learning Outcomes:
A student completing this module should be able to:
1. understand different programming paradigms. (A, D)
2. understand the syntax and semantic of programming languages. (A, B)
3. develop different projects using different programming languages. (B, C, D)
4. design a new programming language. (B)

Assessment of Learning Outcome:
Learning outcome (1) and (2) are assessed by homework, assignments, and examinations. Learning outcome (3) is assessed by project assignment and examinations. Learning outcome (4) could be assessed by project.

Contribution to Programme Learning Outcomes:
A1, B1, B3, C5, D4, D6

Synopsis: Introduction; A survey of Programming Paradigms; Imperative Programming: Names, Bindings, and Type Checking; Scopes; Data Types: Primitive Data Types, Character String Type, User-Defined Ordinal Types; Data Types : Array Types, Record Types, Union Types, Set Types, and Pointer Types; Statement-Level Control; Subprograms; Abstract Data Types; Support for Object-Oriented Programming; Functional Programming; Logic Programming; Scripting Languages

Modes of Assessment:
Two 1-hour midterm exams (15% each); Assignments (15%); Tutorial contribution (5%) + 2-hours Final Unseen Exam (40%) + Project (10%)

Textbooks and Supporting Material:

1- Concepts of Programming Languages, Tenth Edition, Robert W. Sebesta, Pearson, 2013
2- Robert W. Sebesta, Concept of Programming Languages, Addison Wesley, 5th Edition, 2002
3- Terrence W. Pratt, Programming Languages: Design and Implementation, Prentice-Hall, 2002
4- Ravi Sethi, Programming Languages Concepts and Constructions, Pearson Education, 1996
5- Allen B. Tucker, Programming Languages, McGraw Hill, 1988
6- C. Ghezzi and M. Jazayeri, Programming Language Concepts, John Wiley and Sons, 1982

……..

750323, Algorithms

Providing Department: Computer Science, Faculty of IT
Module Coordinator(s):
Year: 3
Credit: 3 credit hours
Prerequisite: 750272 + 721223

Aims:
The aim of this module is to learn how to develop efficient algorithms for simple computational tasks and reasoning about the correctness of them. Through the complexity measures, different range of behaviours of algorithms and the notion of tractable and intractable problems will be understood. The module introduces formal techniques to support the design and analysis of algorithms, focusing on both the underlying

mathematical theory and practical considerations of efficiency. Topics include asymptotic complexity bounds, techniques of analysis, and algorithmic strategies.

Teaching Methods: 38 hours Lectures (2 per week (including two 1-hour midterm exams)) + 10 hours Tutorials (average 1 hour per week)

Learning Outcomes:
When completing this module, you should be able to:
1. understand basic ideas about algorithms (A)
2. develop efficient algorithms for simple computational tasks (B)
3. reason about the correctness of algorithms (B)
4. understand the concepts of time and space complexity, worst case, average case and best case complexities and the big-O notation (A)
5. compute complexity measures of algorithms, including recursive algorithms using recurrence relations (B)
6. understand the range of behaviours of algorithms and the notion of tractable and intractable problems (A, B)
7. know and understand a wide range of searching and sorting algorithms (A, B)

Assessment of Learning Outcomes:
All learning outcomes are assessed by examinations and tutorials. Learning outcomes (4), (5), and (6) are assessed by examinations and coursework.

Contribution to Programme Learning Outcomes:
A1, A2, B1, B2, B3

Synopsis: Introduction, Algorithm definition, Algorithm Analysis; **Mathematical Induction;** Summation Techniques; Recurrence Relations; **Design & Analysis of Algorithms: Divide and conquer**, Greedy Algorithm, Dynamic Programming, Backtracking, Branch-Bound; Lower Bound Theory; Sorting and Searching; NP-Complete Problems: Basic Concepts, NP-Hard & NP-Complete Problem

Modes of Assessment:
Two 1-hour midterm exams (15% each); Tutorial contributions (5%), Coursework (15%); Final written Examination (50%)

Textbooks and Supporting Material:
1- Jon Kleinberg, Eva Tardos, Algorithm design**,** Boston: Pearson Education Limited, 2014.
2- Alwan, Raad F., Design and Analysis of Algorithms, Dar Majdalawi Publication & Distribution, Amman, 2010.
3- Sara Baase, Computer Algorithms: Introduction to Design and Analysis, Third Edition, Addison-Wesley, 2000.
4- Udi Manber, Introduction to Algorithms: a Creative Approach, Addison-Wesley, 1997.
5- T. Cormen, et.al., Introduction to Algorithms, 1999.
6- R. Sedgewick, Algorithms in C++, 2002.
…………………………………………………………………………………………………

750332, Computer Architecture

Course Hours: 3 hours per week, 3 credit hours (total of 48 hours)
Level: 3
Prerequisite: **750233**

Aims:
The aim of this course is to architecture of the computer components. The module emphasizes on the following knowledge areas: Digital components used in the organization and design of digital

computer, serial and parallel transfer, Flow of information and timing signals. Designing a micro-programmed control unit, organization and architecture of input-output and memory, interfacing and communication, memory hierarchy, cache memory, pipelining data path, pipelining control path, and data hazard.

Teaching Methods*Duration*: 16 weeks, 80 hours in total
Lectures: 32 hours (2 hours per week), *Tutorials*: 16 hours (1 per week),
Laboratories: 32 hours, 2 per week

Synopsis: Control unit and Control memory, Micro-instruction, Input-Output Organization, Memory Organization, Cache Memory, Pipelining data path and Control path.

Assessment: Two 1-hour midterm exams (20% each); Assignments (20%); One 2-hours Final Examination (40%).
Text Book: Computer system architecture, M Morris Mano, New Delhi : Prentice-Hall of India, 2008.

Computer System Architecture, M. Morris Mano, Prentice Hall, International edition, 2005. 4rd edition.

……………………………………………………………………………………………………..

750350, Intelligent Systems

Course Hours: 3 hours per week, 3 credit hours (total of 48 hours)
Level: 3
Prerequisite: **250231, 721224**

Aims:
This module aims is to introduce principles of intelligent systems and teach students basic approaches used in this field. These approaches are based on problem solving strategies, knowledge representation and reasoning, uncertainty processing, learning and cooperation.

Teaching Methods *Duration*: 16 weeks, 80 hours in total
Lectures: 32 hours (2 hours per week), *Tutorials*: 16 hours (1 per week),
Laboratories: 32 hours, 2 per week

Synopsis: Introduction, Informed Search, Heuristic Search, Propositional Logic, 1’st Order Logic, Knowledge Representation, Automatic Reasoning, Planning, Neural Networks, Uncertainty, Machine Learning.

Assessment: Two 1-hour midterm exams (20% each); Assignments (20%); One 2-hours Final Examination (40%).

Textbook:
Artificial intelligence a conceptual approach
Author(s)/Editor(s): Anamitra Deshmukh-Nimbalkar
Publisher: New Delhi: Technical Publications, 2014

………………………………………………………………………………………………

731340, Fundamentals of Computer Networks

Providing Department: Computer Information Systems, Faculty of IT
Module Coordinator(s):
Year: 3
Credit: 3 credit hours
Prerequisite: 721224

Aims:
This module is the first module of the curriculum related to the computer network field. Its aim is to provide students with a broad coverage of the basic computer networking concepts of the four layers of ISO, circuit switch, packet switch, etc.
The module, however, does not focus on a detailed study or cover the technologies. The concepts given in this module will be deeply handled in the next level module (750441).

Teaching Methods: 32 hours Lectures (2 per week (including two 1-hour midterm exams)) + 16 hours Tutorial (1 per week) + 16 hours Laboratory

Learning Outcomes:
A student completing this module should be able to:
1. Discuss important network standards in their historical context (A)
2. Describe the responsibilities of the first four layers of the ISO reference model. (A)
3. Discuss the differences between circuit switching and packet switching along with the advantages and disadvantages of each. (A, B)
4. Explain how a network can detect and correct transmission errors. (A, B)
5. Illustrate how a packet is routed over the Internet. (C)
6. Install a simple network with two clients and a single server using standard host-configuration software tools such as DHCP. (C, D)

Assessment of Learning Outcomes:
Learning outcomes (1) - (3) are assessed by examination and tutorials. Learning outcomes (4) – (6) are assessed by assignments and seminars.

Contribution to Programme Learning Outcomes
A3, A4, B2, C6, D2, D5, D6.
Synopsis: Introduction; Network Model; Data and Signal**;** Digital signal; Analog signal; Switching; Error Detection and Control; Error Detection and Control; Data Link Control; Multiple Access; Network Layer: Logical Addressing; Network Layer: Delivery, Forwarding, and Routing; Network Layer: Delivery, Forwarding, and Routing; Process-to process Delivery; Congestion Control and Quality of service.

Modes of Assessment:
Two 1-hour midterm exams (20% each); Assignments (20%); One 2-hours Final Examination (40%)

Textbooks and Supporting Material:
1- Behrouz A. Forouzan, Data Communications and Networking, McGraw Hill Higher Education, Fourth Edition, 2007
2- Andrew S. Tanenbaum, Computer Networks, Prentice Hall, Last Edition.
<u>Website(s):</u>
www.mhhe.com/forouzan
……………………………………………………………………………………………

750335, Operating Systems

Providing Department: Computer Science, Faculty of IT
Module Coordinator:
Year: 3
Credit: 3 credit hours

Prerequisite: 750332
***Prerequisite for*:** 750334

Aims:
The aims of this module are to introduce the basic principles of computer systems organization and operation; to show how hardware is controlled by program at the hardware/software interface; to outline the basic OS resource management functions: memory, file, device (I/O), process management, and OS security/protection. Two concrete examples of operating systems are used to illustrate how principles and techniques are deployed in practice.

Teaching Method: 40 hours Lectures (2-3 per week) + 8 hours Tutorials (1 each fortnight)

Learning Outcomes:
On completing the module, students should:
1- Have knowledge and understanding of the overall structure and functionality of a modern operating system and of its interactions with the underlying computer hardware and overlying user-program. (A)
2- Have knowledge and understanding of the operation of the following major components of an operating system: the I/O device manager; the memory manager; the process manager; the file manager; OS security/protection manager (A)
3- Have the ability to design and implement (an emulation of) a prototypical process manager. (B, C)
4- Be aware of how fundamental techniques in (1) and (2) are applied in practice in two distinct modern operating systems. (A)

Assessment of Learning Outcomes:
Learning outcomes (1) and (2) are assessed by examination. Learning outcome (3) is assessed via course project. Learning outcome (4) is not formally assessed.

Contribution to Programme Learning Outcomes
A3, B3, C5.

Synopsis*:* Operating System overview; Operating System Structures: System components, Operating system services, System calls, System structures, Virtual machine; Processes: Process concept, Process scheduling, Operation on process, Cooperative process, Inter process communication; Threads: Thread overview, Benefits, User and kernel threads, Multithreading model, Solaris 2 threads; CPU Scheduling: Basic concept, Scheduling criteria, Scheduling algorithm, Thread scheduling, Algorithm evaluation; Process synchronization and mutual exclusion: Critical section problem, Two task solution, Synchronization hardware, Semaphore, Classical synchronization problem; Deadlock and starvation: System model, Deadlock characterization, Method for handling deadlock, Deadlock prevention, Deadlock avoidance, Deadlock detection, Recovery from deadlock; Memory management: Background, Swapping, Paging, Virtual memory, Background, Demand paging, Page replacement, Allocation of frame, Thrashing; File system implementation and management: File concept, Access method, Directory structure, Protection, File system structure, Allocation method, Free space management, Directory implementation, Efficiency and performance, I/O management and disk scheduling, Application I/O interface, Kernel I/O subsystem, I/O request handling, Disk structure, Disk scheduling, Disk management, Swap space management, Disk reliability, Stable storage implementation

Modes of Assessment:
Two 1-hour midterm exams (20% each); Assignments (20%); One 2-hours Final Examination (40%)

Textbooks and Supporting Material:
1- Operating System Concepts, 9th Edition International Student Version
Abraham Silberschatz, Peter B. Galvin, Greg Gagne
ISBN: 978-1-118-09375-7 Addison-Wiley , **2013**

2- A. Silberschatz and Peter Galvin, Applied Operating Systems Concepts, First edition, John Wiley & sons, Inc, 2000
3- J. Bacon, Concurrent Systems: Database and Distributed Systems, 2nd Edition, (ISBN 0-201-177-676), Addison Wesley, 1998.
4-A. S. Tanenbaum, Modern Operating Systems, Prentice Hall, 1992

……………………………………………………………………………………………………

750399, Practical Training

Providing Department: Computer Science, Faculty of IT
Module Coordinator(s):
Year: 3
Credit: 3 credit hours
Prerequisite: 721210; It is implanted according to the Faculty regulations

Aims:
The main aim of this module is that students will have practice in different industrial, commercial, administrative enterprises or companies. By this module, students may apply, in the real world, what they have learned during the first three years of their study in the University. The module also aims to teach students how to be self-confident when they face problems in their practical life.

Teaching Methods:
Duration: at least 9 weeks (18 training hours per week at least). This may be distributed onto at most two semesters.

Regulations for Training:
Students have to register on at most 15 credit hours in the semester in which they register on the practical training module.
2- Students must be full-time trainees for at least 2 days per week.
3- Students who take this module should arrange their timetable for other modules in a way that enables them to enrol in the pre-specified enterprise or company at least two days per week during the semester period.
4- The student has to get an official letter from the Faculty requesting a placement, and the Faculty provides a standard document that the placement provider could use to confirm that appropriate opportunities would be available to the student.
5- There is an academic supervisor for any trainee from the Department in addition to the supervisor from the placement provider.
6- Student should submit a report at the end of the training period.
7- At the end of the training period, the student and the placement provider fill some forms that will be used in assessing the student.
8- More information about training can be found in the Practical Training Handbook.

Learning Outcomes:
A student completing this module should:
1- be able to prepare and write any technical report. (C)
2- be prepared for any practical work (C)
3- be able to use IT skills (D)
4- learn how to work with and for others. (D)

Assessment of Learning Outcomes:
Learning outcome (1) is assessed by report evaluation, learning outcomes (3) – (4) are assessed by the observation of the training committee.

Contribution to Programme Learning Outcomes

C2, C3, D1, D2, D3

Synopsis: This module requires no syllabus, but any previously taught module will be valuable and can be applied in the practice.

Modes of Assessment:
A committee from the department supervises the students along their training period, where one supervisor is assigned on one group of students. The student should submit a technical report to this committee in 2 weeks time after completing the training session. In addition, the trainer body presents a report to the committee. The grade "pass" is given to students who complete the training requirements successfully and discuss their reports with the supervision committee.
………………………………………………………………………………………………………

750324, Compiler Construction

Providing Department: Computer Science, Faculty of IT
Module Coordinator(s):
Year: 3
Credit: 3 credit hours
Prerequisite: 750223

Aims:
This module aims to show how to apply the theory of language translation introduced in the prerequisite courses to build compilers and interpreters. It covers the building of translators both from scratch and using compiler generators. In the process, the module also identifies and explores the main issues of the design of translators. Topics include compiler design, lexical analysis, parsing, symbol tables, declaration and storage management, code generation, and optimization techniques. The construction of a compiler/interpreter for a small language is a necessary component of this module, so students can obtain the necessary skills.

Teaching Methods: 32 hours Lectures (2 per week) + 14 hours Tutorials (1 per week) (except the last two) + 2 hours Seminars

Learning Outcomes:
A student completing this module should be able to:
1. Understand the structure of compilers. (A, B)
2. Understand the basic techniques used in compiler construction such as lexical analysis, top-down, bottom-up parsing, context-sensitive analysis, and intermediate code generation. (B, C)
3. Understand the basic data structures used in compiler construction such as abstract syntax trees, symbol tables, three-address code, and stack machines. (A, D)
4. Design and implement a compiler using a software engineering approach. (A, B)
5. Use generators (e.g. Lex and Yacc) (A)

Assessment of Learning Outcomes:
Learning outcomes (1), (2), and (3) are assessed by examinations, tutorials and coursework. Learning outcomes (4) and (5) are assessed by projects and seminars.

Contribution to Programme Learning Outcomes:
A2, A3, B2, B3, C1, C5, C6, D5

Synopsis: Introduction to Compilers: The role of language translation in the programming process; Comparison of interpreters and compilers, language translation phases, machine-dependent and machine-independent aspects of translation, language translation as a software engineering activity; Lexical Analysis: Application of regular expressions in lexical scanners, hand coded scanner vs. automatically generated scanners, formal definition of tokens, implementation of finite state automata; Syntax Analysis: Revision of formal definition of grammars, BNF and EBNF; bottom-up vs. top-down parsing, tabular vs. recursive-descent parsers, error handling; Parsers Implementation: automatic generation of tabular parsers, symbol table management, the use of tools in support of the translation process; Semantic Analysis: Data type as set of values with set of operations, data types, type- checking models, semantic models of user-defined types, parametric polymorphism, subtype polymorphism, type-checking algorithms; Intermediate Representation, code generation: Intermediate and object code, intermediate representations, implementation of code generators; Code generation: code generation by tree walking; context sensitive translation, register use; Code optimization: Machine-independent optimization; data-flow analysis; loop optimizations; machine-dependent optimization; Error Detection and Recovery; Error Repair, Compiler Implementation; Compiler design options and examples: C Compilers, C++, Java, and YACC Compilers

Modes of Assessment:
Two 1-hour midterm exams (15% each); Seminars (5%); Assignments (15%); 2-hours Final Exam (50%)

Textbooks and Supporting Material:

1- Kaushal Kishor Rastogi, Compiler Design, New Delhi: Global Academic Publishers & Distributors, 2014.
2- A. Aho, R. Sethi, J. D. Ullman, Compilers: Principles, Techniques, and Tools, Addison-Wesley, 1986
3- W. Appel, Modern Compiler Implementation in Java, Prentice Hall, 2002
4- D. Watt, Brown, Programming Language Processors in Java: Compilers and Interpreters, Prentice hall, 2000

………..

750362 Database Applications Programming

Course Hours: 3 hours per week, 3 credit hours (total of 48 hours)

Level: 3

Prerequisite: **0731221**

Aims:

This module aims to strengthen database concepts by applying SQL and PL/SQL practically and know how to write code using these languages. It also aims to give the students some new concepts regarding databases such as data definition language, data manipulation language and data control language in addition to blocks, functions, triggers, cursors, repetition statements and conditional statements. And finally it gives the student the ability to work on oracle forms, reports, graphics and oracle 11G

Teaching Methods*Duration*: 16 weeks, 80 hours in total
Lectures: 32 hours (2 hours per week), *Tutorials*: 16 hours (1 per week),
Laboratories: 32 hours, 2 per week

Synopsis: : data and database definitions, relational database main terms such as entities, relationships, keys, etc, and relational database limitations, Structured Query Languages (SQL), SQL* plus, DDL, DML, DCL, PL/SQL anonymous blocks, repeatition statements, if statements, etc, PL/SQL programming using Functions, Triggers, Procedures, Cursors and Packages, Oracle Developer/2000; Oracle forms, Reports, Graphics, DBA.

.

Assessment: Two 1-hour midterm exams (20% each); Assignments (20%); One 2-hours Final Examination (40%).

Textbook:
1- Beginning Oracle SQL : for Oracle Database 12c, Lex De Haan, *Berkeley: Apress, 2014.*

2- Database systems: models, languages, design, and application programming, Elmasri , Ramez, Boston: Pearson 2011 .

3- Oracle PL/SQL Programming, 5th Edition
Covers Versions Through Oracle Database 11g Release 2
By Steven Feuerstein, Bill Pribyl
Publisher: O'Reilly Media
Released: September 2009
……………………………..………………………………..……………………………..……………

750472, Modelling and Simulation

Providing Department: Computer Science, Faculty of IT
***Module Coordinator(s)*:**
Year: 4
Credit: 3 credit hours
Prerequisite: 210103 + 721211

Aims:
This module aims to present methodologies used in computer simulation, to show simulation as complementary to laboratory field experimentation in the development of better understandings of complex phenomena and to discuss analysis, appropriate use, and limitations of simulation models. The module presents applications of software simulation process with supporting techniques.

Teaching Methods: 33 hours (2-3 per week) + 12 hours Tutorials (1 each fortnight) + 3 hours Seminars (in last 3 weeks)

Learning Outcomes:
On successful completion of this module, student will:
1- be able to describe the components of continuous and discrete systems and simulate them (A, B)
2- understand different methods for random number generation (A)
3- be able to model any system from different fields (B)
4- know how to simulate any discrete system using queuing systems (B, C)
5- Be able to implement numerical algorithm to meet simple requirements, expressed in English (B)
6- Be able to work effectively with others (D)
7- Be able to discuss the simulation methods and select the suitable technique on the problems. (B)
8- Have a clear understanding of the need for the development process to initiate the real problem. (A)
9- Have a clear understanding of principles and techniques of simulation methods informed by research direction. (A, B, C)

Assessment of Learning Outcomes:
Learning outcomes (1), (3), (5), and (7) are assessed by assignments and Seminars. Learning outcomes (2) (4), (8), and (9) are assessed by examination and assignments.

Contribution to Programme Learning Outcomes:
A2, A3, B1, B2, B3, C1, C3, C5, D1, D3

Synopsis: Introduction and overview of systems and simulation; Modelling; Scope of simulation; Types of simulations; Random numbers and random variables; Gathering Observations; Overview of programming languages for simulation

Mode of Assessment:
Two 1-hour midterm exams (20% each); Assignments (20%); One 2-hours Final Examination (40%)

Textbooks and Supporting Materials:

1 Discrete Event Simulations - Development and Applications
By Eldin Wee Chuan Lim, ISBN 978-953-51-0741-5, 196 pages, Publisher: InTech, Chapters published September 06, 2012

2- Fracis Neelamkavil, Computer Simulation and Modelling, John Wiley & Sons, 1989
3- R. M. Davies and R. M. O'Keefe, Simulation Modelling with Pascal, Prentice Hall, 1989
4- J. A. Payne, Introduction to Simulation: Programming Techniques and Methods of Analysis, McGraw-Hill, 1988
5- Banks, Carson, Nicol, Discrete Event System Simulation, Prentice Hall,

……………………………………………………………………………………

750497, Research Project (1) + 750498, Research Project (2)

Providing Department: Computer Science, Faculty of IT
Module Coordinator(s):
Year: 4
Credit: 3 credit hours
Prerequisite: 750398

General Descriptions:
The graduation project consists of a single project on which the student works over a period of 16 weeks that can be extended to 32 weeks (2 semesters). It is assumed that the student spends a nominal 192 hours (or 384 hours), the equivalent of 12 hours per week, working on this. There are three deliverables: demonstration, discussion, and a written report.

A student works under the supervision of a member of staff, the Supervisor. Most of the projects involve three students working together on the same project; apart from these, all students do different projects.

- How to choose a project
- Organisation for projects
- Demonstrations
- Report Standards
- Staff List

Aims:
The aims for the project work done in the fourth year are:
1- To manage and execute a substantial project in a limited time.
2- To identify and learn whatever new skills are needed to complete the project.
3- To apply design and engineering skills in the accomplishment of a single task. In this context the skills mentioned may be in the general area of design and engineering in its broadest sense, or may be very specifically related to particular tools.

Teaching methods: Duration: 32 weeks (2 semesters) starts in first semester: Lectures: 6 or 7 in total, spread through the 2 semesters + Laboratories: none scheduled, 120 hours expected through semester

Learning Outcomes:
On completion of this module, a student should have

1. Used the project supervisor appropriately as project consultant or customer. (D)
2. Planned, executed and completed a significant design and, as appropriate, implementation within the time budget available. (B, C)
3. Given a demonstration showing practical competence and demonstrating the results of the project. (C).
4. Documented the project in a final report. (C)

Assessment of Learning Outcomes:
There is no examination.
Learning outcome (1) is assessed by the supervisor. Learning outcomes (3) is assessed by the project examination committee and by judging the demonstration. Learning outcome (4) is assessed by judging the report. Learning outcome (2) is assessed by all of these mechanisms.

Modes of Assessment:
Supervisor mark: 35% + Project Examination Committee mark: 65% (demonstration 20%, Report 25%, discussion 20%)

Contribution to Programme Learning Outcomes
B1, B2, B3, C1, C2, C3, D1, D2, D3, D4, D5

Syllabus
The occasional lectures are on topics of particular interest to students doing a project in their final year.

- Overview of projects and project assessment.
- Career advice.
- How to give a seminar.
- Writing English.
- How to give a demonstration.
- How to write a project report.

……………………………….…………………………….….……………………………….………

750445, Wireless and Mobile Computing

Providing Department: Computer Information Systems, Faculty of IT
Module Coordinator(s):
Year: 4
Credit: 3 credit hours
Prerequisite: 731340

Aims:
To impart an understanding of fundamental concepts underlying current developments in mobile communication systems and wireless computer networks.

Teaching Methods: 32 hours Lectures (2 per week) + 8 hours Tutorials (1 per 2 weeks) + 8 hours Projects/Seminars (1 per 2 weeks)

Learning Outcomes:
At the end of the course, students will have acquired the following knowledge and skills.

1. Understanding of characteristics of radio propagation and interference in multipath propagation and channel model description (A)
2. Understanding of a range of digital transmission systems as used for applications in mobile telephony and wireless computer networks, pulse shaping and equalisation techniques (A)
3. Understanding of the issues and techniques used in the design of Medium Access Control protocols for wireless Networks (A)
4. Understanding of the systems, protocols and mechanisms to support mobility for mobile internet users (A)
5. The ability to investigate fundamental aspects of transmission and modulation by writing MATLAB programs. The experience of using an industrial standard network simulation package(A,B,C,D)

Assessment of Learning Outcomes:

Learning outcomes (1-6) are assessed by examinations, tutorials, seminars and projects.

Contribution to Programme Learning Outcomes:
A1 – A5, B1, B3, C3, C4, C5, D6

Synopsis: Introduction to wireless networking, Advantages and disadvantages of wireless networking, Characteristics of radio propagation, Fading, Multipath propagation, Introduction to digital transmission, Definition of bit-rate and signalling rate, Introduction to synchronous transmission, The need for pulse shaping, synchronisation and line-coding, Calculation of bit-error probabilities when the channel is affected by the addition of Gaussian noise, Narrowband digital modulation, The need for modulation, Binary and multi-level (M-ary) amplitude-shift keying (ASK), frequency-shift keying (FSK) and phase-shift keying (PSK), Wideband modulation techniques to cope with intersymbol interference , Direct sequence spread spectrum Adaptive Equalization Orthogonal frequency division multiplex, Medium Access Control (MAC), MAC protocols for digital cellular systems such as GSM, MAC protocols for wireless LANs such as Hidden and exposed terminals, Collision Avoidance (RTS-CTS) protocols, Protocols supporting mobility, Mobile network layer protocols such as mobile-IP, Dynamic Host Configuration Protocol (DHCP), Mobile transport layer protocols such as mobile-TCP, indirect-TCP, Wireless Application Protocol (WAP).

Modes of Assessment:
Two 1-hour midterm exams (15% each); Assignments (10%); Seminars (10%); Final Examination: 2-hours written exam (35%) + Project (15%)

Textbooks and Supporting Material:
1. Wireless and mobile networks, Sunilkumar S. Manvi and Mahabaleshwar S. Kakkasageri, Wiley, 2010
2. Mobile computing, V. Jeyasri Arokiamary, Pune: Technical Publications, 2014
3. Schiller, Mobile communications, ISBN: 0-321-12381-6, Addison-Wesley, 2003
4. T.S. Rappaport, Wireless communications; Principle and Practice, ISBN: 0-13-375536-3
5. A S. Tanenbaum, Computer Networks (Fourth Edition), Publisher: Prentice Hall PTR; ISBN: 0130661023; August, 2002.

……………………………………………………………………………………………………. . …. . .

750413, Concurrent and Distributed Programming

Providing Department: Software Engineering,, Faculty of IT
Module Coordinator(s):
Year: 3
Credit: 3 credit hours
Prerequisite: 750215

Aims:

The aim of this module is to study, learn, and understand the main concepts of concurrency. Hardware and software features to support concurrency, language features for concurrent and distributed systems, and concurrent and distributed algorithms and middleware.

Teaching Methods: 35 hours Lectures (2-3 hours per week) + 5 hours Laboratory (1 per 3 weeks) + 8 hours Seminars (1 each fortnight)

Learning Outcomes:
A student completing this module should be able to:
1. Outline the potential benefits of concurrent and distributed systems. (A).
2. Apply standard design principles in the construction of these systems. (A, B)
3. Select appropriate approaches for building a range of distributed systems, including some that employ middleware. (B)
4. Summarize the major security issues associated with distributed systems along with the range of techniques available for increasing system security. (A)

Assessment of Learning Outcomes:
Learning outcomes (1) – (4) are assessed by examinations, assignments, and seminars.

Contribution to Programme Learning Outcomes
A2, A5, B2, B3.

Synopsis: Concurrent model of execution; interleaving; atomic operation; critical sections and mutual exclusion; deadlock; starvation; invariants; Concurrent and distributed algorithms: producer-consumer; reader-writer problems; dining philosophers; Architectural features to support concurrent and distributed systems; Language features for concurrent and distributed systems; Performance evaluation; Middleware.

Modes of Assessment:
Two 1-hour midterm exams (15% each); Assignments (10%); Seminars (10%); Final Examination: 2-hours written exam (30%) + defended project (20%)

Textbooks and Supporting Material:
- Concurrent programming: algorithms, principles, and foundations, Raynal,Michel, Heidelberg: Springer, 2013.

……..…..

750446, Information Security

Providing Department: Computer Science, Faculty of IT
Module Coordinator(s):
Year: 4
Credit: 3 credit hours
Prerequisite: 731340

Aims:
Upon successful completion of the course, the student will be knowledgeable of network security principles and implementation, including the technologies used and principles involved in creating a secure computer networking environment; authentication, types of attacks and malicious code that may be used against a network; threats and countermeasures for e-mail, Web applications, remote access, and file and print services; security topologies; technologies and concepts used for providing secure communications channels, secure internetworking devices, and network medium; intrusion detection systems, firewalls, and physical security concepts; security policies, disaster recovery, and computer forensics; and daily tasks involved with managing and troubleshooting security technologies.

Teaching Methods: 40 hours Lectures (2-3 hours per week) + 4 hours Tutorials (1 per 3 weeks) + 4 hours Lab (1 per 3 weeks)

Learning Outcomes:
Students completing this module should be able to*:*

1. To provide the student with basic knowledge of general security concepts, including authentication methods, common network attacks and how to safeguard against them. (A)
2. To provide the student with basic knowledge of communication security, including remote access, e-mail, the Web, directory and file transfer, and wireless data. (A)
3. To provide the student with basic knowledge of infrastructure security, including various network devices and media, and the proper use of perimeter topologies such as DMZs, Extranets, and Intranets to establish network security. (B)
4. To provide the student with basic knowledge of cryptography basics, including the differences between asymmetric and symmetric algorithms, and the different types of PKI certificates and their usage. (B,C)
5. To provide the student with basic knowledge of operational/organizational security, including its relationship to physical security, disaster recovery, and business continuity, as well as computer forensics and how it relates to further avenues. (C,D)

Assessments of Learning Outcomes:

Learning outcomes (1) and (2) are assessed by examinations. Learning outcomes (3) and (4) are assessed by assignments and research.

Contribution to Programme Learning Outcomes
A3, A5, B2, C2, C4, C5, D1, D4, D5.

Synopsis: Layered communication architecture: layers, services, protocols, layer entities, service access points, information Security Fundamentals, Attackers and Their Attacks, Security Basics, Security Baselines, Securing the Network Infrastructure, Web Security, Protecting Advanced, Communications, Scrambling Through Cryptography, Using and Managing Keys, Operational Security, Policies and Procedures, Security Management, Advanced Security and Beyond

Modes of Assessment:
Two 1-hour midterm exams (15% each); Assignments (20%); Final Examination: 2-hours written exam (35%) + a research project (15%).

Textbooks and Supporting Material:

1- Network security essentials : applications and standards, William Stallings, Harlow: Pearson Education Limited, 2014.
2- Information security and cyber laws, Sanjeev Puri, New Delhi: Technical Publications, 2014
3- Security+ Guide to Network Security Fundamentals, Second Edition, by Mark Chiampa Course Technology, 2004

4- William Stallings, Wireless Communications & Networks, 2nd edition, Prentice-Hall Pearson, 2005

3.5 Elective Modules

Each student should select 2 modules out of a list of 4 modules according to his/her interest. The Department has a list of elective modules, which can be updated according to the staff expertise and the most recent trends in the field of Computer Science. The current list of such modules is shown in Table (3-4), where some modules are marked with (R) to indicate that these modules are research-oriented according to the staff expertise.

Table (3-4) Elective Modules in Computer Science Department

Module No.	Module Name
0731423	Data mining *
0750460	Special Topics
0750464	Information retrieval
0750474	Digital Image Processing

……………………………..……………………………..……………………………..……………

731423, Data Mining

3 hours per week (48 hours in total), 3 credit hours, Fourth year, any semester, prerequisite: **731221**
Teaching Method: *3*0 hours lectures (2 hours per week) + 10 hours seminars (1-2 hours per 2 weeks) + 5 hours tutorials (1 per 2 weeks).
Aims*:* The main goal is to provide the student with an understanding of the concepts and elements of data warehousing and data mining both from a business and technology prospective, including hands-on experience with a sample of tools used in decision support environments. At the conclusion of this course the students will be able to:
Explain the purpose for developing a data warehouse, including the differences between operational and decision support systems.
Describe and use the dimensional modelling technique for designing a data warehouse.
Describe the architecture of a data warehouse.
Understand the project planning aspect of building a data warehouse.
Use OLAP analysis with contemporary analysis and visualization tools.
Understand and explain the purpose of data mining.
Understand the knowledge discovery process.
Understand several different data mining techniques such as market basket analysis, clustering, genetic algorithms, as well as which kinds of problems these techniques are applicable to.
Textbook:
Modern Data Warehousing, Mining, and Visualization, by George M. Marakas, 2003, Prentice-Hall.
References:
The Data Warehouse Toolkit, by Ralph Kimball and M. Ross, 2002, Wiley
Synopsis:

This course covers the fundamentals of data warehousing architecture and the issues involved in planning, designing, building, populating and maintaining a successful data warehouse. The course introduces students to data mining and how it relates to data warehousing. Specific topics covered include the logical design of a data warehouse, the data staging area and extract-transform-load processing, the use of multi-dimensional analysis using OLAP techniques, and coverage of the knowledge discovery process including common data mining modelling techniques.

This course covers the fundamentals of data warehousing architecture and the issues involved in planning, designing, building, populating and maintaining a successful data warehouse. The course introduces students to data mining and how it relates to data warehousing. Specific topics covered include the logical design of a data warehouse, the data staging area and extract-transform-load processing, the use of multi-dimensional analysis using OLAP techniques, and coverage of the knowledge discovery process including common data mining modelling techniques.

Assessment: Two 1-hour unit tests (20% each) + Assignments (20%) + 2-hours final exam (40%).

…………………………………………………………………………………………………….

750464 Information Retrieval

Providing Department: Computer Information Systems, Faculty of IT
Module Coordinator(s):
Year: 4
Credit: 3 credit hours
Prerequisite: 731221

Aims:
Information Retrieval (IR) is a really HOT subject these days. All thanks to the World Wide Web, Web Search, and our friends at Google, Yahoo!, MSN, and all the other search engines that have come and gone!!! But, there's a lot that happens between the typing of 2-3 keywords in a small box at the User Interface, and receiving the results. In the next ten weeks we'll look at issues surrounding information retrieval systems. We will examine information system design and evaluation issues, and look under the hood of the search engines to pick at what's going on and why.

Teaching Methods: 36 hours Lectures (2-3 hours per week) + 8 hours Seminars (1 per 2 weeks) + 4 hours Laboratory (1 per 3 week)

Learning Outcomes:
A student completing this module unit should be able to:

1- become familiar with basic issues and current practice in IR (A)
2- familiarize student's selves with IR tools (B)
3- review important research in IR. (C,D)

Assessment of Learning Outcomes

Learning outcomes (1-5) are assessed by examinations, tutorial and in the laboratory. Learning outcomes (6-8) and (10)are assessed by tutorials and in laboratory. Learning outcomes (9) and (11) are assessed by seminars and/or workshop

Contribution to Programme Learning Outcomes
A1, A2, A3, A4, A5, A6, C1, C2, C3, C4, D1, D2, D4, D5.

Synopsis: Introduction to IR. Overview of the components of an IRS, Queries, Documents, Indexing, Theories & Models in IR (Retrieval Techniques) for text, hypermedia, web, Exploring Google search, Queries and Information Needs, Document Analysis, Structure of documents, parsing, stemming, morphological analysis, tokenization, Retrieval Techniques, Exact Match vs. Partial Match Weighted Ranked Retrieval, Vector Space & Probabilistic retrieval models, Relevance Feedback, Retrieval Techniques, Retrieval Techniques, Exact Match vs. Partial Match Weighted Ranked Retrieval, Vector Space & Probabilistic retrieval models, Relevance Feedback, Query Processing for IR ,Query Formulation, Expansion, Refinement, Web search. Hypertext, Internet Search Engines, Link Analysis, Evaluation of IRS, Performance Measures. Relevance, User centered evaluation of IR systems, Evaluation of IRS, Evaluating Exploratory Search Systems, Web search Crawlers, Web graph. Log Analysis, evaluation revisited, Faceted Search.

Mode of Assessment:

Two 1-hour midterm exams (15% each); Coursework (15%); Tutorial contribution (5%); Final Examination: written (unseen) exam (35%) + Lab Exam (15%)

Textbook and Supporting Material:

Title: Information retrieval models : foundations and relationships
Author(s)/Editor(s): Poelleke,Thomas
Publisher: Morgan & Claypool, 2013
Website: http://www.morganclaypool.com/doi/abs/10.2200/S00494ED1V01Y201304ICR027
……………………………………………………………………………………………………….

750491, Special Topics

Providing Department: Computer Science, Faculty of IT
Module Coordinator(s):
Year: 4
Credit: 3 credit hours
Prerequisite: Department Agreement

Aims:
This module aims to offer any recent topic in computer science. The chosen topic may be different from semester to another.

Teaching Methods: 48 Lectures or it depends on the chosen topic that might include seminars hours as well.

Learning Outcome:
It depends on the chosen topic.

Assessment of Learning Outcome:
It depends on the chosen topic.

Modes of Assessment:
It depends on the chosen topic.

Synopsis: For this module, the department can choose any recent topic to cover it within one semester.

Textbooks and Supporting Material:
According to the selected topic

750474, Digital Image Processing

Providing Department: Computer Science, Faculty of IT
***Module Coordinator(s)*:**
Year: 4
Credit: 3 credit hours
***Prerequisite:* 750272 + 750323**

Aims:

The main objective of this module is to make it possible to effectively communicate visual results. This course prepares students in the fundamentals of digital image processing as used in various applications as outlined above and illustrates the various effects one can achieve with digital images and how to extract fundamental information.

Teaching Methods:

Duration: 16 weeks in first semester, 60 hours in total
Lectures: 32 hours (2 hours per week),
Tutorials: 13 hours, 1 per week,
Laboratories: 16 hours, 1 per week
Project Presentation: 3 hours

Learning Outcomes:

- **Knowledge and understanding**
  - Have a knowledge and understanding of the structure of an interactive image processing.
  - Have a knowledge and understanding of image transformations.
  - Have a knowledge and understanding of techniques for representing color/mono images.
  - Have a knowledge and understanding of interaction techniques.

Assessment Instruments

Allocation of Marks	
Assessment Instruments	**Mark**
First examination	**15%**
Second examination	**15%**
Final Exam (written unseen exam) Final project (defended)	**40 %** **10%**
Reports, assignments, Quizzes, Home works, Tutorials	**20%**
Total	**100%**

** Make-up exams will be offered for valid reasons only with consent of the Dean. Make-up exams may be different from regular exams in content and format.*

Synopsis

Overview, Computer imaging systems, Image analysis, preprocessing, Human visual system, image model, Image enhancement, gray scale modes, histogram mode, Discrete transforms, fourier, discrete cosine, walsh-hadamard, Haar, PCT, filtering, filtering, wavelet transform, pseudocolor, Image enhancement, sharpening, smoothing, Image restoration, overview, system model, noise.

Module References

Students will be expected to give the same attention to these references as given to the Module textbook(s)

Textbooks:

1-Introduction to digital image processing, William K. Pratt, London: CRC Press, Taylor & Francis Group, 2014

2- Digital Image Processing Using Matlab, Gonzalez, R.C., Woods, R.E. and Eddins, S.L, Gatesmark Publishing; 2nd edition (2009)

Other References

- Computer Imaging: Digital Image Analysis and Processing , SE Umbaugh, Publisher: CRC Press, 2005
- The Scientist and Engineer's Guide to Digital Signal Processing, Steven W. Smith
- Digital Image Processing: 3rd Edition, William K. Pratt
- *1a. Computer Vision and Image Processing: A Practical Approach Using CVIPtools - S. E Umbaugh, Prentice Hall PTR, Upper Saddle, NJ, 1998*
- Digital Image Processing - R.C.Gonzalez & P.Wintz
- Robot Vision - B.K.P.Horn
- Computer Vision - D.H.Ballard & C.M.Brown
- Syntactic Pattern Recognition : An introduction -R.C.Gonzalez and M.G.Thomason
- Pattern Recognition - A Statistical Approach - P.A. Devijver and J. Kittler
- Digital Image Processing - W. K. Pratt
- Fundamentals of Digital Image Processing - A.K. Jain
- Digital Picture Processing - A. Rosenfeld and A.C. Kak
- Pattern Classification and Scene Analysis - R.O. Duda and P.E. Hart
- Object Recognition by Computer - W.E.L. Grimson
- Digital Pictures - A.N. Netravali and B.G. Haskell
- Vision in Man and Machine - M.D. Levine
- Pattern Recognition Statistical, Structural and Neural Approaches, R.J Schalkoff, John Wiley & Sons NY
- Digital Image Processing and Computer Vision, R.J. Schalkoff, Wiley
- Artificial Intelligence: An Engineering Approach, R.J. Schalkoff, McGraw-Hill
- Algorithms for Graphics and Image Processing, Theo Pavlidis, Computer Science Press, call no.: T385.P381982
- Handbook of Pattern Recognition and Image Processing, K.S. Fu and T.Y. Young, Academic Press
- The Image Processing Handbook, John C. Russ, CRC Press SIUE Library call #: TA1632.R881992 (reference)

Journals

- IEEE Transactions on Pattern Analysis and Machine Intelligence

- IEEE Transactions on Computers
- Pattern Recognition
- Computer Vision, Graphics and Image Processing
- IEEE Transactions on Medical Imaging
- Computerized Medical Imaging and Graphics
- IEEE Transactions on Image Processing
- IEEE Engineering in Medicine and Biology
- IEEE Transactions on Signal Processing
- IEEE Transactions on Neural Networks
- IEEE Transactions on Geoscience and Remote Sensing
- Photogrammetric Engineering and Remote Sensing
- International Journal of Remote Sensing
- Journal of Visual Communication and Image Representation

Website(s):

- www.cee.hw.zc.uk/~pjbk/pathways/cpp1/cpp1.html
- www.edm2.com/0507/introcpp1.html
- www.doc.ic.ac.uk/~wjk/C++intro
- www.cprogramming.com/tutorial.html
- www.cs.umd.edu/users/cml/cstyle/ellemtel-rules.html
- www.deakin.edu.au/~agoodman/Ctutorial.html
- www.tldp.org/howto/c++programming.howto.html
- www.vb-bookmark.com/cpptutorial.html

Digital Image Processing PowerPoint Lecture Slides

CVIPbook_PPLec\Chapter1.ppt
CVIPbook_PPLec\Chapter2.ppt
CVIPbook_PPLec\Chapter3a.ppt
CVIPbook_PPLec\Chapter7.ppt
CVIPbook_PPLec\Chapter8.ppt
CVIPbook_PPLec\Chapter5.ppt
CVIPbook_PPLec\Chapter9.ppt
CVIPbook_PPLec\Chapter10_439.ppt

…….

Appendix A
Guidance Plan for the
2018 / 2019

Year	Semester	Module Number	Module Title	Prerequisi	Type of Requirements
First	First (18 Credit Hours)	0114101	Arabic Language Skills (1)	0114099	(UR)
		0130101	English Language Skills (1)	0130099	(UR)
			University Elective (1)	------	(UR)
		0750113	Programming Fundamentals (1)	------	(FR)
		0250101	Differentiation and integration (1)	------	(SR)
		0731110	Introduction to Information Systems and Technology	------	(FR)
	Second (18 Credit Hours)	0111101	National Education	------	(UR)
		0780110	Introduction to Internet and Web Technology	------	(FR)
		0750114	Programming Fundamentals (2)	0750113	(FR)
		0750120	Discrete Mathematics	0750099	(DR)
		0721111	Software Engineering Fundamentals	0731110	(SR)
		0130102	English Language Skills (2)	0130101	(UR)
Second	First (18 Credit Hours)	0721240	Computing Ethics	0731110	(FR)
		3072122	Object-Oriented Programming	0750114	(FR)
		0731213	Introduction to Web Programming	0750114	(FR)
		0750230	Digital Logic Design	0731110	(DR)
		0750224	Theory of Computation	0250104	(DR)
		0750272	Numerical Analysis	0750114	(DR)
	Second (18 Credit Hours)	0721224	Data Structures	0721223	(SR)
		0731221	Database Fundamentals	0721223	(SR)
		0750233	Computer Organization and Design	0750230	(DR)
		0250241	Linear Algebra (1)	0250101	(SR)
		0750215	Visual Programming	0721223	(FR)
		0250231	Introduction to Statistics and Probabilities		(SR)
Third	First (18 Credit Hours)	0731321	Systems Analysis and Design	0721111	(SR)
		0750321	Concepts of Programming Languages	0721224	(DR)
		0750323	Algorithms	0721224	(DR)
		0750332	Computer Architecture	0750233	(DR)
		----	University Elective (2)	-----	(UR)
		0750350	Intelligent Systems	0250231	(DR)
	Second (15 Credit Hours)	0731340	Computer Networks Fundamentals	0721224	(SR)
		0750335	Operating Systems	0750332	(DR)
		0750399	Practical Training	90h	(DR)
		0750362	Database Applications Programming	0731221	(DR)
		0750324	Compiler Construction	0750224	(DR)
Fourth	First (13 Credit Hours)	0750472	Modeling and Simulation	075272	(DR)
		0750497	Research Project 1	-----	(DR)
		----	Department Elective (1)	-----	(DR)
		-----	University Elective (3)	-----	(UR)
		-----	Department Elective (2)		(DR)
	Second (14 Credit Hours)	-----	University Elective (4)	-----	(UR)
		0111100	Military Sciences (Or UE Non-Jordanians Students)	----	(UR)
		----	Department Elective (3)	----	(DR)
		0750446	Information Security	0731340	(DR)
		0750498	Research project 2	0750497	(DR)

(UR) University Req.	**(UE) University Elective**	**(FR) Faculty Req.**
(DR) Dept. Req.	**(DE) Department Elective**	**(SR) Supplementary Req.**