MAKERERE UNIVERSITY

COLLEGE OF COMPUTING AND INFORMATION SCIENCES

SCHOOL OF COMPUTING AND INFORMATICS TECHNOLOGY

DEPARTMENT OF NETWORKS

P.O. BOX 7062, KAMPALA, UGANDA

BACHELOR OF SCIENCE IN SOFTWARE ENGINEERING (BSSE) DEGREE PROGRAMME

DECEMBER 2012

(DAY/ EVENING PROGRAMME)

Table of Contents

Bachelor of Science in Software Engineering (BSE)

- **Background to BSE**

This programme was first launched in August 2009. It offers a course of study leading to the B.Sc. in Software Engineering. The objectives of the B.Sc. in Software Engineering programme are: -

I. To produce graduates who are well educated in the fundamental concepts of software engineering and able to continue their professional development throughout their careers. The course combines theory with consideration of its application in software engineering practice.
II. To build human resource capacity in the Software engineering discipline in both the public and private sectors to students who wish to become proficient in developing software in a variety of languages, platforms and applications using a methodical approach.
III. To produce graduates with good communication skills capable of functioning responsibly in diverse environments and able to work in teams.
IV. To produce graduates who are innovative and are capable of creating jobs;

a. Who is a Software Engineering Graduate

A Software Engineering Graduate has an adequate grasp of the required principles and techniques to produce software systems on time, within budget and with few or no defects. Our graduate is expected to apply these principles of engineering to the design, development, maintaining, testing, and evaluation of the software and systems that make computers or anything containing software work.

Thus, the training given in our program ensures that our graduates have a personal, business and technical skill to advance their *career* wherever they want to go. In particular, we prepare them to understand that choosing an occupation, getting that first graduate job, and growing in that job will require planning, self awareness, flexibility and a forward-looking attitude to help manage the personal transitions, as well as the technological and economic changes of the future.

In that regard, our program offers diverse career options as including:

2. software developer/ engineer,
3. software architect,
4. analyst/ programmer,
5. games developer,
6. project manager,
7. mobile application developer,
8. test analyst/engineer, and
9. IT consultant.

- **Rationale for reviewing the programme**

With the technology shifts and an economy that is based on different skills, we realized a need for revising the curriculum for our programme so as to mitigate some of the obvious catastrophes and also provide the best for our graduates. We understand that for us to achieve an effective curriculum revision, it requires a thorough understanding of the processes and principles of the changing paradigms affecting curriculum development. In lieu of this, we considered various aspects in this process including:

- *Cohort:* The National Council on Higher Education, Uganda (NCHE) requires that every 3 years after the running curriculum for courses on the market in higher institutions of learning be revised. Thus, this was in fulfillment of this required as stipulated by the NCHE.
- *Market analysis:* We undertook a preliminary market analysis to evaluate the attractiveness and the dynamics of the Software Engineering Market within East Africa. Special attention was paid on the kind of jobs/roles being advertised within this market segment and the different skills set that are being sought. We compared this with the Internship feedback obtained from our industrial partners where our students get posted. With this information, we were able to identify the opportunities, strengths, weaknesses and threats of our program.
- *Industrial and Alumni feedback:* We also held a workshop with our industrial partners and Alumni to generate input into this process. In particular, key stakeholders and experts from regulatory authorities, industry and academia were invited to evaluate our current program. The feedback generated in this final phase of this process was very important in reconsidering certain content of the courses we have been offering.
- *Student sample*: A sizable sample of our students was also sampled. In here, we were testing their understanding of what we are offering against what they expected. These respondents pointed us to various areas which they considered a duplication of effort or redundancy in the program. Their feedback was thus important in helping us to eliminate duplicates and also for strengthening certain areas of our program.
- *External examiners` reports*: These reports were also a great input into this review process. Although some of the comments given were majorly on assuring quality of students' learning experiences and also the fair assessment, other comments given to improve the program were very significant in highlighting areas which were found lacking in our program.
- *QA reports:*

- **The program**

3.1 Target group

The programme targets two categories of people, namely A' level certificate holders and Diploma holders in relevant programmes.

3.2 Nature of the Programme

This is a day programme that is both government and private. There will also be the evening program that shall be only privately sponsored.

3.3. Duration

The duration for the BSc. In Software Engineering degree programme are four (4) academic years comprising 8 semesters and two recess terms.

3.4. Tuition Fees

Tuition fees for privately sponsored students shall be 3,024,000 Uganda Shillings per year for Ugandans and 4,536,000 Uganda Shillings per year for non-Ugandans.

- **Regulations**

Here we give regulations specific to the programme. Additional normal regulations that relate to illness, absence from the program conduct of examinations can be found in the undergraduate hand- book available at Academic registrars office and School of Computing and Informatics Technology.

- **Admission Requirements**

Admission to the B.Sc. in Software Engineering (B.Sc.SE) degree course will be through three avenues; Direct entry, Mature age and Diploma entry schemes. To be admitted for a course leading to the award of Bachelor of Science in Software Engineering (B.Sc.SE.) Degree, a candidate must satisfy the general minimum entrance requirements of Makerere University. In addition, the following regulations shall hold for the BSSE Degree:

- **Direct Entry**

A candidate must satisfy the general minimum entry requirements of Makerere University. In addition, the following regulations shall hold: Candidates seeking admission through this avenue must have obtained: -

- The Uganda Certificate of Education (UCE) or its equivalent, with credits in English and Mathematics.
- At least two principal passes at the same sitting in Uganda Advanced Certificate of Education (UACE) in any two subjects (Mathematics, Physics, Economics, Chemistry, Biology, Geography, Literature and Entrepreneurship).
- For purposes of computing weighted points, the advanced level subjects shall be

Group	Weight	Subjects
Essential	3	Two best done of the following subjects: Mathematics, Physics, Chemistry, Economics, Geography, Biology
Relevant	2	The third best done of the following subjects: Mathematics, Physics, Chemistry, Economics, Geography, Biology
Desirable	1	General Paper, Sub-Mathematics;
Others	½	All others.

- **Diploma Holders**

Applicants should possess at least a second class (lower division) Diploma in Computer Science, Engi- neering, Statistics or any other diploma with either Mathematics or Computer Science, as one of the subjects from any recognised Institution.

4.2 Progression

Progression shall be regarded as normal, probationary or discontinuation as per the standard Makerere University Senate guidelines.

4.2.1 Normal Progress

This occurs when a student passes each course taken with a minimum Grade Point of 2.0.

4.2.2 Probationary

This is a warning stage and occurs if either the cumulative grade point average (CGPA) is less than 2.0 and/ or the student has failed a core course. Probation is waved when these conditions cease to hold.

4.2.3. Discontinuation

When a student accumulates three consecutive probations based on the CGPA or the same core course(s), he/she shall be discontinued.

4.2.4. Retaking a Course

A Student may re-take any course when it is offered again in order to pass if the student had failed the course. A Student may take a substitute elective, where the Student does not wish to re-take a failed elective.

4.3 Weighting System

The weighting unit is the Credit Unit (CU). The Credit Unit is a contact hour per week per semester. A contact hour is equal to (i) one lecture hour, (ii) two practical hours or (iii) two tutorial hours.

4.4 Minimum Graduation Load

To qualify for the award of the degree of Bachelor of Science in Software Engineering, a candidate is required to obtain a minimum of 167 credit units for courses passed including all the compulsory courses and required number of elective courses within a period stipulated by the University Senate and Council. 140 are from core course units and at least 15 are from elective course units. A student must have at least 3 credit units from elective courses in every semester where electives are offered.

4.5. Course Assessments

i. Each Course will be assessed on the basis of 100 total marks with proportions as follows:
    1. Coursework – 40 and
    2. Examination – 60
ii. A minimum of two course assignments/tests shall be required per course.
iii. Course work shall consist of tests, group assignments and presentations in each semester.

N.B: Course work shall consist of individual tests, group assignment (these shall entail case studies, field visits and mini projects), presentations in each semester

4.6 Grading of Courses

a) Each Course will be graded out of a maximum of 100 marks and assigned an appropriate letter grade and a grade point as follows:

MARKS %	LETTER GRADE	GRADE POINT
90-100	A+	5.0
80 – 89	A	5.0
75 - 79	B+	4.5
70 - 74	B	4.0
65 - 69	C+	3.5
60 - 64	C	3.0
55 - 59	D+	2.5
50 - 54	D	2.0
45 - 49	E	1.5
40 - 44	E-	1.0
Below 40	F	0.0

b) The following additional letters will be used, where appropriate: -

W - Withdraw from Course;
I - Incomplete;
AU - Audited Course Only;
P - Pass;
F - Failure.

4.7 Minimum Pass Mark

A minimum pass grade for each course shall be 2.0 grade points.

4.8 Calculation of Cumulative Grade Point Average (CGPA)

The CGPA shall be calculated as follows:

$$CGPA = \frac{\sum_{i=1}^{n}(GP_i * CU_i)}{\sum_{i=1}^{n} CU_i}$$

Where GP_i is the Grade Point Score of a Particular course CU_i is the number of credit units of course I n is the number of courses so far done.

4.9 Knowledge Areas Covered in the Program

4.9.1 Categorization

The curriculum is based on 10 broad knowledge areas that make up practical and resourceful Information Technology specialists. These are:-

1. Software requirements
2. Software design
3. Software construction
4. Software testing
5. Software maintenance
6. Software configuration management
7. Software engineering management (Engineering management)
8. Software engineering process
9. Software engineering tools and methods
10. Software quality

4.9.2 Content Distribution by Knowledge Area

Below is a summary of the distribution of the different course units in the different knowledge areas:

- **Software requirements**
    a. BSE 3111 Requirements Engineering
    b. BIS 1206 Systems Analysis And Design
    c. BIS 1100 Foundations of Information Systems
    d. BSE 1201 Software Development Principles
    e. BIS 1204 Data and Information Management I
    f. BSE 2105 Formal Methods

- **Software design**
    a. BSE 3110 Object-oriented Analysis and Design
    b. BSE 3201 Software Architecture
    c. CSC 1104 Computer Organization and Architecture
    d. BSE 3109 Introduction to Embedded Systems
    e. BSE 3108 Communications Systems Design
    f. CSC 3110 User Interface Design
    g. BSE 4201 Software Design Patterns
    h. BSE 4202 Software Security
    i. MTH 3105 Discrete Mathematics
    j. MTH 2203 Numerical Analysis I
    k. MTH 1203 Calculus Mathematics
- **Software construction**
    a. BSE 1106 Problem Solving and Programming Concepts
    b. BSE 1202 Introduction to Internet Programming
    c. CSC 2116 Object Oriented Programming
    d. CSC 1205 Structured Programming with C
    e. CSC 2100 Data Structures and Algorithms
    f. BSE 3205 Distributed Systems Development
    g. CSC 2209 Systems Programming
    h. BSE 2205 Network Application Development
    i. BSE 3204 Advanced Object-Oriented Programming
    j. CSC 2114 Artificial Intelligence
    k. CSC 3209 Computer Graphics
- **Software testing**
    a. BSE 4101 Software Reliability and Testing
- **Software maintenance**
    a. BIT 2205 Systems Administration

b. BSE 3105 Software Evolution

- **Software configuration management**
    a. CSC 2200 Operating Systems
    b. BSE 2206 Data Communications
    c. BSE 3106 Mobile Networks and Computing
    d. BSE 2103 Computer Networks
    e. BIT 1102 Communication Technology and Internet
- **Software engineering management (Engineering management)**
    a. BSE 1301 Professional Software Engineering Mini Practical Project I
    b. BSE 2301 Professional Software Engineering Mini Practical Project II
    c. BSE 4100 Software Engineering Project I
    d. BSE 4200 Software Engineering Project II
- **Software engineering process**
    a. BIT 2207 Research Methodology
    b. BSE 3301 Internship also during Yr3 Sem I
    c. BSE 4102 Ethics for Professional Engineers
    d. BAM 2102 Business & Entrepreneurship
    e. BSE 4203 Business Law
- **Software engineering tools and methods**
    a. CSK 1101 Communication Skills
    b. CSC 1100 Computer Literacy
- **Software quality**
    a. BSE 3104 Software Metrics

- **The curriculum**

As a base for the curriculum review, we have taken a "Computing Curricula Software Engineering Volume" by Association of Computer Machinery1. We should take into consideration though that these curricula are intended to US students which are supposed to be computer-literate when entering the university. So for BSE students of the College of Computing and Information Sciences the introductory course of Computer Literacy is necessary.

5.1 Course outline

The degree programme will extend over a period of 4 years. An academic year shall consist of two semesters of 17 weeks (15 weeks for classes and 2 weeks for examinations). The first and second years will in addition have a recess term of 10 weeks. Students without the Cisco Certified Networking Associate (CCNA) certification will take CCNA as an audited course during the first year recess term. A full-time student shall not carry less than 15 credit units and not more than 21 credit units per semester. All the students must make extensive use of the computing facilities outside the scheduled lecture, tutorial and practical hours. The details of the course structure are shown below, where LH, TH, PH, CH and CU stand for Lecture Hours, Tutorial Hours, Practical Hours, Contact Hours and Credit Units respectively.

- **Year 1 Semester 1 (17 CUs)**

CODE	Name	LH	PH	TH	CH	CU	Type	Remark	MOTHER DEPT.
Core:-	**(5 Core Courses)**								
Cores									
BSE 1106	Problem Solving and Programming Concepts	30	30	--	45	3	Core	New	NW
BIS 1104	Communication Skills for IT	30	60	--	60	4	Core	Modified	IS
CSC 1100	Computer Literacy	30	60	--	60	4	Core	Old	CS
CSC 1107	Structured Programming	30	30	--	45	3	Core	Modified	CS
BIS 1100	Foundations of Information Systems	45	--	--	45	3	Core	Modified	IS
Total CU						**17**			

5.1.2 Year 1 Semester 2 (19CUs)

CODE	Name	LH	PH	TH	CH	CU	Type	Remarks	MOTHER DEPARTMENT
Core:-	**(5 Core Courses)**								
Cores									
BSE 1206	Software Development Principles	45	--	30	60	4	Core	Modified	NW
MTH 2203	Numerical Analysis I	45	--	30	45	3	Core	Old	MATH
BSE 1207	Introduction to Internet Programming	30	30	30	60	4	Core	Modified	NW
BIS 1206	Systems Analysis and Design	45	--	30	60	4	Core	Modified	IS
BIS 1204	Data and Information Management I	30	60	--	60	4	Core	New	IS
Electives: -	**(No Elective Course)**					19			
Total CU									

5.1.3 Year 1 Recess Term (4 CUs)

CODE	Name	LH	PH	TH	CH	CU	Type	Remarks	MOTHER DEPT
**BSE 1302	Professional Software Engineering Mini Practical Project I	--	60	60	60	4	Core	Modified	NW
Total CU						**4**			

5.1.4 Year 2 Semester 1 (18 CUs)

CODE	Name	LH	PH	TH	CH	CU	Type	Remarks	MOTHER DEPT
Cores: -	**(4 Core Courses)**								
Cores									
CSC 2100	Data Structures and Algorithms	45	--	30	60	4	Core	Old	CS
MTH 3105	Discrete Mathematics	30	--	30	45	3	Core	Old	MATH
CSC 2114	Artificial Intelligence	30	--	30	45	3	Core	New	CS
BSE 2106	Computer Networks	45	30	--	60	4	Core	New	NW
Electives:	**(1 Elective Course)**								
CSC 1104	Computer Organization and Architecture	60	--	--	60	4	Elective	Old	CS
BSE 2105	Formal Methods	30	30	--	45	3	Elective	Old	NW
Total CU						**18**			

5.1.5 Year 2 Semester 2 (20 CUs)

CODE	Name	LH	PH	TH	CH	CU	Type	Remarks	MOTHER DEPARTMENT
Cores: -	**(5 Core Courses)**								
Cores									
CSC 2200	Operating Systems	45	--	30	60	4	Core	New	CS
BSE 2205	Network Application Development	45	30	--	60	4	Core	Modified	NW
BSE 2206	Data Communications	45	30	--	60	4	Core	New	NW
MTH 1203	Calculus 1	45	--	30	60	4	Core	New	MATH
CSC 1214	Object Oriented Programming	30	60	--	60	4	Core	Modified	CS
Electives: -	**(No Elective Course)**								
Total CU						**20**			

5.1.6: Year 2 Recess Term (4 CUs)

CODE	Name	LH	PH	TH	CH	CU	Type	Remarks	MOTHER DEPT
**BSE 2302	Professional Software Engineering Mini Practical Project II	--	60	60	60	4	Core	Modified	NW
Total CU						4			

5.1.7 Year 3 Semester 1 (19 CUs)

CODE	Name	LH	PH	TH	CH	CU	Type	Remarks	MOTHER DEPT
Cores: -	**(4 Core Courses)**								
Cores									
BSE 3110	Object-oriented Analysis and Design	45	30	--	60	4	Core	Modified	NW
BSE 3111	Requirements Engineering	45	--	30	60	4	Core	Modified	NW
BSE 3104	Software Metrics	30	--	30	45	3	Core	Old	NW
CSC 3119	User Interface Design	45	30	--	60	4	Core	Modified	CS
Electives: -	**(At least 1 Elective Course)**								
BSE 3106	Mobile Networks and Computing	45	30	--	60	4	Elective	Old	NW
BSE 3109	Introduction to Embedded Systems	45	--	30	60	4	Elective	Modified	NW
BSE 3108	Communications Systems Design	45	30	--	60	4	Elective	New	NW
BSE 3105	Software Evolution	45	--	30	60	4	Elective	Old	NW
Total CU						**19**			

5.1.8 Year 3 Semester 2 (19 CUs)

CODE	Name	LH	PH	TH	CH	CU	Type	Remarks	MOTHER DEPT
Cores: -	**(4 Core Courses)**								
Cores									
BSE 3201	Software Architecture	45	--	30	60	4	Core	Old	NW
BSE 3208	Distributed Systems Development	45	30	--	60	4	Core	Modified	NW
BSE 3209	Advanced Object-Oriented Programming	30	60	--	45	4	Core	Modified	NW
BIT 2207	Research Methodology	30	--	30	45	3	Core	Modified	IT
Electives: -	**(At least 1 Elective Course)**								
BIT 2208	Systems Administration	45	30	--	60	4	Elective	New	IT
CSC 2209	Systems Programming	45	--	30	60	4	Elective	Old	CS
Total CU						**19**			

5.1.9 Year 3: Recess Term

CODE	Name	LH	PH	TH	CH	CU	Type	Remarks	MOTHER DEPT
**BSE 3302	Field Attachment	--	--	120	60	4	Core	Modified	NW
Total CU						**4**			

** Changed due to change in course name from Internship to Field Attachment

5.1.10 Year 4 Semester 1 (17 CUs)

CODE	Name	LH	PH	TH	CH	CU	Type	Remarks	MOTHER DEPT
Cores: -	**(4 Core Courses)**								
Cores									
BSE 4100	Software Engineering Project I	--	--	150	75	5	Core	Old	NW
BSE 4101	Software Reliability and Testing	45	--	30	60	4	Core	Old	NW
BIS 3106	Business Process Management	45	30	--	45	4	Core	Old	NW
****BAM 2102	Entrepreneurship Principles	45	30	--	60	4	Core	New	IS
Total CU						**17**			

5.1.11 Year 4 Semester 2 (16 CUs)

CODE	Name	LH	PH	TH	CH	CU	Type	Remarks	MOTHER DEPT
Cores: -	**(4 Core Courses)**								

Cores									
BSE 4200	Software Engineering Project II	--	--	150	75	5	Core	Old	NW
BSE 4201	Software Design Patterns	45	30	--	60	4	Core	Old	NW
BSE 4202	Software Security	45	30	-	60	4	Core	New	NW
BIT 2209	IT Law and Ethics	30	--	30	45	3	Core	Modified	IT
Total CU						**16**			

- **Detailed Curricular**
  - **Semester I**
    - **BSE 1106: Problem Solving and Programming Concepts**

Pre-requisites: None

a) Course Description

The course trains students from different backgrounds how to design programs. A Student is taught how to articulate thoughts about a program. The course gives design guidelines that lead students from a problem statement to a computational solution in step-by-step fashion with well-defined intermediate products.

a) Aims

Providing students with skills including critical reading, analytical thinking, creative synthesis, and attention to detail.

b) Learning outcomes

After successfully completing this course you will be able to:

Be able to read, analyze, organize, experiment, and think in a systematic manner. Write simple program algorithms, Write programs from algorithm, Learn tools for designing algorithms from human understanding to computer understanding

c) Teaching & Learning patterns

Lectures, Class discussions and quiz

d) Indicative content

- Introduction to problem solving techniques
- problem solving with Flow charts
- Algorithms and their interpretation
- Programming language classifications By abstraction level(Low level, high level, very high level), By domain (business languages, scientific languages, AI languages, systems languages, scripting languages, XML-based languages), By generality (general purpose vs. special purpose) ,By implementation methods (Interpreted vs. compiled), By paradigm (imperative, object-oriented, logic-based, functional).

- Elements of Programming Languages(Syntax , Semantics, Data).
- Programming environments (editors, compilers, linker, loader, debugger, interpreters etc)
- Program design guidelines that show the reader how to analyze a problem statement; how to formulate concise goals; how to make up examples; how to develop an outline of the solution, based on the analysis; how to finish the program; and how to test. The logical and physical structure of programs and data.

e) Assessment Method

Course work ((Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final written exam: 60%

f) Reading List

- Maureen Sprankle, Problem solving and Programming Concepts, Pearson Education, New Delhi, 7th Edition
- Compilation Notes, Department of Information Technology, SRM University
- Elizabeth A. Dickson, Computer Program Design, Tata McGraw Hill Edition, 2002
- Harold AbelsonHarold Abelson, Gerald Jay Sussman, Structure and Interpretation of Computer Programs - 2nd Edition (MIT Electrical Engineering and Computer Science)
- Matthias Felleisen, How to Design Programs: An Introduction to Programming and Computing by Matthias Felleisen
- Sally Fincher, *Studying Programming*, Palgrave MacMillan, 2006

- **CSK 1101: Communication Skills**

Pre-requisites: None

a) Course Description:
This course provides students with skills of effective communication. These include verbal, written, and gesture. The course aims at facilitating students to appropriately and clearly communicate with others.

b) Aims:
The aims of this course unit are to:
- Improve the communication competencies of the students
- Improve problem solving strategies of students
- Improve the art of critical thinking within the student
- Improve the student's ability to collect and synthesize information
- Provide students with knowledge to utilize the library and other educational resources.

c) Learning outcomes
At the end of this course, the student will be able to:
- Effectively explain to diverse audiences – orally, in writing, and through design artifacts and other information graphics – how business needs are addressed through software.
- Listen to and learn from other members of diverse teams.

d) Teaching and Learning Pattern:
Teaching and learning will be in form of class room lectures, demonstrations and students practical projects.

e) Indicative Content:
- Writing skills: Thinking critically / selectively before the writing process; selecting the relevant details logically; writing the reports, essays, letters, and taking notes in appropriate register; avoiding

ambiguities, fallacies, irrationalities; providing supportive evidence; editing documents, proof reading; writing and expanding information; quoting and citing references; writing a curriculum vitae.

- Reading skills: The use of skimming; scanning inference and prediction in reading; intensive and critical reading; acquisition of specific reading skills; interpretation of non-linear texts; locating information and comprehension.
- Speaking and listening skills to enhance effective public relations: The art of persuasion in effective speaking; conducting interviews; conducting meetings; participating in group discussions and tutorials; non-verbal communication clues; presentation seminars, seeking clarification, etc.; Expression of politeness; public speaking; proper listening skills.
- Examination skills preparing for examinations: How much one gets from group discussions; proper revision; understanding examination rubric; budgeting time during examination processes; writing examinations and following instructions.

f) Assessment Method:

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%

Final written exam: 60%

g) Reading List:

- Bennie Bough. 2005. 101 Ways to improve your communication skills instantly, 4th Edition.
- Peggy Klavs. 2008. The hard Truth About soft skills: Work place lessons smart people wish they had learned sooner
- The Communicating Organisation: Using Communication to Support the Development of High-Performing Organisations UK Department of Health, November 2009 http://www.dh.gov.uk/prod_consum_dh/groups/dh_digitalassets/documents/digitalasset/dh_110342.pdf
- Haeuser, Jamie L, Communication Strategies for Getting the Results You Want, *Healthcare Executive*, Vol. 20(1) January 2005 pp. 16-20
- Charan, Ram,Conquering a Culture of Indecision *Harvard Business Review*, Vol. 79(4) April 2001 pp. 75-82
- Hemp, Paul, Death by Information Overload, *Harvard Business Review*, Vol. 87(9) September 2009 pp. 83-89

▪ CSC 1100: Computer Literacy

Pre-requisites: None

a)Course Description:

In this course, students are to learn about the basic organization, concepts and terminologies in a computerized environment. They are also to get an in depth understanding of common computer applications. The use of related applications in different operating systems will be explored. The aims of the course unit are to: Equip students with basic knowledge about computer organization; office applications; expose students to different operating systems and equip students with skills of how to use the Internet. By the end of the course unit, the student should be able to: Describe the different parts of a computer; the historical evolution of computers; competently use the common office applications in at least two operating systems;

b) Aims

The aims of the course unit are to:

- Equip students with basic knowledge about computer organization;
- Equip students with skills of using common office applications;
- Expose students to different operating systems;
- Equip students with skills of how to use the Internet and

- Equip students with knowledge about common text editors in different operating systems.

c)Learning outcomes

By the end of the course unit, the student should be able to:

- Describe the different parts of a computer;
- Describe the historical evolution of computers;
- Competently use the common office applications in at least two operating systems;
- Competently use common text editors in at least two operating systems.

d) Teaching & Learning patterns

Teaching will be by lectures and laboratory demonstrations/practicals

e)Indicative content

- General computer organisation;
- Historical perspectives of computing;
- common Microsoft office packages;
- office packages in other operating systems;
- text editors;
- Common Linux and
- Using the web.

f) Assessment Method

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final written exam: 60%

i. Reading List

- Preston J, Ferret, R and Gaskin,S. *Computer Literacy*, 2007.
- Janrich J and Oja, D, *Practical Computer Literacy*, 2001.
- Asan, A. (2003). Computer technology awareness by elementary school teachers: A case study from Turkey. *Journal of Information Technology Education,* 2, 153-164. Available at http://jite.org/documents/Vol2/v2p153-164-109.pdf
- Bartholomew, K, Johnson, D., Ormond, P., & Mulbery, K. (2003). Computer literacy: Use IT or lose it! *Utah Valley State College School of Business Journal, 1*, 6-14.
- Bartholomew, K. (2004). Computer literacy: Is the emperor still exposed after all these years? *Journal of Computing Sciences in Colleges, 20*(1), 323-331.

- **CSC 1107: Structured Programming**

Pre-requisites: Problem Solving and Programming Concepts

- **Course Description**

The course is to create a strong base in the principles and practice of functional programming. A high level programming language like C is to be used. Students are to cover both theoretical principles and hands on practical skills. The main concepts to cover include program structure, data structures, syntactical and semantic correctness, planning and segmentation in programming as well as working with files

- **Aims**

Comprehensive knowledge about structured oriented programming; Knowledge in planning and organization of programming projects; Knowledge and techniques of evaluating syntactic and semantic correctness of a computer program; Strong practical basis in programming

- **Learning outcomes**

After successfully completing this course you will be able to:

- Solve problems using the C language.
- Write applications in C.
- Use modular decomposition to reduce the complexity of problems.
- Construct well structured C language programs.
- Test and validate software solutions.

- **Teaching & Learning patterns**

The course will be taught with a big practical component. Students will be expected to have at least one supervised practical session per week. They will also be given several programming assignments some of which will be marked and contribute to the coursework scores. Assignments given shall be based on real life problems to enable students make a connection between what is taught and what is in practice

- **Indicative content**

- Program structure
- Variables and Operators
- Conditional statements
- Looping statements
- Arrays and Strings
- Functions
- Advanced data types
- Pointers, dynamic memory allocation and dynamic structures
- Working with files

- **Assessment**

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%

Final examination (60%)

- **Reading List**

- Brian W. Kernighan, Dennis M. Ritchie, C Programming Language ; Prentice Hall, 2000.

- Samuel P. Harbison, C: A Reference Manual (5th Edition); Prentice Hall, 2002.
- Behrouz A. Forouzan, Richard F. Gilberg, Structured Programming with C, 3ed.2006, ISBN-10: **0534491324**
- Nell Dale , Chip Weems, Programming and Problem Solving with C++, Jones & Bartlett Learning; 5 edition (May 15 2009)
- Stephen G. Kochan, Programming in Objective-C (5th Edition), Addison-Wesley Professional; 5 edition (Dec 4 2012)

- **BIS 1100: Foundations of Information Systems (3 CU)**

Pre-requisites: None

- **Course Description:**

Information systems are an integral part of all business activities and careers. This course is designed to introduce students to contemporary information systems and demonstrate how these systems are used throughout global organizations. The focus of this course will be on the key components of information systems –people, software, hardware, data, and communication technologies, and how these can be integrated and managed to create competitive advantage. Through the knowledge of how IS provides a competitive advantage students will gain an understanding of how information is used in organizations and how IT enables improvement in quality, speed, and agility. This course also provides an introduction to systems and development concepts, technology acquisition, and various types of application software that have become prevalent or are emerging in modern organizations and society.

- **Aims:**

These include, to:

a. make students understand how and why information systems are used today
b. introduce students to the technology, people, and organizational components of information systems
c. make students understand globalization and the role information systems have played in this evolution
d. make students understand how businesses are using information systems for competitive advantages vs. competitive necessity
e. make students understand the value of information systems investments as well as to learn to formulate a business case for a new information system, including estimation of both costs and benefits

- **Learning outcomes:**

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

- explain correctly how and why information systems are used today
- describe the technology, people, and organizational components of information systems
- describe globalization and the role information systems have played in this evolution
- explain correctly how businesses are using information systems for competitive advantage vs. Competitive necessity
- describe the value of information systems investments as well as formulate a business case for a new information system, including estimation of both costs and benefits

- **Teaching and Learning patterns:**

Teaching will be in terms of lectures, problem–based/case studies, group work, and presentations.

- **Indicative content:**

- Characteristics of the Digital World
- Information systems components
- Information systems in organizations
- Globalization
- Valuing information systems
- Information systems infrastructure
- The Internet and WWW
- Security of information systems
- Business intelligence
- Enterprise-wide information systems
- Development and acquisition
- Information systems ethics and crime

- **Assessment:**

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

- **Reading List:**

1. Ralph Stair and George Reynolds. 2013. *Fundamentals of Information Systems*, 7th Edition. Publisher: Cengage South-Western. ISBN 9781133629627.
2. Ralph M Stair, George Reynolds, Ralph Stair, George Walter Reynolds. 2009. *Principles of Information Systems: A Managerial Approach*, 9th Edition. Publisher: Cengage Learning. ISBN 9780324665284.
3. Vladimir Zwass, 1997. *Foundations of Information Systems*. Publisher: McGraw Hill. ISBN: 9780071156387.
4. Joseph Valacich and Christoph Schneider, 2012. *Information Systems Today*, 5th Edition. Publisher: Prentice Hall. ISBN-10: 0137066996, ISBN-13: 9780137066995.
5. Steven Alter, Information systems: foundation of e-business, Prentice Hall, 2002

- **Year 1 Semester II**
  - **BSE 1206: Software Development Principles**

Pre-requisites:

Programming 1, Programming 2, Programming Techniques, Software Engineering Fundamentals
Minimum CGPA 3.0

- **Course Description**

This course introduces students to the basic software Development Principles & Processes highlighting the development of software systems with a focus on leading software development and management processes and practices. This is design to orient a student a Systems Engineering perspective, that is, an interdisciplinary, collaborative approach to the engineering of system solutions which aims to capture

stakeholder needs and objectives and to transform these into a holistic, life-cycle balanced system solution which both satisfies the minimum requirements of the stakeholders, and optimises overall solution effectiveness according to the values of the stakeholders.

- **Aims**

Having studied this course it is intended that the student should be able to:

- demonstrate written and spoken communication skills through the presentation of reports and workshops;
- offer value judgments on technical, ethical and professional issues that are relevant to the work undertaken;
- reflect on your experiences and:
  - relate them to your existing knowledge, understanding and attitudes;
  - use them to synthesize new knowledge, understanding and attitudes, and
  - use them to develop a longer-term perspective on your future career development.

- **Learning outcomes**

At the conclusion of this course the student will:

a) be better prepared to specify, plan, develop, deliver, maintain and operate a software intensive system. Given the constant evolution of software development methodologies, delegates who are already experienced software professionals will further develop their skills.

b) become an advocate in your organization of practical methods to improve software project performance.

c) identify causes of software development problems and drive project performance inprovements.

d) value the substantial body of public domain knowledge that defines world's leading practice in software engineering. For example, ISO/IEC/IEEE 12207, other IEEE and ISO standards, the Software Engineering Institute (SEI) Capability Maturity Models and the Guide to the Software Engineering Body of Knowledge (SWEBOK).

- **Teaching and Learning patterns:**

Teaching will be in terms of lectures, problem–based/case studies, group work, and presentations.

- **Indicative content**

- Introduction
  - Why is Systematic Software Development necessary?
- Phases of System Development Lifecycle
- System development approaches
- System development methodologies or models
- Software Systems Engineering Process Frameworks
- Agile Methods and Techniques
- Software Design & construction
- System Integration
- Quality Management
- Technical reviews
- Other verification and validation approaches and methods
- Project Management Frameworks and activities

- **Assessment**

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

- **Reading List**

- Roger S. Pressman, *Software Engineering: A Practicioners' Approach* (SEPA), seventh edition, McGraw-Hill (required).
- Fredrick Brooks, *The Mythical Man-Month* (MMM), Anniversary edition, Addison-Wesley (required).
- Kernigham and Ritchie, *The C Programming Language*.
- Davis, A. M. 201 *Principles* of *Software Development*. (New York, NY: McGraw-Hill, Inc., 1995.)
- McLaughlin, Brett D., et. al., *Head First Object-Oriented Analysis & Design*. (Sebastopol, CA: O'Reilly Media, Inc., 2007.)

- **MTH 2203: Numerical Analysis I**

Pre-requisites: None

a) Course Description:

Upon completion of the course, the student should be able to: (i) Demonstrate factual knowledge including the mathematical notation and terminology used in the course; (ii) Describe the fundamental principles including the laws and theorems arising from the concepts covered in this course; (iii) Apply course material along with techniques and procedures covered in this course to solve practical problems; and (iv) Write numerical programs, such as Matlab programs, to solve the above problems

b) Aims

Numerical linear algebra, numerical solution of systems of non-linear equations, approximations, Fast Fourier-Transformation, numerical integration, difference equations and numerical solution of ordinary differential equations. Problem solving is an important part of the course.

c) Learning outcomes

By the end of the course, the student should be able to

- Derive schemes for different set ups of numerical problems
- Test for convergence of different schemes
- Correctly use the schemes to generate solutions to the numerical problems to the required precision

d) Teaching & Learning patterns

The course will be taught theoretically. The developed schemes can be implemented and run in any programming language.

e) Indicative content

a) Errors and their propagations
b) Lagrange's Interpolating Polynomial
c) Iterated interpolation
d) Finite difference interpolating polynomials
e) Numerical Differentiation
f) Numerical Integration
g) Adoptive iteration
h) Solutions to non- linear equations
i) Systems of non -linear equations
j) Systems of linear equations

f) Assessment

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

g) Reading List

- Burden, R.L. and Faires, D.J , Numerical Analysis,, Eighth Edition,Brooks/Cole Publishing Co.Pacific Grove, CA, 2001. ISBN: 0-534-38216-9

- M.J. Overton, *Numerical Computing and the IEEE Floating Point Standard*, SIAM, 2001
- H.C. Elman, D.J. Silvester, A.J. Wathen, *Finite Elements And Fast Iterative Solvers,* Oxford University Press, 2005
- H. Wendland: *Scattered Data Approximation*, Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, Cambridge, 2005.
- J. Nocedal and S.J. Wright: *Numerical Optimization,* 2nd ed., Springer, 2006.

- **BSE 1207: Introduction to Internet Programming**

Pre-requisites: None

a) **Course Description**

This course introduces the student to Internet and Web technologies. The student is equipped with Web development techniques for adding useful, interactive and dynamic elements to a website.

b) **Course aim and objectives**

To introduce the student to Web based applications development

c) **Learning outcomes**

At the end of this course, the student should;

a) Identify the main concepts in the web development environment
b) Create web pages using HTML and CSS
c) Add interactivity to web pages using JavaScript/Ajax
d) Create and manipulate XML documents
e) Identify alternative web development technologies

d) **Teaching & Learning patterns**

In this course a combination of lectures and practicals will be used

e) **Indicative content**

- Introduction to the Internet and Web
- Web page design with HTML and CSS
- Introduction to Web 2.0 features and any other related latest technologies
- Client-side programming with JavaScript and Ajax
- Introduction to XML and manipulation of XML documents
- Other web development Technologies

f) **Assessment**

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)
.

g) **Reading List**

- Paul J. Deitel and Harvey M. Deitel, Internet & World Wide Web: How to Program, ISBN 10:0131752421, Prentice Hall; 4 edition September, 2007.
- Steven M. Schafer (2005), HTML, CSS, JavaScript® , Perl, Python®, and PHP Web Standards Programmer's Reference, ISBN: 81-265-0620-2, Wiley Publishing Inc., USA (Indian Edition)
- Craig D. Knuckles, Introduction to Interactive Programming on the Internet: Using HTML & JavaScript, ISBN-10: 047138366X, Wiley; 1 edition (November 23, 2000)
- Thomas Powell, HTML & XHTML: The Complete Reference (Osborne Complete Reference Series), McGraw-Hill Osborne Media; 4 edition (August 19, 2003)
- Jennifer Niederst,Web Design in a Nutshell, O'Reilly Media, Inc.; 2nd edition (October 15, 2001)

- **BIS 1206: Systems Analysis And Design**

Pre-requisites: None

1) **Course Description:**
   This course discusses the processes, methods, techniques and tools that organizations use to determine how they should conduct their business; with a particular focus on how computer-based technologies can most effectively contribute to the way business is organized. The course covers a systematic methodology for analyzing a business problem or opportunity, determining what role, if any, computer based technologies can play in addressing the business need, articulating business requirements for the technology solution, specifying alternative approaches to acquiring the technology capabilities needed to address business requirements, and specifying the requirements for the information systems solution in particular, in-house development, development from third-party providers, or purchased commercial-off-the-shelf (COTS) packages.
2) **Aims**
   To:
   a. Make students understand the types of business needs that can be addressed using information technology based solutions.
   b. Equip students with the skills of initiating, specifying, and prioritizing information systems projects and for determining various aspects of feasibility of these projects.
   c. Equip students with the skills of writing clear and concise business requirements documents that can be converted into technical specifications within the context of the methodologies they learn
   d. Equip students with skills for: communicating effectively with various organizational stakeholders, collecting information using various techniques, and conveying proposed solution characteristics to the stakeholders
   e. Introduce students to various systems acquisition alternatives, including packaged systems (such as ERP, CRM, SCM, etc.) and outsourced design and development resources.

3) **Learning outcomes:**
   At the end of this course, students should be able to:
   - Describe the types of business needs that can be addressed using information technology-based solutions
   - Initiate, specify, and prioritize information systems projects and determine various aspects of feasibility of these projects
   - Clearly define problems, opportunities, or mandates that initiate projects
   - Use at least one specific methodology for analyzing a business situation (a problem or opportunity), modeling it using a formal technique, and specifying requirements for a system that enables a productive change in a way the business is conducted
   - Write clear and concise business requirements documents and convert them into technical specifications within the methodologies that they learn.
   - Communicate effectively with various organizational stakeholders to collect information using various techniques and to convey proposed solution characteristics to them.

4) **Teaching and learning patterns:**
   - Lectures
   - Project-like assignments to be done in groups
   - Class discussions about the project-like assignments

5) **Indicative content:**

- Identification of opportunities for IT-enabled organizational change
- Business process management
- Analysis of business requirements
- Structuring of IT-based opportunities into projects
- Project specialization
- Project prioritization
- Analysis of project feasibility
- Fundamentals of IS project management in the global context
- Using globally distributed communication and collaboration platforms
- Analysis and specification of system requirements
- Different approaches to implementing information systems to support business requirements
- Specifying implementation alternatives for a specific system
- Impact of implementation alternatives on system requirements specification
- Methods for comparing systems implementation approaches
- Organizational implementation of a new information system
- Different approaches to systems analysis and design: structured SDLC, unified process/UML, agile methods

6) Assessment:

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

7) Reading List

- Gary B. Shelly and Harry J. Rosenblatt. 2012. *Systems Analysis and Design* (9th Edition). Publisher: Course Technology, Cengage Learning, Boston. ISBN 9780538481618.
- I.T. Hawryszkiewycz. 2001. *Introduction to Systems Analysis and Design* (5th Edition). Publisher: Prentice Hall. ISBN-10: 1740092805 ISBN-13: 9781740092807.
- Shouhong Wang and Hai Wang. 2012. *Information Systems Analysis and Design*. Publisher: Universal-Publishers
- Jeffrey Whitten , Lonnie Bentley Systems Analysis and Design Methods, McGraw-Hill/Irwin; 7th edition (November 22, 2005)
- Alan Dennis, Systems Analysis and Design, Wiley; 5 edition (January 18, 2012)

- **BIS 1204 Data and Information Management I (4 CU)**

Pre-requisites: Foundations of Information Systems

- **Course Description:**

This course provides the students with an introduction to the core concepts in data and information management. It is centered around the core skills of identifying organizational information requirements, modeling them using conceptual data modeling techniques, converting the conceptual data models into relational data models and verifying its structural characteristics with normalization techniques, and implementing and utilizing a relational database using an industrial-strength database management system.

- **Aims:**

The aims of this course are to:

- Provide the students with systematic approaches to the design and implementation of database

applications

- Give hands on experience and knowledge in developing database (driven) applications

- **Learning outcomes**

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

- Understand the role of databases and database management systems in managing organizational data and information.
- Understand the historical development of database management systems and logical data models.
- Understand the basics of how data is physically stored and accessed.
- Understand the fundamentals of the basic file organization techniques.
- Apply information requirements specification processes in the broader systems analysis & design context.
- Use at least one conceptual data modeling technique (such as entity-relationship modeling) to capture the information requirements for an enterprise domain.
- Link to each other the results of data/information modeling and process modeling
- Design high-quality relational databases.
- Understand the purpose and principles of normalizing a relational database structure. Design a relational database so that it is at least in 3NF.
- Implement a relational database design using an industrial-strength database management system, including the principles of data type selection and indexing.
- Use the data definition, data manipulation, and data control language components of
- SQL in the context of one widely used implementation of the language.
- Understand the basic mechanisms for accessing relational databases from various types of application development environments.
- Understand the role of databases and database management systems in the context of enterprise systems.

- **Teaching and learning patterns**

This course will be delivered through lectures, tutorials and laboratory Practicals. Students will be offered a range of experiences that include:

a) Large group lectures
b) Working with team members in small groups (small group tutorials)
c) Take home assignments
d) Oral presentations
e) Self-assessment and peer assessment
f) Electronic discussion forum.

- **Indicative content:**

This course will in the preliminary cover operations like requirements gathering and database planning. The course will also introduce students to developing application programs that talk to a database. These applications may be online or offline. The course will also include the following topics:

- Database approach
- Types of database management systems
- Basic file processing concepts and physical data storage concepts including file organizations techniques
- Database Development Life Cycle
- Conceptual data model
  - Entity-relationship model

- Object-oriented data model
- Specific modeling grammars

- Logical Data Model
- Hierarchical data model
- Network data model
- Relational data model
  - Relations and relational structures
  - Relational database design
  - Mapping conceptual schema to a relational schema
  - Normalization
- Physical data model
  - Indexing
  - Data types
- Database Application Development
    - Database Languages
  - SQL: DDL, DML, and DCL
- Scripting
- Interface Development

- **Assessment:**

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

Reading List

1. Thomas Connolly and Carolyn Begg. 2003. *Database Solutions: A Step by Step Guide to Building Databases* (2nd Edition). Publisher: Pearson Addison Wesley. ISBN 0321173503.
2. Thomas Connolly and Carolyn Begg. 2002. *Database Systems: A Practical Approach to Design, Implementation, and Management* (3rd Edition). Publisher: Pearson Addison Wesley.
3. Silberschatz A., Korth H. F. and Sudarshan. 2002. *Database Systems Concepts* (4th Edition). Publisher:McGraw-Hill.
4. Greg R. 2001. *Principles of Database Systems with Internet and Java Application*. Publisher: Addison Wesley.
5. Len Silverston, The Data Model Resource Book, Vol. 1: A Library of Universal Data Models for All Enterprises, Wiley; Revised Edition, Volume 1 edition (March 6, 2001)

- **Year 1 Recess Term**
  - **BSE 1302: Professional Software Engineering Mini Practical Project I**

Pre-requisites: This course assumes the students have covered introductory C and web programming courses

- **Course Description**

This course provides students with a holistic hands-on experience in building software products in the C and web programming platforms in a team environment

- **Aims**

These are, to:

a) equip students with practical skills in software engineering from requirements solicitation to testing;
b) provide students with mentorship in writing software products in C and web programming environments;
c) give students an opportunity to build team-work and leadership skills needed in the workforce.

- **Learning outcomes**

Upon successful completion of the course, the student should:

1) Demonstrate mastery of a specific (C and web) programing language and development platform
2) Carry out requirement engineering for a particular project; and
3) Demonstrate ability to develop a mini application or systems with minimal difficulty.

- **Teaching & Learning patterns**

Lectures, practicals, group work, & tutorials

- **Indicative content**

An introduction to a specific professional software certification course is covered.

- **Assessment**

Assignment, Tests and Exam

- **Reading List**
- Andrew Koenig and Barbara E. Moo, Accelerated C++: Practical Programming by Example, Addison-Wesley, January 2008, ISBN 020170353X
- Jeff Friesen Beginning Java SE 6 Platform: From Novice to Professional, , Apress October 2007, 159059830X
- Robert Freeman Easy Oracle Jumpstart: Oracle Database Management Concepts and Administration, Steve Karam, and, ISBN 0-9759135-5-7
- Steven Feuerstein, Oracle PL/SQL Language Pocket Reference, Bill Pribyl and Chip Dawes, Fourth Edition, October 2007, OReilly, ISBN 10: 0-596-51404-2

- **Year 2 Semester I**

- **CSC 2100: Data Structures and Algorithms - update from BSc. CS curriculum**

Pre-requisites: None

- **Course Description**

The course gives students a firm foundation of data structures and algorithms. The course trains students on systematic development and analysis of algorithms. The importance of algorithm complexity on computer performance is emphasized. Typical computational problems and their solutions/analysis are to be covered.

- **Aims**

The aims of the course are to

- Make students appreciate the role of data structures and algorithms in computer programs;
- Improve students' problem solving skills by subjecting them to step by step analysis and design of computer algorithms;
- Introduce students to concepts Data structures;
- Introduce students to concepts of algorithm analysis;
- To expose students generic algorithmic problems and apply them to other computational

scenarios.

- **Learning outcomes**

At the end of the course the student will be able to

- Describe the usage of various data structures,
- Explain the operations for maintaining common data structures,
- Recognize the associated algorithms' operations and complexity
- Design and apply appropriate data structures for solving computing problems
- Develop computer programs to implement different data structures and related algorithms
- Possess the ability to design simple algorithms for solving computing problems
- Think critically
- Solve problems independently

- **Teaching & Learning patterns**

The teaching pattern is by lecture, practical lab work, group discussion and class presentations.

- **Indicative content**

- Complexity analysis (Big-O notation, orders of growth, worst case, average case and amortized analysis);
- NP-complete problems;
- Greedy algorithms;
- Dynamic programming;
- Design patterns for data structures;
- Graphical representation of optimization problems
- Parallel algorithms.
- Sorting and searching;
- Divide-and-conquer algorithms;
- Elementary data structures;
- Recursive data structures (stacks, queues, linked lists, trees);
- Storing and searching (hash tables, search trees);
- Graph algorithms on graphs (shortest path, spanning trees).

- **Assessment method**

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

- **Reading List**

- Alfred V. Aho, Jeffrey D. Ullman, John E. Hopcroft, Data Structures and Algorithms. Addison-Wesley, 1983
- Alfred V. Aho, The Design and Analysis of Computer Algorithms, Addison-Wesley Longman, 1974
- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein,,Introduction to Algorithms 2nd Ed. McGraw-Hill, 2008.
- C.Xavier and S.S. Iyengar, Introduction to parallel algorithms, John Wiley & Sons, 1998
- Richard Bellman, Dynamic Programming, Dover Publications; Reprint edition (March 4, 2003)

- **MTH 3105: Discrete Mathematics**

Pre-requisites: None

a) **Course Description**

The course applies mathematics to finite or discontinuous quantities in order to master the process of problem- solving, communication, reasoning, and modeling. It gives a basic understanding of mathematical structures that are fundamentally discrete. Objects studied in discrete mathematics are largely countable sets such as integers, finite graphs, and formal languages. Applications of such concepts to computer science are to be studied. Concepts and notations from discrete mathematics are useful in studying and describing objects and problems in computer algorithms and programming languages.

b) Aims

The aim of this course is to provide the student with: -

- A basic understanding of mathematical objects that assume only distinct, separate values, rather than values on a continuum.
- The main ideas studied in the broad area of Discrete Mathematics especially clear algorithmic aspects.
- An understanding of what the relation between problems is.

c) Learning outcomes

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

- familiar with the terminology, operations, and symbols of set theory, and with formal logic.
- able to use logic to determine the validity of an argument.
- able to construct the proof of a theorem directly, by the contra positive, by cases, by contradiction, truth table, by counter-example, and by mathematical induction.
- able to identify a relation; specifically, a partial order, equivalence relation, or total 16 order.
- able to identify a function; specifically, subjective, injective, and bijective functions.

d) Teaching and Learning pattern

- Teaching and learning will be by lectures and tutorials

e) Indicative content

- Logics and set theory
- Number theory
- Relations and functions
- Languages
- Finite state machines (and optionally Finite state automata)
- Groups and Modulo Arithmetic
- Number of solutions of a linear equation
- Recurrence relations
- Searching algorithms

1. Assessment:

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

2. Reading list

- Bobrow, L.S. and Arbib, M.A. Discrete Mathematics: Applied Algebra for Computer and Infor-mation Science. Philadelphia, PA: Saunders, 1974.
- Dossey, J.A.; Otto, AD.; Spence, L.; and Eynden, C.V. Discrete Mathematics, 3rd ed. Reading, MA: Addison-Wesley, 1997.
- Balakrishnan, V.K. Introductory Discrete Mathematics. New York: Dover, 1997.
- Rosen K.H. *Discrete Mathematics and Its Applications* (third edition, McGraw-Hill, 2007).
- Graham A Stephen, String Searching Algorithms, Lecture Notes Series on Computing – Vol 6, World Scientific, 1994

- **CSC 2114: Artificial Intelligence**

Pre-requisites: None

- **Course Description**

This course examines the concepts, techniques, applications, and theories of Artificial Intelligence. The focus of the course is on the theory and application of artificial intelligence. Topics include logic, search, and reasoning with an emphasis on fundamentals and recent advances in AI. Given the broad range of topics addressed by the AI field, topics for discussion must, necessarily, be limited. Therefore, this course will focus on issues of search, knowledge representation, reasoning, decision making, and learning from the perspective of an intelligent agent.

- **Aims**

- To give students an advanced understanding of, and competence with, the theories, concepts, technologies and techniques of Computer Games development.
- To produce graduates possessing awareness, knowledge and practical skills in the field of Computer Games enabling them to follow a programme of study that will offer relevant specialization and career options.
- To develop students professional attitudes, interpersonal and entrepreneurial skills which are required by a practitioner in the industry.
- To provide students with critical and evaluative perspectives related to Computer Games development and develop students capacity for independent and self-reflective learning, ensuring their future contribution to research and development.

- **Learning outcomes**

At the end of this course, the student should;

- Formulate and assess problems in artificial intelligence.
- Assess the strengths and weaknesses of several methods for representing knowledge.
- Assess the strengths and weaknesses of several AI algorithms in areas such as heuristic search, game search, logical inference, statistical inference, decision theory, planning, machine learning, neural networks, and natural language processing.
- Implement software solutions to a wide-variety of problems generally considered to require artificial intelligence.

- **Teaching & Learning patterns**

In this course a combination of lectures and practicals will be used

- **Indicative content**

- Introduction to Artificial Intelligence: Simulation of Intelligence behavior, in different areas;
- Problem solving: games, natural language question answering, visual perception, learning; Aim oriented (heuristic) algorithms versus solution-guaranteed algorithms.
- Understanding Natural Languages: Parsing techniques, context-free and transformational grammars, transition nets, augmented transition nets, grammar-free analyzers, sentence generation.
- Knowledge Representation: First-order predicate calculus; PROLOG and LISP languages; Semantic nets; partitioned nets; Production rules; knowledge base, the inference system, forward and backward deduction.
- Expert System: Existing systems (DENDRAL, MYCIN), Domain exploration; Meta-knowledge, expertise transfer, self-explaining systems.
- Pattern Recognition Structured Descriptions: Symbolic description, machine perception, line finding, interpretation, semantics and models, object identification and speech recognition.

- **Assessment**

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group

projects – 20%): 40%
Final examination (60%)

- **Reading List**

- R. Duda, P.Hart, Pattern Classification and Scene Analysis, Wiley, 1973.
- E.A. Feigenbaum, J.Feldman; Computers and Thoughts, AAAI Press, 1995.
- N.J. Jilsson, Problem Solving Methods in Artificial Intelligence, McGrawHill, 1971.
- J.Lloyd, Foundation of Logic Programming, Springer-Verlag, 1993.
- Stuart Russell and Peter Norvig Artificial Intelligence: A Modern Approach (3rd Edition), Prentice Hall; 3 edition (December 11, 2009)

- **BSE 2106: Computer Networks**

Pre-requisites: None

a) Course Description

This is an introductory course in computer networks This course provides a broad overview of computer networking, covering application layer, transport layer, network layer, and link layers. It covers basic concepts in computer networking as well as the prominent Internet protocols. It also ensures that students practically appreciate them.

b) Aims

The course objectives are, to:

- Familiarise students with the terminology and concepts of the OSI reference model and the TCP/IP reference model protocols
- Explore performance issues in local area networks and wide area networks, and wireless networks;
- Give students knowledge and skills in network tools, practical network set up and network trouble shooting

c) Learning outcomes

Upon successful completion of this course students should be able to:

- Master the terminology and concepts of the OSI reference model and the TCP/IP reference model protocols;
- Appreciate performance issues in local area networks and wide area networks, and wireless networks;
- Demonstrate knowledge of network tools.
- Practically set up a network, trouble shoot

d) Teaching & Learning patterns

This course will be delivered through class lectures, discussions and laboratory sessions. In addition, students shall be assigned with practical project in which they shall demonstrate practical knowledge of network setup, configuration and troubleshooting.

e) Indicative content

a) Introduction to Networks: definition, advantages, types, configurations; network topologies, types of networks

b) The OSI/ISO reference model; Transmission media: magnetic media, twisted pair, coaxial, fiber-optics;

c) Packet switching, delay and loss concepts, physical media, protocol layering,

d) Application layer: Web, E-mail, DNS, FTP,

e) Transport layer: how UDP and TCP,

f) Network layer and routing routing protocol basics, RIP , IPv4 and IPv6, addressing and CIDR,

g) Computer Network security basics

f) Assessment Method

- (Assignments, Tests): 20%

- Group practical project 20%
- Final Examination 60%

g) Reading List

- James F. Kurose and Keith W. Ross. Computer Networking - A Top Down Approach Featuring the Internet, 5th edition, Addison-Wesley, ISBN 0-321-22735-2.
- Andrew S. Tanenbaum Computer Network (4th Edition) Prentice Hall
- W. Richard Stevens, **TCP/IP Illustrated, Vol. 1: The Protocols,** Addison-Wesley Professional; US ed edition (December 31, 1993)
- Eric Rescorla , **SSL and TLS: Designing and Building Secure Systems,** Addison-Wesley Professional; 1 edition (October 27, 2000)

- **CSC 1104 Computer Organization and Architecture**

Pre-requisites: None

- **Course Description**

This course introduces data representation and basic digital circuits. It highlights the lower end operations. The course opens up a student to be an informed user of the computer rather than a passive recipient of the computer services.

- **Aims**

The aims of the course are:

- To highlight to students the way computers process and store the data;
- To highlight internal management issues in computer systems.

- **Learning outcomes**

By the end of the course, the student will be able to know processing and storage mechanisms of computer systems.

- **Teaching & Learning patterns**

Teaching will be in terms of lectures as well as tutorials

- **Indicative content**

- Data Representation: Integer Formats, Binary, Octal and Hexadecimal Systems, Negative integers and 2's Complement, Floating Point Formats, BCD Formats, Alphanumeric Codes.
- Basic Digital Circuits: Logic gates, Karnaugh maps, Combinatorial Circuits, Binary Adders, Multiplexers and Demultiplexers, Comparators, Decoders and Encoders, Code Converters, ROMS and PLA's, Sequential Circuits, Flip Flops and Latches, R-S flip flops, J-K flip flops, T flip Flops, D flip flops, Registers, Shift Registers and Data Transmission, Sequential Network Design.

- **Assessment Method**

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

- **Reading List**
  - M. Morris Mano, Computer Systems Architecture, Prentice Hall, 1993
  - Glenn B. Gibson, Computer Systems Concepts and Design, Prentice Hall, 1991
  - Linda Null and Julia Lobur, Essentials of Computer Organization and Architecture, Jones & Bartlett Pub; 3 Int Stu edition (December 9, 2010

- William Stallings, Computer Organization And Architecture: Designing For Performance, Prentice Hall, 2006
- David A. Patterson, John L. Hennessy, Computer Organization and Design, Fourth Edition: The Hardware/Software Interface, Morgan Kaufmann, 2008

BSE 2105: Formal Methods

Pre-requisites: Discrete Mathematics

Course Description

The course introduces students to application of mathematical techniques to reason and think about computations.

Course Aims and Objectives

Upon completion of the course, students will be able to

- have an appreciation of the professional need to establish formal properties of software;
- have a belief that formal specifications can improve the quality of software.

Learning Outcomes

- use the Event-B notation to develop and prove software specifications;
- install a Event-B Toolkit on a Unix/Linux/Windows platform;
- write basic Event-B specifications;
- Refine and extend more advanced Event-B specifications.

Teaching and Learning Patterns

Lecture notes, student presentations, short exploratory assignments

Indicative Content

Topics to be covered include:

- Review of set theory,
- the predicate calculus,
- relations,
- relational algebra and formal specification concepts;
- algebraic and model based specifications;
- the role of formal specifications in software engineering.
- The Event-B notation,
- data and algorithm design;
- data and operation refinement;
- proofs of correctness; proof obligations.

Assessment Method

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

Reading List

- Jean-Raymond Abrial, The B-Book: Assigning Programs Meanings, , Cambridge University Press, 1996, ISBN 0-521-49619-5
- John Wordworth Software Engineering with B, Addison Wesley Longman, 1996, ISBN 0-201-40356-0
- Kevin Lano Specifications in B: An Introduction using B Toolkit, , World Scientific Publishing Company, Imperial College Press, ISBN 1-86094-008-0
- Jean François Monin, Michael Gerard Hinchey, Understanding Formal Methods, Springer, 2003

- Prof. Jonathan Bowen, Centre for Applied Formal Methods, London South Bank University, Formal Specification and Documentation using Z: A Case Study Approach, Thomson Publishing , 2003

Year 2 Semester II

CSC 2200: Operating Systems

Pre-requisites: None

- **Course Description**

The course covers organization of multi-user and multi-tasking computers. The principles of operating systems are covered with reference to the underlying hardware requirements and are illustrated by case studies.

- **Aims**

By the end of this course (i) Students will understand the various levels of system and application software; (ii) They will be familiar with the major Operating System services such as file systems, memory management, process management, device control and network services; (iii) They will understand how design decisions in Operating Systems affect users of the system; (iv) In addition, students will have used a major Operating System extensively, with experience in using an interactive command line programming language; and (v) They will also will have experience in using a systems programming language with an Application Programmers Interface to the Operating System for its services based on Unix OS, and the C systems programming language.

- **Learning outcomes**

A knowledge and understanding of

- Program OS components, such as job and process schedulers, page replacement algorithms, and file management subsystems, as well as programming interrupt handlers and contact switching

- **Teaching and Learning Patterns**

Lecture notes, student presentations, short exploratory assignments

- **Indicative content:**

a) Topics include: operating system structure and services, multi-programming processes, CPU scheduling. Memory management, synchronization, deadlocks, virtual memory and file systems

b) This unit looks at the necessary system architecture introduction for further study of operating systems, computer architectures, and the implementation of higher level languages.

c) It goes further and builds upon that by looking at the concepts underlying Operating Systems, and to show how different choices in Operating System design and implementation have effects on applications, application programmers and user environments.

- **Assessment Method**

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%

Final examination (60%)

- **Reading List**

- Leland L. Beck, System Software: An Introduction to Systems Programming , , Addison Wesley; 3rd edition, August, 1996, ISBN-10: 0201423006
- Andrew S. Tanenbaum Modern Operating Systems,, 2nd Edition , Prentice Hall 2001, ISBN 0130926418
- Abraham Silberschatz, Peter Baer Galvin and Greg Gagne, Operating Systems Concepts, , 6th Edition, John Wiley & Sons 2002, ISBN 0471250600
- D. R. Engler, M. F. Kaashoek, and J. O'Toole Jr. Exokernel: An Operating System Architecture for Application-Level Resource Management. [ps] *Proc. 15th SOSP*, December 1995, pp. 251-266.

- M. Frans Kaashoek, Dawson R. Engler, Gregory R. Ganger, Hector M. Briceno, Russell Hunt, David Mazieres, Thomas Pinckney, Robert Grimm, John Jannotti, and Kenneth Mackenzie. Application Performance and Flexibility on Exokernel Systems. Proc. of the 16th ACM. Symposium on Operating Systems Principles, October 1997.

▪ BSE 2205: Network Application Development

Pre-requisites: Java programming, web fundamentals (HTML and Javascript), XML

- **Course Description**

The course introduces students to the development of enterprise web applications using servlets and JSP. Students learn how to design, build, test and configure web applications using java Servlets and JSP technology

- **Aims**

- To understand the HTTP-client /server design in web projects
- To describe the architecture of java servlets
- To deal with servlet containers and understand their architecture and how they work
- To master the basic concepts about java servlets
- To enable trainees to appreciate the difference between servlets and JSP
- To enable trainees to work with java beans from JSP
- To enable students set up web servers, and web projects using java

- **Learning outcomes**

a) Understand the HTTP-Client server model
b) Describe the architecture of java servlets
c) To deal with different servlet containers and understand the architecture and method of work
d) Configure and set up web servers
e) Master basic concepts about java servlets
f) To have hands on experience on using java servlets
g) To enable traineed to work with java beans
h) Differentiate between servlets and JSP

- **Teaching & Learning patterns**

Lectures , Tutorials (lab) , Practical Classes, supervised projects

- **Indicative content**

- What a servlet is and its advantage,
- understanding the difference between an application server and a web server,
- servlets lifecycle,
- the difference interfaces and classes of servlets package and HTTP servlet package,
- cookies and session tracking in servlets and JSP,
- introduction to JSP,
- JSP architecture and syntax,
- java beans,
- configuring web servers

- **Assessment**

20% project, 20% test and coursework and tests, 60 % exam

- **Reading List**

- Marty Hall Core Servlets and Java Server Pages, Volume 1: Core Technologies by (2nd Edition)
- Andrew Thomas *et.al*. Proffessional Java Servlets 2.3., 2004
- Ralph F. Grove, Web Based Application Development, Johns & Bartlet, 2010
- Santosh Kumar K. , Kogent Solutions Inc., Santosh Kumar K. And Kogent Solutions Inc., Jdbc, Servlets, And Jsp Black Book, New Edition, Dreamtech Press, 2009

- **BSE 2206: Data Communications**

Pre-requisites: Computer Networks

- **Course Description**

This is a theoretical course that covers the fundamentals of data communication Formatting and transmission of digital information over various media

- **Aims**

The goal of this module is to:

- provide a good understanding of the electrical characteristics of digital signals and the basic methods of data transmission.
- introduce the concept of communication protocols and give an overview of Data Communication Standards.
- give an introduction to the area of computer networks, with emphasis on the range of communication protocols utilized.
- explore the concept of Open Systems, giving an overview of Transport and Application Support Protocols.

- **Learning outcomes**

The goal of this course is for students to:

- understand basic functions on which modern communication systems are built
- know how packets find their way through the Internet, and how congestion is avoided
- know how wireless communication works
- be able to develop programs that efficiently communicate over a network
- to improve C programming skills
- have the required basis for working in this area

- **Teaching & Learning patterns**

This course will be delivered through class lectures, discussions. Simulations will also be used to explain abstract concepts.

- **Indicative content**

a) Introduction to communication
b) Digital versus Analog transmission; Modems
c) Transmission media: magnetic media, twisted pair, coaxial, fiber-optics;
d) Data encoding: straight, Manchester, differential Manchester, satellite;,
e) modulation and their standards, codes and pulse code modulation;
f) Integrated Services Digital Networks (ISDN);
g) Network Access Protocols; Passive versus dynamic allocation;
h) LAN standards:802.3 (Ethernet),802.4 (token bus), 802.5 (token ring);

i. Assessment Method

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

ii. Reading List

b. Behrouz A. Forouzan, Data Communications and Networking,: Fourth Edition
c. Kurose, JF & Ross, KW, Computer Networking: A Top Down Approach, 4th edn, Addison-Wesley, 2007.
d. David Stamper et al, *Business Data Communications, 6th Edition*, 2003, Prentice Hall.
e. Fred Halsall, *Data Communications, Computer Networks, and Open Systems, 4th Edition*, 1998, Addison-Wesley.

- **MTH 1203: Calculus I**

Pre-requisites: None

a) Course Description

The course gives the students a strong mathematical base to be able to tackle other computer problems. The course provides techniques that commonly used in the general area of computer science. It builds a foundation for other courses that need special mathematical backgrounds.

b) Aims

The aims of the course are:

- To provide students with a mathematical base that is to be used to solve computer science problems
- To improve the problem solving skills of students

c) Learning outcomes

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

a) interpret a function from an algebraic, numerical, graphical and verbal perspective and extract information relevant to the phenomenon modeled by the function.
b) find points of discontinuity for functions and classify them
c) understand the consequences of the intermediate value theorem for continuous functions
d) sketch the graph of the derivative from the given graph of a function.
e) compute the value of the derivative at a point algebraically using the (limit) definition compute the expression for the derivative of a composite function using the chain rule of differentiation

d) Teaching and learning pattern

The teaching will largely involve lectures, together with tutorials and take home assignments.

e) Indicative content

- Functions
  - Polynomials (linear, quadratic) rational, composite functions
  - Transedental functions: logarithmic and exponential functions;
  - The trigonometrical functions and their inverses
  - the hyperbolic functions and their inverses
- Limits
  - Informal definition of limits of functions and continuity;
  - one sided limits
  - removable discontinuity
  - Techniques and theorems of evaluating limits

- Formal definition of limits
- application to definition and properties of continuous functions
- Use of the definition in proofs and problem of limits and continuity

- Differentiation
  - Definition of a derivative, continuity and differentiability
  - Rules and theorems of determining derivatives
  - Inverse functions: their derivatives and graphs
  - Differentials: applications to approximation. Rolle's theorem, mean Value Theorem, L'Ohopital's Rule
  - Anti-derivatives: Techniques and theorems for determining anti derivatives
  - Intergration: Define Intergral, Rieman sums, the definite intergral and area,
  - The fundamental theorem of calculus: application to evaluation f definite intergrals (by substitution)
  - Functions defined by integration: f(t) dt as a n anti derivative of f(x), mean value theorem for intergrals

f) Assessment method

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

g) Reading List

- George B. and Ross L. Finney Calculus and Analytic Geometry (9th Edition), Addison Wesley, 1995.
- R. E Walpole. Introduction to Statistics by, 3rd Ed, Prentice Hall, 1982
- Stewart, James, Calculus: Early Transcendentals, Brooks Cole, 2010
- Bittinger, Marvin L., Ellenbogen, David J., Surgent, Scott Adam, Calculus and Its Applications, Addison Wesley, 2011
- Goldstein, Larry J., Schneider, David I., Lay, David C. , Calculus and Its, Applications, Prentice Hall Professional Technical Ref, 2009

▪ CSC 1214: Object-Oriented programming

Pre-requisites: BSE 1100 and CSC 1205

a) Course Description

Description The course is to give an in depth understanding of Object Oriented programming. It is to introduce students to Object Oriented programming practices like inheritance, interfaces, exception handling, action handling, abstraction, software reuse etc.

b) Aims

The aim of the course is to:

- Introduce students to object oriented programming using Java
- Train students to develop complete computer applications.

c) Learning outcomes

- An understanding of the principles and practice of object oriented analysis and design in the construction of robust, maintainable programs which satisfy their requirements;
- Designing, writing, compiling, testing and executing straightforward programs using java
- Appreciating the principles of object oriented programming;
- Becoming aware of the need for a professional approach to design and the importance of good documentation to the finished programs.

d) Teaching & Learning patterns

This will include lectures and lab assignments. A number of real-fife problems shall be given to the students to apply the theoretical aspects

e) Indicative content

- The object oriented paradigm
- Classes and objects
- Inheritance and visibility modifiers
- Interfaces and abstract classes
- Java methods
- Exception handling
- Java input/out put and file handling
- Arrays and commandline arguments
- Applets
- Graphical user interface and action handlers

f) Assessment Method

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

g) Reading List

a) John Lewis and William Loftus, Java Software Solution: Foundations of Program Design (6th Edition), Addison-Wesley, 2009.

b) Khalid A. Mughal and Rolf W. Rasmussen, A Programmer's Guide to JavaTM Certification: A comprehensive Primer , Addison-Wesley, 1999.

c) Barnes, David and Kolling, Michael (2009). Objects First with Java (4th edition). Pearson ISBN 0137005628

d) Erich Gamma (Author), Richard Helm (Author), Ralph Johnson, Design patterns : elements of reusable object-oriented software, Addison Wesley; 1 edition (31 Oct 1994)

e) Peter Coad and Jill Nicola ,Object-Oriented Programming, Prentice Hall; 1 edition (February 13, 1993)

- **Year 2 Recess Term**
  - **BSE 2302: Professional Software Engineering Mini Practical Project II**

Pre-requisites: This course assumes the students have covered a mobile programming course

a) Course Description

This course provides students with a holistic hands-on experience in building software products on the mobile platform in a team environment

b) Course aim and Objectives

These are, to:

- equip students with practical skills in software engineering from requirements solicitation to testing;
- provide students with mentorship in writing software products for mobile devices;
- give students an opportunity to build team-work and leadership skills needed in the workforce.

c) Learning outcomes

Upon successful completion of the course, the student should: (i) Demonstrate mastery of a (mobile) specific programing language and development platform; (ii) Carry out requirement engineering for a particular project; and (iii) Demonstrate ability to develop a mini application or systems with minimal difficulty.

d) Teaching & Learning patterns

Lectures, practicals, group work, & tutorials

e) Indicative content

An introduction to a specific professional software certification course is covered.

f) Assessment Method

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

g) Reading List

- Andrew Koenig and Barbara E. Moo, Accelerated C++: Practical Programming by Example, , Addison-Wesley, January 2008, ISBN 020170353X
- Jeff Friesen, Beginning Java SE 6 Platform: From Novice to Professional, , Apress October 2007, 159059830X
- Steve Karam, and Robert Freeman, Easy Oracle Jumpstart: Oracle Database Management Concepts and Administration, ISBN 0-9759135-5-7
- Steven Feuerstein, Bill Pribyl and Chip Dawes,,Oracle PL/SQL Language Pocket Reference, by Fourth Edition, October 2007, OReilly, ISBN 10: 0-596-51404-2

- **Year 3 Semester I**
  - **BSE 3110: Object-Oriented Analysis and Design**

Pre-requisites:

- Foundations of Software Engineering or equivalent background.
- Ability to program in an OO programming language (e.g., C++, Smalltalk, Java, C#, Python, Delphi, VB.net, etc.)

a) Course Description

This course introduces the fundamental principles of object oriented approaches to modeling software requirements and design. In this course, the students will learn how to produce detailed object models and designs from system requirements; use the modeling concepts provided by UML; identify use cases and expand into full behavioral designs; expand the analysis into a design ready for implementation and construct designs that are reliable. The course begins with an overview of the object oriented analysis and design. The following figure shows the flow of the course.

b) Course aim and objectives

By the end of this course the student should be able to:

- Apply an iterative process such as the Unified Process.
- Analyze software requirements and document these requirements using Use Cases.
- Perform software analysis and record the results using UML notation.
- Perform software design and record the results using UML notation.
- Apply object-oriented patterns.
- Discuss how object-oriented software development affects testing and quality

c) Learning outcomes

At the end of the course, each graduate should be able to;

a) Design and implementation of programs
b) Formulate and solve problems in computing."
c) Understand design and performance requirements of software systems.
d) Apply sound principles to the synthesis and analysis of computer systems
e) Engage in lifelong learning and expect to embrace change
f) Communicate effectively and think critically and creatively, both independently and with others
g) Be aware of social and ethical issues of computers in society

d) Teaching & Learning patterns

The teaching and learning approaches will combine classroom lectures with theories and discussion of case studies in groups. Take home assignments / coursework will be administered.

e) Indicative content

a) Principles of Object Technology
b) Project Organization and Communication
c) Fundamentals of Visual Modeling with UML: Business Modeling.
d) Object-Oriented Analysis with UML.
e) Object-Oriented Design with UML (Part 1).
f) Object-Oriented Design with UML (Part 2).
g) Dealing with Complexity
h) Mapping Models to Code
i) Testing

f) Assessment Method

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group

projects – 20%): 40%
Final examination (60%)

g) Reading List

- Jim Arlow and Ila Neustadt, UML2.0 and the Unified Process: Practical Object-Oriented Analysis and Design.
- Bernd Bruegge and Allen H. Dutoit. *Object-Oriented Software Engineering, Using UML, Patterns, and Java, 3rd Edition*, Prentice-Hall, 2010, ISBN-10: 0136066836.
- Craig Larman, *Applying UML and Patterns - An Introduction to Object-Oriented Analysis and Design and Iterative Development*, Prentice Hall, 2004, ISBN: 0-13-148906-2. resources
- Timothy C. Lethbridge and Robert Laganiere, *Object-Oriented Software Engineering*, by, McGraw-Hill, 2001, ISBN: 0-07-709761-0.
- Bernd Oestereich*, Developing Software with UML, Object-Oriented Analysis and Design in Practice*, Addison-Wesley, 1999, QA76.9.03503713 1999.
- G. Booch, Benjamin/Cummings, *Object-Oriented Analysis and Design with Applications, 2nd ed.*, Redwood City, CA, 1994, QA76.64.B66 1994.
- Anton Eliens, *Principles of Object-Oriented Software Development*, Addison-Wesley, 1995, ISBN: 0-201-62444-3.

- **BSE 3111: Requirements Engineering**

Pre-requisites: None

- **Course Description**
  The course will discuss concepts for systematically establishing, defining and managing the requirements for large, complex, changing and software-intensive systems, from technical, organizational and management perspectives. The course will consider the past, present and future paradigms and methodologies in requirements engineering. The course will cover informal, semi-formal and formal approaches, while striking a balance between theory and practice. The course will involve building models of both requirements engineering process and requirements engineering product, concerning both functional and non-functional goals/requirements/specifications, using a systematic decision-making process.
- **Aims**
  At the successful completion of this course, learners should be able to understand:
  - the need for requirements for large-scale systems;
  - the stakeholders involved in requirements engineering;
  - requirements engineering processes;
  - models of requirements;
  - functional requirements;
  - non-functional requirements;
  - scenario analysis; and
  - object-oriented and goal-oriented requirements engineering.
- **Learning outcomes**
  On completion of the course, the student will be able to :
- understand the basics of Requirements Engineering
- prepare for, and undertake the requirements elicitation tasks
- analyse client needs
- create models of requirements using a variety of notations and techniques

- prepare software requirements specifications using an industry standard, and
- prepare for, and undertake formal specification reviews.

- **Teaching & Learning patterns**
  The teaching and learning approaches will combine classroom lectures with theories and discussion of case studies in groups. Take home assignments / coursework will be administered.

- **Indicative Content**
  - Introduction to Requirements Engineering
  - Requirements Engineering Processes
  - Requirements Models
  - Requirements Analysis, and Modeling
  - Requirements Elicitation
  - Scenario Analysis
  - Modeling Enterprises
  - Structured Analysis
  - Goal-Oriented Requirements Engineering
  - Specifications & Validation
  - Managing Change and Inconsistency
- **Assessments Method**
  Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
  Final examination (60%)
- **References**
  - Axel van Lamsweerde Requirements Engineering: From System Goals to UML Models to Software Specifications, , John Wiley Sons
  - G. Kotonya and I. Sommerville,Requirements Engineering: Processes and Techniques, John Wiley Sons
  - I. Sommerville and P. Sawyer Requirements Engineering - A Good Practice Guide, Wiley
  - Moore , James W, *The Road Map to Software Engineering: A Standards Based Guide* , 1st edition,., Wiley-IEEE Computer Society Press, 2005.
  - Software Engineering Code of Ethics and Professional Practices. Full Version. IEEE-CS/ACM Joint Task Force on Software Engineering Ethics and Professional Practices.

- **BSE 3104: Software Metrics**

Pre-requisites: None

- **Course Description**
  This course introduces students to measurement theory in software looking at measurement techniques, attributes in both processes and products.
- **Aims**
  The course is composed of the following basic modules: Measurement theory (overview of software metrics, basics of measurement theory, goal-based framework for software measurement, empirical investigation in software engineering), Software product and process measurements (measuring internal product attributes: size and structure, measuring external product attributes: quality, measuring cost and effort, measuring software reliability, software test metrics, object-oriented metrics Measurement management

- **Learning outcomes**
  Upon successful completion of this course students should be able to:
  - Describe software metrics;
  - Understand the foundations of measurement theory and models of software engineering measurement;
  - Appreciate software products metrics, software process metrics and measuring management.
- **Teaching & Learning patterns**
  Lectures, class presentations, assignments
- **Indicative content**

  - Measurement: what is it and why do it?
  - The basic of measurement
  - A goal-based framework for software measurement
  - Empirical investigation
  - Software-metrics data collection
  - Analyzing software-measurement data
  - Measuring internal product attributes: size
  - Measuring internal product attributes: structure
  - Measuring external product attributes
  - Software reliability: measurement and prediction
  - Resource measurement: productivity, teams, and tools
  - Making process predictions
  - Planning a measurement program
  - Measurement in practice
- **Assessment Method**
  Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
  Final examination (60%)
- **Reading List**
  - Norman Fenton, James Bieman, Taylor & Francis, Software Metrics: A Rigorous and Practical Approach, (3rd ed. illustrated, revised) (652p.), 2013. ISBN: 1439838224, 9781439838228.
  - Stephen H. Kan Metrics and Models in Software Quality Engineering, , 2nd ed. (560 p.), Addison-Wesley Professional (2002). ISBN:0201729156.
  - John C. Munson Software Engineering Measurement, , Auerbach Publications, 2003 (443 pages) ISBN:0849315034
  - BA Kitchenham Software Metrics: Measurement for Software Process Improvement, , Blackwell Pub, 1996.ISBN: 1855548208.
  - C. Jones Applied Software Measurement: Assuring Productivity and Quality, , McGraw-Hill, 1996

- **CSC 3119: User Interface Design**

Prerequisite : None

a) **Description:**

The course introduces the principles of user interface development, focusing on design, implementation and evaluation.

b) **Aims**

The course aims at providing the skills listed below to students:

- Developing efficient, flexible and interactive User Interfaces(UI) Provide ability to identifying system users, the tasks they want to carry out and the environment in which they will be working;
- Creating conceptual designs;
- Designing various kinds of UI, in particular graphical user interfaces (GUIs) and web sites;
- evaluating UIs;
- Appreciation of realities of developing usable UIs in an organization
- Teaching and Learning Patter
- The teaching pattern is by lectures, lab sessions and projects

c) **Indicative content**

- Usability
- User-Centered Design
- UI Software Architecture
- Human Capabilities
- Output Models
- Conceptual Models and Metaphors
- Input Models
- Design Principles
- Paper Prototyping
- Constraints and Layouts
- Graphic Design
- Computer Prototyping
- Heuristic Evaluation
- User Testing
- Experiment Design
- Experiment Analysis

d) **Assessment Method**

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%

Final examination (60%)

e) Reading List

- Norman, D. A. The Design of Everyday Things. New York, NY: Doubleday, 1990. ISBN:0385267746.
- Nielsen, J. Usability Engineering. Burlington, MA: Academic Press, 1994. ISBN: 0125184069.
- Mullet, K., and D. Sano.Designing Visual Interfaces: Communication oriented techniques Prentice Hall, 1994. ISBN: 0133033899.
- Steve Krug, Don't Make Me Think: A Common Sense Approach to Web Usability, 2nd Edition, New Riders; 2nd edition (August 28, 2005)
- Jef Raskin, The Humane Interface: New Directions for Designing Interactive Systems, Addison-Wesley Professional (April 8, 2000)

- **BSE 3106: Mobile Networks and Computing**

Pre-requisites: None

- **Course Description**

  This course examines principles, design, implementation, and performance of mobile computing and wireless networking. The aim is to lay a foundation in the student's understanding and skills in mobile computing and wireless networking standards, technologies, application and services. Ideally the course is an integration of Wireless Networking and Mobile Computing.

- **Aims**

- To introduce students to the theory and practice of Mobile networking and computing.
- To facilitate the development of technical skills in mobile application development platforms particularly JME and Android.
- To enhance students skills in mobile application development using Java and other technologies
- To introduce students to the fundamental concepts in wireless technology and mobile computing including standards, technologies, devices and services
- To use and experiment with new technology and cutting-edge projects
- To understand how networking research is done
- Investigate novel ideas in the area via semester-long skill development projects.

- **Learning outcomes**

- Have gained an understanding of the theory and practical aspects of mobile computing and wireless networking.
- Discuss the considerations in wireless mobile networking architectures
- To design and implement wireless and mobile networks using Bluetooth, Wi-Fi among others
- Work with the JME and or Android platforms with minimal difficulty
- Develop simple mobile applications deployable on Java and/or Android enabled mobile devices
- Discuss the current research directions in mobile and wireless networking
- Demonstrate mastery of development and deployment of secure mobile services.

- **Teaching & Learning patterns**

  Lectures, practical laboratory sessions, class presentations

- **Indicative content**

- Subjects of study under Mobile Networking will include;
- Wireless Network technologies (including GSM/GPRS/3G & Wireless LANs),
- Convergence networks, NextGen,
- Mobile IP,
- wireless ATM,
- Wireless Ad Hoc Networks and Bluetooth.
- While subjects of discussion under Mobile Computing will include; Mobile Computing Architectures (including SMS/MMS/SIM, WAP, I-mode and JME), location–based services.

- **Assessment Method**

  Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%

  Final examination (60%)

- **Reading List**

- Zheng P, Ni LM (2006) Smart phone and next generation mobile computing. Burlington: Morgan Kaufmann. 429p.
- John W. Muchow, Core J2ME™ Technology & MIDP

- T. Imielinski and H. Korth, Kluwer, Introduction to Mobile Computing edited, Academic Publishers, 1996.
- Mark Beaulieu, "Wireless Internet Applications & Architectures: Building Professional Wireless Applications Worldwide", Addison-Wesley, 2002.

▪ BSE 3109: Introduction to Embedded Systems

Pre-requisites: Computer organization and architecture, programming with C

a) Course Description

The course introduces students to the design and implementation of embedded systems. Students are able to interface different peripherals to a microprocessor in order to create intelligent systems.

b) Aims

To address practical issues in design and development of *embedded systems* composed of *software* and hardware, Understand the process and fundamentals of integrating microprocessor-based embedded system elements to realize systems that not only meet functional requirements, but timing and performance requirements as well, Use practical skills to design and integrate a real-time operating systems with a microprocessor to host real-time service data processing

c) Learning outcomes

Upon completion of the course, students should

- Be able to analyse embedded systems problems, identify requirements and design the systems
- Be able to interface different peripherals to embedded systems

d) Teaching & Learning patterns

Lectures combined with practical simulations, Lab sessions during which devices are assembled and programmed. Student group projects and research

e) Indicative content

- Interfacing both input and output devices to a selected microprocessor i.e. Sensor and Actuator IO: ADC, DAC, servos, relays, stepper motors, H-bridge,
- Real-time embedded test equipment,
- software debug tools, and methods of performance profiling and tracing,
- Real-time applications including digital control, and robotic system command and control.
- Using tools like keil and proteus to simulate embedded systems.
- Serial communication, programming timers, interrupts and serial communication.

f) Assessment Method

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%

Final examination (60%)

g) Reading List

- Jivan *et. al*. Exploring C for microcontrollers: a hands on approach. Springer 2007
- David Calcutt, Fred Cowan, Hassan Parchizadeh, 8051 microcontroller: An applications based introduction, 2004
- Dogan Ibrahim, Microcontroller project in C for the 8051, 2000
- Ian R. Sinclair Sensors and transducers
- Marcus O. Durham, Systems design and the 8051, he hardware, firmware, and software design of microprocessor systems, PhD, PE, 2004

▪ BSE 3108: Communications Systems Design

Prerequisites:

- Basic programming
- Data communications or computer networks or Network management

- Computer hardware basics
- Some knowledge in operating systems

- **Course Description**

This course offers students an opportunity to learn about advanced emerging technologies within the associated market and business development. Students work in teams together with other students from different programmes and/or different universities, with an interdisciplinary teaching team, and representatives from industry in projects where specialist knowledge from different disciplines is blended. It is an advanced, demanding, fast-paced, comprehensive, and inspiring learning environment. The course is for anyone who likes to work in a context where the rule is: teach yourself and learn from others in the course, as well as those that have done similar work previously in similar or other contexts.

- **Aims**

This course aims to allow students experience real life working environment and achieve skills to increase their future employability.

- **Learning Outcomes**

At the end of this course the student will be able to:

- Demonstrate the ability to solve a real-world problem
- Demonstrate independent learning ability
- Demonstrate effective project management
- Show communication skills while working on the project and in presenting the solution
- Work as a successful team

- **Indicative content**

The content of this course should be focused on practical implementation of new technology, or new fields of use from existing technology. The students are expected to work as a team and show the ability to comprehend a real world problem and produce a solution.

- **Teaching & Learning patterns**

The teaching pattern is by lectures, lab sessions and group projects

- **Assessment Method**

Assessment will be based on 6 parts:

- Attendance: 5 %
- Written material: 30%
  - Project plan
  - Final report
- Presentations, 10 %
- Knowledge and project performance quality, 50 %
- Group dynamics : 5 %

- **References**

- B. Carlson and P. B. Crilly, Communication Systems, 5th edition, McGraw Hill, 2010
- J.Proakis and M. Salehi, Digital Communications, 5th Edition, McGraw Hill, 2008
- J.Proakis and M. Salehi, Digital Communications, 5th Edition, McGraw Hill, 2008
- A. Goldsmith, Wireless Communications, Cambridge, 2005
- D. Tse and P. Viswanath, Fundamentals of Wireless Communication, Cambridge, 2005

- **BSE 3105: Software Evolution**

Pre-requisites: None

a) **Course Description**

This course introduces the student to evolution of systems. It focuses on issues in maintaining software, change management, among others.

b) **Aims**

To enable the student understand how systems evolve and how to manage their evolution.

c) **Learning outcomes**

At the end of this course, the student should understand ;

- Software evolution compared to customization and adaptation
- How to manage software changes
- How to re-engineer systems to improve performance
- How to manage legacy systems
- How to manage products of system change.

d) **Teaching & Learning patterns**

In this course a combination of lectures and practicals will be used

e) **Indicative content**

- Introduction to Software evolution
- Managing software change
- Software Re-engineering
- Evolution of legacy system
- Configuration management

f) **Assessment Method**

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

g) **Reading List**

- Lowell Jay Arthur Software Evolution: A Software Maintenance Challenge, John Wiley & Sons, 1988, ASIN: 0471628719.
- Somerville, Software Engineering. The 8th Edition contains a size-able section on software evolution, which provides a basic overview. Other editions like 6th edition, 7th edition (chapters 18 and 19 and 21) have some nice literature on software evolution.
- Pigoski, Thomas, Practical Software Maintenance, M. New York, Wiley Computer Publishing, 1997. ISBN 0471170011
- Lewis, William, Software Testing and Continuous Quality Improvement,. CRC Press 2000. ISBN 0849398339
- Gilb, Tom, Principles of Software Engineering Management, , Reading, Mass.: Addison Wesley, 1988. ISBN 0-2011-9246-2

- **Year 3 Semeser II**
  - **BSE 3201: Software Architecture**

Prerequisites:

- Appreciation of the software development process
- Familiarity with typical software development roles

a) **Course Description**

This course provides the student with a structured overview of the process of architecting a software-intensive system. It describes the tasks in which the architect is involved throughout the life of a project in conjunction with different approaches (waterfall, iterative, agile). It discusses the specific concerns of the architect such as the definition of architecture, the characteristics of the architect, the benefits of architecting, documenting a software architecture, reuse, the relationship between requirements and architecture, and architecting at both a logical and physical level. It also touches upon more advanced topics such as architecting complex systems.

b) Aims

a) Describe the key tasks in which the architect is involved
b) Describe the key work products that the architect produces
c) Describe the characteristics of a software architect
d) Position the benefits of architecting
e) Describe the involvement of the architect throughout the life of a project
f) Document a software architecture
g) Consider the systematic reuse of assets on a project
h) Understand the involvement of the architect in requirements definition and management
i) Apply proven architecture-related techniques in defining a solution

c) Learning outcomes

At the end of the course, students will:

- Be familiar with the latest state-of- the-art software architecture;
- Appreciate software system design; and
- Understand how system's components are meant to interact with each other.

a. Indicative Content

The topics to be covered in this course unit are: Architectural styles, Components of architectural design, Connectors, components, composition, Architectural design guidance and Tools for architectural design, Achieving quality goals with architectural styles, Formal models and specifications,
Analyzing software architecture with SAAM, Architecture description languages (ADLs), Architecture-based development, Patterns in software architecture, Reusing software architecture

b. **Teaching and learning pattern**:

The teaching and learning approaches will combine classroom lectures, discussions and group activities, quizzes and take home assignments. A group project shall form part of the coursework. The material presented in class will overlap that of the text but will contain additions and variations

c. **Assessment method**:

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

d. Reading List

a) Len Bass, Paul Clements and Rick Kazman Software Architecture in Practice, Addison Wesley 1998.
b) Shaw and Garlan Software Architecture: Perspectives on an Emerging Discipline, by, Prentice Hall 1996.
c) Cheesman and Daniels"UML Components: A simple process for specifying component-based software", Addison-Wesley 2000.

- **BSE 3208: Distributed Systems Development**

Pre-requisites:

- **Course Description**

A distributed system is broadly categorised as a collection or network of loosely coupled, autonomous

computers that can communicate with each other and execute logically separate computations, though these may be related to concurrent computations on other nodes. Distributed systems have become pervasive---many applications now require the cooperation of two or more computers--yet the design and implementation of such systems remain challenging and complex tasks. Difficulties arise from the concurrency of components, the lack of a global clock and the possibility of independent failure of components. Moreover designs must aim to provide inter-operability, transparency and autonomy. The emphasis of this module is on gaining understanding of the principles and concepts that are used to design distributed systems and experience of software platforms which underpin their development.

- **Aims**

To give students a firm grounding in developing distributed web-based systems. The student will learn about and gain practical experience in the tools and technologies of distributed systems development. Students will also be introduced to the theory of distributed system requirements.

- **Learning outcomes**

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

  - Present a conceptual model of distributed systems;
  - Describe key components of a distributed system and evaluate the tradeoffs of alternative architectural models and distributed systems frameworks;
  - Suggest algorithms suitable for application in distributed systems;
  - Build prototype implementations of distributed systems; and (v)Demonstrate an understanding of the challenges faced by future distributed systems

- **Teaching & Learning patterns**

Lectures, practical laboratory sessions and assignments, class presentations

- **Indicative content**

  - Event-driven software architectures, distributed object computing, and developing, documenting, and testing applications using distributed computing frameworks like Java Enterprise Edition (JEE) Web and Business Components, XML and RESTful web services.
  - Techniques that enable the construction of reusable, extensible, efficient, and maintainable concurrent and distributed software systems are emphasized using Distributed Systems Design Patterns.
  - Abstraction based on patterns and object-oriented techniques will be crucial throughout the course, and their application studied in in-depth using case studies and their application and implementation by the distributed computing frameworks.

- **Assessment Method**

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

- **References**

  - S. Tanenbaum and M. V. Steen, Distributed Systems: Principles and Paradigms, Second Edition, Prentice Hall, 2006, ISBN: 0132392275.
  - Anura Gurege, Web Services: Theory and Practice, 1rd Edition, Digital Press, 2004, ISBN: 1555582826.
  - R. Anderson, Security Engineering: A Guide to Building Dependable Distributed Systems, John Wiley & Sons, 2001, ISBN: 0471389226.

- **BSE 3209: Advanced Object-Oriented Programming**

Pre-requisites Object-Oriented programming

- **Course Description**

The course provides advanced concepts of java to build both desktop and web projects. Students are expected to use the skills attained in the java introductory course to build GUI programs, work with java web frameworks like spring, modular programs and be able to work with popular IDEs like netbeans

- **Aims**

The course is meant to introduce advanced object oriented programming with java

- **Learning outcomes**

At the end of the course, students should

- Demonstrate mastery of advanced java concepts like use of frameworks, building of modular projects etc
- NetBeans Platform 6.9 Developer's Guide by Jürgen Petri 2010

- **Teaching & Learning patterns**

Students with the help of the lecturer will do intensive research. The teacher will introduce concepts and students will be required to present projects for every concept through implementing projects in form of course works using a selected framework

- **Indicative content**

JDBC, persistence with hibernate, Java with socket programming, thread programming, GUI with Swing, java frameworks (netbeans or Spring)

- **Assessment Method**

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

- **References**

- Eric Pugh, Joseph D. Gradecki, Professional Hibernate
- Paul Hyde, Java Thread Programming, Learn how to use threads for faster, more efficient Java programming by by Sams © 1999, 510 pages
- Cay S. Horstmann, Gary Cornell, Core Java™ 2: Volume II–Advanced Features Prentice Hall PTR, 2001 ISBN : 0-13-092738-4

- **BIT 2207: Research Methodology**

Pre-requisites: Systems Analysis and Design

a) Course Description: This course unit enables students to learn and apply principles of conducting scientific research, or undertaking a systematic research study.

b) Aim: To enable the student to relate data gathering techniques and principles taught in Systems Analysis and Design (and software engineering or other course units in his/her programme) to his/her final year research project.

c) Learning outcomes: A Student who has undertaken this course unit will be able to learn skills that will enable him/her to successfully undertake a research project. He/she will be able to:

a) Identity a relevant or significant research problem.

b) Identity the aims of a research project that can solve a given problem.
c) Select appropriate research methods to be used in solving a given project.
d) Select appropriate data collection techniques that can be used to gather data required to solve a given project.
e) Select appropriate data analysis techniques and use them to process collected data, interpret data analysis results.

d) Teaching and learning pattern

- Lectures
- Group work and discussions aiming at applying research concepts and principles in e.g. identifying problems, defining objectives, defining research questions, formulating hypothesis, conducting literature reviews, selecting research methods etc.

e) Indicative content

- The course unit is divided into mainly five (5) parts.
- Part I covers introduction concepts of research. E.g. what is research, understanding the research process and fundamental concepts of research, how to formulate a research problem, research objectives, research questions, how to define scope, how to conduct literature review in a given study etc.
- Part II covers the various research methods. Student learns the various quantitative and qualitative research methods. Student also learns the data gathering techniques.
- Part III covers the evaluation phase of a research project, testing and validation of developed tool or research results. E.g. How to evaluate, testing, and validate research results; what are the evaluation criteria? What are the validation methods? What is one testing in the testing phase?
- Part IV covers the practical application of concepts learned from parts I to III. The lecturer suggests one or several class projects that require practical application of concepts learned from parts I to III.
- Part V covers report writing. E.g. how to prepare a research report, how to report research results/findings, how to present feedback or findings from evaluating the developed tool or research results etc.

f). Assessment Method

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

- **Reading lists:**

- Cooper, H. (1998). Synthesizing Research: A Guide for Literature Reviews. Thousand Oaks, California: Sage Publications.
- Saunders, M, Lewis, P & Thornhill, A (2003), Research Methods for Students, 3rd edn, UK, Financial Times, Prentice Hall.
- Peter Bock, Getting It Right: R&D Methods for Science and Engineering, Academic Press; 1 edition (September 13, 2001), 406 pages, ISBN-10: 0121088529, ISBN-13: 978-0121088521
- Justin Zobel, Writing for Computer Science, Springer; 2nd edition (April 27, 2004), paperback: 280 pages, ISBN-10: 1852338024, ISBN-13: 978-1852338022; web page for the book

- Wayne C. Booth, Gregory G. Colomb, and Joseph M. Williams, The Craft of Research, 2nd edition (Chicago Guides to Writing, Editing, and Publishing), University Of Chicago Press; 1 edition (March 2003), paperback: 336 pages, ISBN-10: 0226065685, ISBN-13: 978-0226065687

- **BIT 2208: Systems Administration**

Pre-requisites: None

- **Course Description**

This course addresses both the technology of computer systems and the users of the technology on an equal basis. It is about putting together a network of computers, getting them running and then keeping them running in spite of the activities of users who tend to cause the systems to fail.

- **Aims**

A student that undertakes this course should:

a) Be able to use Windows and Linux operating systems.
b) Be able to perform basic administration under Windows and Linux.
c) Be able to design a network, which is logical and efficient.
d) Be able to deploy large numbers of machines, which can be easily upgraded later.
e) Be able to decide what services are needed.

- **Learning outcomes**

On completion of this course unit, the students will be able to:

f) Demonstrate understanding of computer networking, computing models, and basic network services.
g) Recognize and describe logical and physical network topologies in terms of the media and network hardware.
h) Demonstrate understanding of computer networking, computing models, and basic network services.
i) Compare current network technologies in terms of speed, access method, operation, topology, and media.
j) Describe basic principles of Unix/Linux multi-user System Administration

d) Teaching and learning patterns

The teaching style will be facilitation, with students being broken up into discussion groups for each major topic after which practical session will be undertaken.

e) Indicative content

- Manage system resources including processes, memory and disk space
- Maintain and interpret log files
- Configure and manage a DNS service
- Create and manage user accounts
- Configure and manage an email system
- Install and manage shared applications
- Use shell scripts to automate procedures
- Understand and manage cross-platform file services
- Understand and manage Windows/Unix/Linux printing systems
- Monitor, analyze and tune system performance
- Manage and configure virus protection strategies
- Implement any other security measures
- Perform system upgrades and version management
- Perform system backup and recovery procedures

f) Assessment Method

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group

projects – 20%): 40%
Final examination (60%)

g) Reading List

a) Mark Burgess (2004) Principles of Network and System Administration. . Published by Wiley and Sons 2nd Edition. ISBN 0-470-86807-4

b) www.tldp.org

c) Thomas Limoncelli (Author), Christina Hogan (Author), Strata Chalup (Author), The Practice of System and Network Administration, Second Edition, Addison-Wesley Professional; 2 edition (July 15, 2007)

d) Evi Nemeth , Garth Snyder, Trent R. Hein , Ben Whaley, UNIX and Linux System Administration Handbook (4th Edition), Prentice Hall; 4th edition (July 24, 2010)

e) Æleen FrischEssential System Administration: Tools and Techniques for Linux and Unix Administration, 3rd Edition, O'Reilly Media; 3rd edition (August 15, 2002)

- **CSC 2209: Systems Programming**

a) Course Description

Systems programming is aimed at teaching students how to write programs using system level services. The system of instruction is UNIX due to availability of free system tools that have been largely developed by and for the academia

b) Aim

Skills in tools are provided by systems, their commands, system calls and understanding for model of computation.

c) Teaching and Learning Pattern

The teaching pattern is by lectures, lab sessions and projects

d) Indicative content

- Introduction and Unix Standardization
- File input and output
- Standard I/O Library
- Files and Directories
- System Data Files and Information
- Process Environment
- Process Control
- Process Relationships
- Signals
- Threads
- Advanced I/O
- Interprocess Communication

e) Assessment Method

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

f) Reading List

⌀ W. Richard Stevens, Advanced programming in the Unix Environment, by, Addison-Wesley**,** 2008

⌀ Adam HooverSystem Programming with C and Unix, Addison-Wesley; 1 edition (February 23, 2009)

⌀ Johnson M. Hart, Windows System Programming (4th Edition), Addison-Wesley Professional; 4 edition (February 26, 2010)

⌀ Srimanta Pal, Systems Programming, Oxford University Press, USA (October 9, 2011)

- **Year 3 Recess Term**
  - **BSE 3302: Field Attachment**

While this course is indicated at being conducted during the Year 3 recess term, students who can find field attachment placement during the Year 3 end Semester I university break (December to January) and can carry on with Field attachment into the first few weeks Semester 2, particularly the evening students will be allowed to undertake their internship starting at the end of Year 3 Semester I exams.

Pre-requisites

This course assumes the students have gained a good amount of practical and professional skills to practice in the field i.e. having successfully completed BSE 1301 & BSE 2301

- **Course Description**

This course provides students with a mentorship opportunity in a real work environment

- Aims

These are, to: (i) equip students with practical skills in software engineering from requirements solicitation to testing; (ii) provide students with mentorship in writing professional software products; (iii) give students an opportunity to build team-work, communication, appearance and leadership skills needed in the workforce.

- **Learning outcomes**

Upon successful completion students will:

- Have gained work experience that allows them to sample professional environments in which they might seek careers; and (ii) Have experience that will help prepare them for careers and research.
  - **Teaching & Learning patterns**

Mentorship, Practicals, & group work

  - **Indicative content**

The students shall acquire hands on training emphasing the software development process. Other relevant areas include e.g. maintenance, servicing and troubleshooting, and training in computer skills and applications as well as software development. Hardware maintenance; Practicals; On Job training; Report writing.

  - **Assessment Method**

Student Journal, Field Supervisor form, Academic supervisor's form and field attachment report

  - **Reading List**

There is no particular reference material for internship, except whatever may be recommended by the student supervisors

.

- **Year 4 Semester I**
  - **BSE 4100: Software Engineering Project I**

a) Course Description

The course covers practical aspects of the initial phases of a software engineering. It ensures practical application of skills in requirements gathering, and design and specification

b) Aims

Upon successful completion of this course the student will have ability to:

- Demonstrate independent skills in collecting requirements, documenting user requirements and technical requirements for non-trivial software engineering/ research projects by pursuing a lengthy Software engineering project; and

- Demonstrate skills of specifying, designing and implementing a project, with assistance of one of the Professors/ Lecturers as adviser/ supervisor.

c) Teaching and Learning Patterns

Lecture notes, student presentations, short exploratory assignments

d) Course Content

The student develops a framework within which research will be conducted and offers evidence of qualifications to pursue the research. Concepts and theories underlying the student's Project research are articulated, the problem is clearly stated, and specific, measurable goals are specified, a literature review is presented, the methods of conducting research are delineated, and strategy to achieve the goal is given.

e) Assessment

- a requirements document clearly specified in well-known and acceptable notation (30%)
- Technical Specification Document (30%)
- Detailed design document. (40%)

f) References

No particular reference will be used for this course unit, except whatever is recommended by the student supervisors.

▪ BSE 4101: Software Reliability and Testing

- **Description**

  This course is a step by step description of software quality and software reliability engineering process. It includes introduction to software quality, prediction and measurement of software size and cost, software reliability engineering process, defining necessary reliability, developing operational profiles, decision making based on the test results, techniques to improve and predict software reliability, application of quality concept to agile and incremental software development processes. The focus is on the reliability of object-oriented software systems. A workshop (project) is designed to reinforce the presented material. In the workshop, the students will actually go through the estimation and evaluation of quality of a realistic software project.
- **Aims**

  Upon successful completion of this course the student will have ability to:
- Understand the software reliability process and reliability growth models;
- Show techniques to improve and predict software reliability; and
- Appreciate concepts such as operational profiles, techniques to improve and predict software reliability, preparing and executing test, black box testing, white box testing, unit testing, system testing, and integration testing.
- **Indicative Content**:

  This course introduces software reliability process, reliability growth models and shows techniques to improve and predict software reliability. Concepts such as defining necessary reliability, developing operational profiles, techniques to improve and predict software reliability, preparing and executing test, black box testing, white box testing, unit testing, system testing, and integration testing will be explained.
- **Reading List**
- John D. Musa Software Reliability Engineering: More Reliable Software Faster and Cheaper, , (632 p.), Authorhouse, 2nd edition, 2004.ISBN 1418493872.

a) Michael R. Lyu(Editor), Handbook of Software Reliability Engineering, McGraw Hill (1996).ISBN: 0-07-039400-8.

b) J.D. Musa, A. Iannini, K. Okumoto Software Reliability: Measurement, Prediction and Application, (621 p.), McGraw-Hill (1987).ISBN 0-07-044093-X.

c) William E. Perry Effective Methods for Software Testing, , 2nd edition, JohnWiley and Sons (2000). ISBN: 0-471-35418-X.

- **BIS 3106: Business Process Management**

Course description: In this course students will be introduced to key concepts and approaches to business process management and improvement. The main focus of this course is both understanding and designing business processes. Students will learn how to identify, document, model, assess, and improve core business processes. Students will be introduced to process design principles. The way in which information technology can be used to manage, transform, and improve business processes is discussed. Students will be exposed to challenges and approaches to organizational change, domestic and offshore outsourcing, and inter-organizational processes.

Aims:

To

equip students with skills for modeling business processes
equip students with skills for benchmarking business process performance
equip students with skills for assessing business process performance
equip students with skills for designing business process improvements
make students understand the role and potential of IT to support business process management
make students understand the challenges of business process change
make students understand how to support business process change
make students understand different approaches to business process modeling and improvement
make students understand the challenges and risks concerning business process outsourcing
equip students with skills for using basic business process modeling tools
equip students with skills for simulating simple business processes and using simulation results in business process analysis

Learning outcomes:

After successfully completing this course, students should be able to:

model business processes
benchmark business process performance
assess business process performance
design business process improvements
describe the role and potential of IT to support business process management
describe the challenges of business process change
describe how to support business process change
describe different approaches to business process modeling and improvement
describe the challenges and risks concerning business process outsourcing
use basic business process modeling tools
simulate simple business processes and use simulation results in business process analysis

Teaching and learning patterns:

Lectures

Project-like assignments to be done in groups
Class discussions about the project-like assignments

Indicative content:

Overview
Understanding organizational processes
Process assessment
Process improvement
Using IT for process management and improvement
Organizational issues in business process management

Assessment:

Course work (Tests (20%), take home assignments, practical exercises (20%)
Final written exam: 60%

References:

John Jeston and Johan Nelis, 2008. Business Process Management, 2nd Edition. Publisher: Elsevier Ltd. ISBN: 978-0-75-068656-3.

Naresh Verma, 2009. Business Process Management: Profiting from Process. Publisher: Global India Publications Pvt. Ltd. ISBN: 978-81-907941-7-6.

Takayuki Kanda, Hiroshi Ishiguro, 2012.. Business Process Management: Concepts, Languages, Architectures, 2012.

John Jeston, 2008.. Business Process Management, Second Edition: Practical Guidelines to Successful Implementations.

Susan Page, 2010. The Power of Business Process Improvement: 10 Simple Steps to Increase Effectiveness, Efficiency, and Adaptability.

- **BAM **** Entrepreneurship Principles**

Pre-requisite Courses: None

g) Course Description:

The course introduces the students to the basic concepts in entrepreneurship, identification of business opportunities, business evaluation and analysis. It provides students with the skills needed to effectively identify, organize, develop, and manage own business ventures. This course is based on creativity and professional development foundations that should orient a student to take adventure, a personal journey, and a seize opportunity for business start-up. The course gives students an opportunity to make creative adjustments to meet personal needs and increase self-drive to achieving success.

h) **Aims:**

A student that undertakes this course should be able to:

- Understand the origins of entrepreneurship and an entrepreneur
- Identify, evaluate, and select business opportunities

- Perform a self-evaluation to match their own characteristics with that of an entrepreneur
- Carry out feasibility and viability of an investment opportunity
- Analyze and exploit the Entrepreneurial Environment provided by the political, socioeconomic and technological conditions.

i) **Learning Outcomes:**

At the end of the course the students should be able to;

- Perform self evaluation to match business opportunities
- Analyse the entrepreneurial environment
- Analyse start-up survival, sustainability of an investment opportunity ,
- identify their own personal entrepreneurial potential, ability, and competences
- identify, and exploit business opportunities and resources

j) Teaching and Learning pattern:

The teaching and learning approaches will combine use of case studies, Keynote lectures, student-led seminar presentations, site visits, and mini research

k) Indicative content:

- Entrepreneurship-Scope: Introduction, Entrepreneurship Defined, Importance of Entrepreneurship, Barriers to Entrepreneurship
- Entrepreneurship theories: Economic theories; Sociological theories ; Psychological theories ; Entrepreneurial Process;
- The Entrepreneur: Entrepreneur defined; Types of entrepreneurs; Emergence of entrepreneurs; Entrepreneurial traits
- Creativity and Innovation: Creativity and innovation defined; creativity and innovation processes; barriers to creativity and innovation; Factors that enhance creativity and innovation
- Feasibility study and analysis: Feasibility and Viability analysis; Idea generation; Process of carrying out a feasibility study; Feasibility Analysis Aspects; Components of a Feasibility report (Market feasibility, Technical feasibility, Financial feasibility, Operational feasibility)
- Business planning: Types of business plans; Uses of business plans; Users of business plans; Business Plan development process (Strategic focus review, Environmental Audit/Analysis, Developing Goals and Objectives, Developing appropriate strategies, Developing implementation plan, Developing a monitoring and control plan)
- Components of a business plan:Cover page, Table of Contents, Introduction, Company background; Shareholders/ Ownership; Legal status of the business; Vision, Mission, Core Values; Environmental Analysis; Target Market; PEST Analysis; SWOT Analysis; Supplier Analysis; Competitor Analysis.
- Business Strategies: Marketing; Financial, Production, Procurement, IT, Human Resources; Implementation Plan:- Activity Timeframes, Budgets; Monitoring and Control Plan

l) **Assessment method:**

Assessment will be in terms of

i. Coursework (tests and practical exercises) (40 %)
ii. Final examination (60%)

m) **Reading List**:

1. Bruce R. Barringer & R. Duane Ireland (2006). Entrepreneurship: Successfully Launching New ventures. Published by Pearson-Prentice Hall. 1/e Edition. ISBN 0-13-061855-1
2. Thoma W. Zimmerer and Norman M. Scarborough (2005) Essentials of Entrepreneurship and Small Business Management. 4th Ed. ISBN 0-13-191856-7
3. Kumar, S. (2003) Entrepreneurship Development. New Age International publication
4. Thomas, W., & Scarborough, N.M. (2004) Effective small business management: An entrepreneurial approach, Prentice Hall International, New Delhi, India
5. Wickham, P.A (2004) Strategic Entrepreneurship. 3rd Ed, London Pitman Publishing

- **Year 4 Semester II**
  - **BSE 4200: Software Engineering Project II**

- **Description**

The course covers practical aspects of the later phases of a software engineering. It ensures practical application of skills in implementation, testing and deployment

- **Course Objectives**

Upon successful completion of this course the student will have ability to:

- Demonstrate independent skills in implementing non-trivial software engineering/ research projects by pursuing a lengthy Software engineering project; and
- Demonstrate skills of Documenting, deploying a testing a well engineered solution, with assistance of one of the Professors/ Lecturers as adviser/ supervisor.

- **Course Content**

The student implements, documents, tests and deploys a software solution using the state of the art principles, concepts and technologies

The specific deliverables are:

- Software Implementation using state of the art technologies
- Detailed software documentation in accordance to well known practices
- Installation Manuals
- User Manual
- Testing and validation strategy.

- **Assessment Method**

Project report and Oral presentation (100%)

- **References**

No particular reference will be used for this course unit, except whatever is recommended by the student supervisors.

- **BSE 4201: Software Design Patterns**

Pre-requisites: Knowledge of UML

1. **Description**

Design patterns are standard solutions to common software design problems. Instead of focusing on how individual components work, design patterns are a systematic approach that focus and describe abstract systems of interaction between classes, objects, and communication flow. This course explores advanced principles of object-oriented design by studying key software design patterns.

2. Aims

This course teaches participants the correct way to create an object-oriented design by teaching them design patterns. The old style of looking for nouns and verbs to design your classes actually leads to brittle structures. After taking this class, attendees will understand what good design is as well as how to achieve it.

3. Learning outcomes

Applying design patterns at different levels of software design, implementation, and deployment using the different views of software systems.

- **Teaching & Learning patterns**

Assignments, class presentations, tests

- **Indicative content**

- Basic Patterns: Structural patterns Creational patterns
- Behavioral Grasp and Cooper, Vlissides Pattern Hatching
- Relationships between Patterns: Prees Metapatterns Zimmers Relations. Tichys classification, Classification in
- Automation of Patterns: Automation of Design Patterns Together, OpenJava and Design Patterns, Compost/Recoder Refactoring Copliens Symmetries
- Historic Roots: The timeless way of building
- Advanced Patterns: Parallelism Patterns Exclusion, State Dependence, PLOP2, Coordination, Reactive pattern,s Analysis Patterns Re engineering Patterns, Automation of Design Patterns, Process Patterns Organizational Patterns
- Applications of Design Patterns: Extreme Programming Cope: Multi-Paradigm Design

- **Assessment Method**

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

- **References**

- Erich Gamma et al. Design Patterns
- Wolfgang Pree. Design Patterns for Object-Oriented Software Development
- Frank Buschmann, Kevlin Henney, Douglas C. Schmidt On Patterns and Pattern Languages
- Erich Gamma, Richard Helm, Ralph Johnson, John M. Vlissides , Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley Professional; 1 edition (November 10, 1994)
- Elisabeth Freeman, Eric Freeman, Bert Bates, Kathy Sierra , Head First Design Patterns, O'Reilly Media; 1 edition (November 1, 2004)

▪ BSE 4202: Software Security

- **Description**

The course addresses the common security problems in software as well as their underlying causes. It then addresses the techniques, guidelines, principles and tools that prevent or detect them.

- **Aims**

- Explore the common security problems in software
- Explore the main causes of the security problems in software
- Explore the techniques of detecting security flaws in software during implementation

- **Learning outcomes**
  Mastery of security techniques in developing complex software systems
- **Teaching & Learning patterns**
  Teaching will be in terms of lectures, class presentations and practical demonstrations

e. Indicative content

- Phishing, Passwords, Difficulties with Reliable Password Entry, Difficulties with Remembering the Password, Naive Password Choice, Social-Engineering Attacks, Phishing Countermeasures, Password Manglers,
- Using the Browser's Password Database , Microsoft Passport, Two-Factor Authentication, Trusted Computing, Two-Channel Authentication,
- System Issues, Attacks on Password Entry, Interface Design, Eavesdropping, Technical Defeats of Password Retry Counters, Attacks on Password Storage, One-Way Encryption
- Password Cracking, CAPTCHAs, **Protocols,** Chosen Protocol Attacks ,Managing Encryption Keys, Basic Key Management, The Needham-Schroeder Protocol, Kerberos, Practical Key Management, **Access Control,** Operating System Access Controls, Groups and Roles, Access Control Lists, Unix Operating System Security,
- Windows—Basic Architecture, Capabilities, Windows—Added Features, Middleware, Database Access Controls,
- Sandboxing and Proof-Carrying Code, Virtualization, Trusted Computing, Cryptograph

f.Assessment Method

Course work (Tests - 20%, group assignment (Take home assignments, case studies, individual / group projects – 20%): 40%
Final examination (60%)

g.References

- *Ross Andersen* Security Engineering: *A guide to building dependable distributed systems 2nd Ed, Wiley*
- *Mark G. Graff and Kenneth R. van Wyk* Secure Coding: *Principles & Practices, O'Reilly, 2003.*

- **BSE 2209: IT Law and Ethics**

Course description: This course gives in-depth knowledge of computing law and ethics topics introduced earlier in pervasive themes of Information Technology/ Information Systems. It is intended for IT/IS undergraduate students who generally have little (or none) work experience in the IT industry. The course therefore provides a survival kit for IT/IS graduates entering the work force.

Aims:

The aims of this course are to:

- ⌀ Develop in-depth knowledge and understanding of Computer Law and Ethics major issues;
- ⌀ Provide the students with the survival legal kit necessary for their future carrier as IT/IS-professionals and;
- ⌀ Form an adoption of IT/IS professional ethics/code of conduct.

Learning outcomes:

At the end of the course, students will have knowledge in;

- ⌀ Information privacy, accuracy, property and accessibility
- ⌀ Computer Crime
- ⌀ Cyber-war or cyber-terrorism
- ⌀ Professional Ethical issues
- ⌀ Legal protection of IT/IS development -non-disclosure agreements,
- ⌀ Employment contracts
- ⌀ Intellectual property law (copyright, patent, licensing, and royalties),
- ⌀ Trademarks

- ⌀ Warranty disclaimers.

Teaching and Learning patterns:
This course is supposed to have only lecture hours. But within the 45 lecture hours, students will be offered a range of experiences that include:

- ⌀ Large group lectures
- ⌀ Working with team members in small groups (small group tutorials)
- ⌀ Assignments
- ⌀ Oral presentations
- ⌀ Self-assessment and peer assessment
- ⌀ Electronic discussion forum.

Indicative content:

- ⌀ First, the course considers the international and national legal frameworks related to IT/IS.
- ⌀ The course covers issues regarding information privacy, accuracy, property and accessibility.
- ⌀ A variety of computer crimes as far as responsibility for committing such crimes are discussed.
- ⌀ The course also considers computer ethical issues, such as information privacy, computer crime, computer terrorism, identity theft, ethical hacking etc. Each issue is analyzed in terms of the major ethical theories.
- ⌀ The course discusses professional ethics of IT societies (professional codes of conduct).
- ⌀ Other issues include; legal protection of IT/IS development -non-disclosure agreements, employment contracts, intellectual property law (copyright, patent, licensing, and royalties), trademarks and warranty disclaimers.

Assessment:
Course work (Tests (20%) and practical exercises (20%)
Final written exam: 60%

References:
Joseph Migga Kizza, 2003. Ethical and Social Issues in the Informational Age (2nd edition). Publisher: Springer-Verlag, New York, Inc.
Patrick Quirk and Jay Forder. 2003. Electronic Commerce and the Law (2nd edition). Publisher: John Wiley & Sons Australia..
Lawrence, E., Corbitt, B., Fisher, J., Lawrence, J., Tidwell, A. 2000. Internet Commerce-Digital Models for Business (2nd Edition). Publisher: John Wiley and Sons Australia
Johnson, D.G. 1994. Computer Ethics (2nd Edition). Publisher: Prentice Hall. ISBN: 0-13-290339-3. Later ed. preferable.
Knight, P and Fitzsimmons, J. 1990. The Legal Environment of Computing. Publisher: Addison- Wesley. Later ed. preferable.

Pre-requisites
This course should be given in the latter years of the programme i.e. 4th year after students have covered the business opportunities available in Software Engineering as a discipline

- **Resources and infrastructure**

The College of Computing and Information Sciences and specifically the Department of Networks have sufficient resources and infrastructure to suitably run the program as further described below.

7.1 Source of Funds

Fees payable by the students will enable the University to sustain the programme.

7.2. Staff

The Department Networks (see list of staff members in Appendix A) in Conjunction with other departments in the College have an adequate number of staff who can competently teach the courses.

7.3. Lecture Space

The College of Computing and Information Sciences is housed on 2,500 and 12,000 square meter buildings known as Block A and B, respectively. Block A mainly accommodates offices and a few laboratories, while Block B has lecture rooms together with the rest of the general and specialized laboratories. The two buildings sufficiently cater for all the lecture and lab space requirements for all the teaching in the faculty. Specifically CIT has 6 lecture theatres each of 400 square meters (600seat capacity); 6 small lecture theatres of total area 1200 square metres and 1800 square metres of circulation space where students are able to access other services such as wireless internet services.

7.4 Computer Laboratories and Software

The College buildings i.e. Block A and B respectively, have general laboratories (for student practice), teaching laboratories and specialized laboratories, that are shared among the four departments. At present, these laboratories have in total approximately 2000 computers. In summary, CIT has **got** 6 computer laboratories each of 800 square meters (1000 seat capacity) and 6 small laboratories of total area 1200 square meters. More lab details can be found on the Faculty website: http://cit.ac.ug/cit/facilities/labs.php.

In addition to the physical computers, different software is installed for usage by students depending on their focus. Most of the software is available as free distributions for academic purposes. The faculty and department therefore have (and can access) enough software that can run the practical aspects of the program.

7.5 Library services

Makerere University Library supports the College Computing and Information Science Library which is located on the First level (Block B Building). The College Library is stocked with up-to-date information resources. The information resources in the College Library have been acquired through purchases made by Makerere University Library and the Faculty. In addition to this facility, the University Library provides access to print books, print journals, electronic journal databases, a well-stocked reference section and connections to many remote databases like the Uganda Scholarly Digital Library at http://dspce3.mak.ac.ug. The print collection is beefed up by the broad variety of electronic resources provided by the University Library and accessible online at http://muklib.mak.ac.ug. Through the Document Delivery Service which is provided by the University Library, users who fail to get access to full-text articles from the available databases can make requests for the articles and delivered to them at no cost. Library users can also access the Online Public Access Catalogue (OPAC) to get bibliographic information about the collections found in the College Library at http://196.43.133.123:8080.

- **Quality Assurances**

Several activities will be carried out as quality assurance measures so as to:
a. Measure the general extent to which the required skills have been achieved
b. Ascertain the Implementation of the methodological changes proposed
c. Create a feed back bench marks for possible future revisions in the curriculum
The following activities will be carried out in the process of monitoring and assuring quality in the program.

8.1 Feedback from students enrolled

In the current set up, each class has 1 student representative. These representatives are in constant contact with the Head of Department in case there are any quality related matters in a particular class. This set up is to be maintained.

In addition, at the end of each semester, samples of students from respective classes/years are given questionnaires to respond to several quality related matters like staff punctuality, delivery mode, course content and the general perceived usefulness of the course unit.

We note that the College has developed a computerized system that will capture and analyze the data collected from the students. With the computerized system:

- ⌀ Every student is required to assess every lecturer teaching him/her, the sample space is therefore increased
- ⌀ No time is required in the analysis of the results. Staff and college management are able to get the feedback instantly
- ⌀ Data is easily archived and therefore the trend of staff performance in the respective areas is easy to visualize

8.2. Class meetings

The College management makes at least 2 meetings with every class every semester. In this meeting, general quality issues are addressed. Students are also given a chance to raise any questions that are answered and/or addressed by the department management. This set up will also continue.

8.3 . Use of ICT in availing lecture materials

Currently, Makerere University has the Muele e-learning tool on its Intranet. Students in the Department of Networks have adequate access to computers. This creates conducive environments for e-learning blended teaching. All courses in the new curriculum will be taught in a blended way. All course materials will be put on Muele.

Staff will, as much as possible, make use of e-learning facilities like discussion forum and drop boxes for assignments. This will increase student activity/participation and reduce staff effort (e.g. staff will not need to dictate notes). This will result to increase in the material covered and taken in by the students.

8.4. Peer review

Course leaders and other departmental staff will enroll (as students) to all classes taught in the department. They will therefore be able to view contents of courses taught by their peers. Course leaders will advise fellow staff on the content, depth and presentation of materials. Consequently, for every course, students will access the best material provided on the online platform which is also viewed by all staff in the department. But the course instructor shall be excluded in this view.

8.5. External examiners' reports

Like it is everywhere in Makerere University, students' exams are reviewed by senior external examiners. This is to bring a 'foreign view' of the quality of the examination. External examiners write reports on their view of the curriculum and examinations. Recommendations for the students about their exams can be implemented immediately or in a longer term. The department will make the maximum possible use of external examiners' reports as a means of assuring quality in the program.

8.6. Tracer studies

The College is devising ways of keeping in contact with its alumni together with their employers. This is with a view of making a tracer study of its graduates. The Department of Information Technology will use outputs of the tracer studies to gauge the quality of the program and whenever necessary improve it.

- ***Appendix A: Department of Networks Staff list***

	Name	Qualification	Rank	Areas of Specialization
1	J.S. Sansa Otim	PhD	L	i. Network Policies ii. Network Protocols
2	B. Kanagwa	PhD	SL	a) Design Patterns b) Service Oriented Architecture
3	T. Bulega	PhD	L	• Fiber Optics Networks • Data Communications
4	Joseph Balikuddembe	PhD	L	1. Value-Based Software Engineering 2. Project Management Strategy
5	A.Namulindwa	MSc.	AL	a. Software Engineering b. Formal Methods
6	D.P.Mirembe	MSc.	AL	1. Wireless Security 2. Mobile networks
7	F.N. Kiwanuka	MSc.	AL	1. Mobile networks 2. Software Security
8	B.S. Muwonge	MSc.	AL	Data Communication
9	R. Mbabazi	MSc.	AL	Data Communication
10	M. Ntanda	MSc.	AL	Requirements Engineering Internet Programming Embedded Systems
11	J. E. Agaba	MSc.	AL	Object-Oriented Programming Software Maintanance
12	M. Nsabagwa	MSc.	AL	Network Applications development
13	G. Kamulegeya	MSc.	AL	Mobile Computing Software Security Design Patterns
14	David Bamutura	MSc.	AL	Distributed systems development
15	Perez Matsiko	MSc.	AL	Data Communications Networkds
16	Odong Steven	MSc.	AL	User Interface Design
17	R. Namisanvu	BSc.	TA	Numerical Analysis
18	N.Kimbugwe	BSc.	TA	Systems Software
19	Bridget Magoba	Bsc.	TA	Software Engineering