

MAKERERE UNIVERSITY  
Faculty of Computing & Information  
Technology  
Department of Computer Science

Bachelor of  
Science in Computer Science degree programme

DAY/EVENING PROGRAMME

September 2009

# Contents

<b>1</b>	<b>Background</b>	<b>5</b>
1.1	The B.Sc Computer Science Program . . . . .	5
<b>2</b>	<b>The Program</b>	<b>5</b>
2.1	Nature of the Program . . . . .	5
2.2	Duration . . . . .	5
2.3	Tuition Fees . . . . .	5
2.4	Target Group . . . . .	5
<b>3</b>	<b>Regulations</b>	<b>6</b>
3.1	Admission requirements . . . . .	6
3.2	Weighting . . . . .	7
3.3	Semester Load and Minimum Graduation Load . . . . .	7
3.4	Assessment . . . . .	7
3.5	Grading and Pass mark . . . . .	7
3.6	Progression . . . . .	8
3.7	Retaking a course . . . . .	8
3.8	Degree Classification . . . . .	8
3.9	Discontinuation . . . . .	9
<b>4</b>	<b>The Proposed Curriculum</b>	<b>9</b>
4.1	Course outline . . . . .	9
4.1.1	Semester 1 . . . . .	9
4.1.2	Semester 2 . . . . .	9
4.1.3	Year I Recess term . . . . .	9
4.1.4	Semester 3 . . . . .	10
4.1.5	Semester 4 . . . . .	10
4.1.6	Year 2 Recess Term . . . . .	10
4.1.7	Semester 5 . . . . .	11
4.1.8	Semester 6 . . . . .	11
4.2	Knowledge Areas Covered in the Curriculum . . . . .	11
4.3	Content Distribution by Knowledge Area . . . . .	12
<b>5</b>	<b>Detailed Curriculum</b>	<b>13</b>
5.1	Semester 1 . . . . .	13
5.1.1	CSC 1100: Computer Literacy . . . . .	13
5.1.2	CSK 1101: Communication Skills . . . . .	14
5.1.3	CSC 1104: Computer Organization & Architecture . . . . .	15
5.1.4	CSC 1105: Numerical Methods . . . . .	17

5.1.5	CSC 1106: Programming Methodology I . . . . .	18
5.2	Semester 2 . . . . .	20
5.2.1	CSC 1204: Research Methodology . . . . .	20
5.2.2	CSC 1206: Computational Mathematics . . . . .	22
5.2.3	CSC 1207: Programming Methodology II . . . . .	23
5.2.4	CSC 1208: Individual Project I . . . . .	24
5.2.5	CSC 1209 Logic Programming . . . . .	25
5.3	Year I Recess Term . . . . .	27
5.3.1	CSC 1301: Practical Skills Development . . . . .	27
5.3.2	CSC 1303: Cisco Certified Network Associate (Au- dited) . . . . .	27
5.4	Semester 3 . . . . .	27
5.4.1	CSC 2100: Data Structures and Algorithms . . . . .	27
5.4.2	BSE 2105 Formal Methods . . . . .	29
5.4.3	CSC 2109: Discrete Mathematics . . . . .	30
5.4.4	CSC 2111: Database Management Systems I . . . . .	31
5.4.5	CSC 2113: Software Engineering . . . . .	32
5.4.6	CSC 2114 Artificial Intelligence . . . . .	35
5.5	Semester 4 . . . . .	37
5.5.1	CSC 2200: Operating Systems . . . . .	37
5.5.2	BSE 2203 Computer Networks & Data Communication	39
5.5.3	CSC 2209: Systems Programming . . . . .	40
5.5.4	CSC 2210 Automata, Complexity and Computability .	41
5.5.5	CSC 2212 Individual Project II . . . . .	42
5.5.6	BSE 2201 Network Applications Development . . . . .	43
5.5.7	CSC 2214 Cryptology and Coding Theory . . . . .	44
5.6	Year 2 Recess Term . . . . .	46
5.6.1	CSC 2301 Industrial Training . . . . .	46
5.7	Semester 5 . . . . .	46
5.7.1	CSC 3103: User Interface Design . . . . .	46
5.7.2	CSC 3110 Database Management Systems II . . . . .	48
5.7.3	CSC 3111 Operations Research . . . . .	49
5.7.4	CSC 3112: Principles of Programming Languages . . .	50
5.7.5	CSC 3113 Emerging Trends in Computer Science . . .	51
5.7.6	BIS 3100: Modeling and Simulation . . . . .	52
5.7.7	CSC 3105 Computer Graphics . . . . .	53
5.7.8	CSC 3114 Selected Topics in Computer Science . . . .	54
5.7.9	CSC 3115 Advanced Programming . . . . .	55
5.7.10	BIT 3102 Entrepreneurship and Business . . . . .	56
5.8	Semester 6 . . . . .	58
5.8.1	BIT 3204 Enterprise Network Management . . . . .	58

5.8.2	CSC 3205 Compiler Design . . . . .	59
5.8.3	CSC 3206 Group Project . . . . .	61
5.8.4	CSC 3207 Computer Security . . . . .	62
5.8.5	BIT 3200 Business Intelligence and Data Warehousing	63
5.8.6	BSE 3202: Distributed Systems Development . . . . .	64
<b>6</b>	<b>Resources and Infrastructure</b>	<b>65</b>
6.1	Funds . . . . .	65
6.2	Staff . . . . .	65
6.3	Lecture Space . . . . .	65
6.4	Computer Laboratories . . . . .	65
6.5	Software . . . . .	66
6.6	Library Services . . . . .	66
<b>7</b>	<b>Quality Assurance</b>	<b>66</b>
7.1	Feedback from students enrolled . . . . .	66
7.2	Class meetings . . . . .	67
7.3	Use of ICT in availing lecture materials . . . . .	67
7.4	Peer review . . . . .	67
7.5	External examiners' reports . . . . .	68
7.6	Industrial Training placement reports . . . . .	68
7.7	Tracer studies . . . . .	68

# **1 Background**

## **1.1 The B.Sc Computer Science Program**

The Bachelor of Science in Computer Science is a day/evening full time programme. It targets A'level leavers and Diploma holders. The objectives of the program are to:

1. Develop professionals with theoretical and practical skills in Computer Science;
2. Strengthen capacity and institutional building in Computer Science in tertiary institutions, the private and public sector;
3. Build capacity with a practical orientation needed to link up the Computer Science sector with Government and Industry under the broader perspective of Information and Communication Technology (ICT) and
4. Provide most of the ICT professionals needed in Uganda and neighboring countries.

# **2 The Program**

## **2.1 Nature of the Program**

The program is full time that is conducted both in the day and in the evening.

## **2.2 Duration**

The duration of the program shall be three academic years consisting of six semesters and two recess terms. Each semester lasts seventeen (17) weeks two of which are for examinations. Each recess term is 10 weeks.

## **2.3 Tuition Fees**

Tuition fees for privately sponsored students shall be 2,520,000 Uganda Shillings per year for Ugandans and 3,780,000 Ugandan Shillings per year for international students.

## **2.4 Target Group**

The programme targets two categories of people. These are A' level leavers and diploma holders in relevant disciplines.

## 3 Regulations

### 3.1 Admission requirements

To be admitted to the B.Sc (Computer Science) program, a candidate must satisfy the general admission requirements for Makerere University. In addition, the following regulations shall hold:

a Direct Entry

Candidates seeking admission through this avenue must have obtained:-

- At least a subsidiary pass in Mathematics in the Uganda Advanced Certificate of Education (UACE) or its equivalent
- At least two principle passes at the same sitting in UACE in any of the following subjects: - Mathematics, Economics, Entrepreneurship, Geography, Physics, Chemistry, Biology, Agriculture, Technical Drawing and Food & Nutrition.
- A minimum weighted point set by the Makerere University Admissions Board.

For purposes of computing weighted points, the A' level subjects shall be grouped and weighted as follows:-

Group	Weight	Subjects
Essential	3	Any two best done of the above subjects
Relevant	2	The third done of the above subjects
Desirable	1	General Paper, Subsidiary Mathematics
Others	$\frac{1}{2}$	All others

b Diploma holders:

For a candidate to be admitted via the diploma scheme, he/she must:

- Have at least 5 passes got at the same sitting of Uganda Certificate of Education or its equivalent
- Have at least 1 principle pass and 2 subsidiary passes from the same sitting of the Uganda Advanced Certificate of Education (UACE) or its equivalent
- Have a Diploma in Computer Science and Information Technology of Makerere University <sup>1</sup> or have at least a second class (lower di-

---

<sup>1</sup>Students with at least a second class (lower division) class enroll on the second year of the program. Those with third class start in first year

vision) diploma in Computer Science, Engineering, Business Studies, Information Technology, Statistics or any other diploma with Mathematics, Computer Science or Information Technology as one of the subjects. The diploma must be from an Institution recognised by the National Council for Higher Education (of Uganda).

### **3.2 Weighting**

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 (LH), (ii) two practical hours (PH) or (iii) two tutorial hours (TH)

### **3.3 Semester Load and Minimum Graduation Load**

The normal semester load is between 16 and 24 credit units. The minimum graduation load is 127 credit units of which 111 credit units are from core course units. The remaining 16 credit units are to be got from elective course units in semesters where elective courses are offered.

### **3.4 Assessment**

Assessment is to be done by progressive assessments (like tests, assignments, group work) during the semester and final examination. The final examination may be purely written, purely practical or having a written and practical component. Progressive assessments constitute 40% of the final score and the final examination will constitute 60%.

### **3.5 Grading and Pass mark**

Grading will be based on the final score for each examination using the ranges bellow

Marks	Letter Grade	Grade Point	
90-100	A+	5	Exceptional
80- 89	A	5	Excellent
75- 79	B+	4.5	Very good
70- 74	B	4	Good
65- 69	C+	3.5	Fairly good
60- 64	C	3	Pass
55- 59	D+	2.5	Marginal fail
50- 54	D	2	Clear fail
45- 49	E	1.5	Bad fail
40- 44	E-	1	Qualified fail
0 - 39	F	0	Qualified fail

A student with a grade point greater or equal to 2 (letter grade D) in a certain course unit is considered to have passed the course unit.

### 3.6 Progression

A student is considered to be under normal progression if he/she has a grade point of at least 2 in all the courses that make up his/her full semester load. A student is under probational progression if he/she has at least a course unit in his/her full semester load where the grade point is less than 2.

### 3.7 Retaking a course

A student will retake every course from which he/she obtains a grade point less than 2

### 3.8 Degree Classification

The degree will be classified using the cumulative grade point average (CGPA)

$$CGPA = \left( \sum_{i=1}^{i=n} GP_i \times CU_i \right) \div \left( \sum_{i=1}^{i=n} CU_i \right)$$

where GP and CU represent grade points and Credit units respectively.

The degree will be classified in accordance with the table below

CLASS	CGPA
First Class	4.40 - 5.00
Second Class - Upper Division	3.60 - 4.39
Second Class - Lower Division	2.80 - 3.59
Pass	2.00 - 2.79

### 3.9 Discontinuation

A student will be discontinued from the program if

- Fails to complete the program in five years
- Fails a course unit three times
- Has a CGPA of less than 2 for three consecutive semesters

## 4 The Proposed Curriculum

### 4.1 Course outline

#### 4.1.1 Semester 1

Code	Name	CU	LH	PH	TH	CH
CSC 1100	Computer Literacy	4	30	60	–	60
CSK 1101	Communication Skills	4	45	30	–	60
CSC 1104	Computer Organization & Architecture	4	60	–	–	60
CSC 1105	Numerical Methods	3	45	–	–	45
CSC 1106	Programming Methodology I	3	30	30	–	45
	Total	18				

#### 4.1.2 Semester 2

Code	Name	CU	LH	PH	TH	CH
CSC 1204	Research Methodology	3	30	–	30	45
CSC 1206	Computational Mathematics	4	45	–	30	60
CSC 1207	Programming Methodology II	4	30	60	–	45
CSC 1208	Individual Project I	4	15	90	–	60
CSC 1209	Logic Programming	3	30	30	–	45
	Total	18				

#### 4.1.3 Year I Recess term

Code	Name	CU	LH	PH	TH	CH
CSC 1301	Practical Skills Development	4	–	120	–	60
CSC 1303	Cisco Certified Network Associate (Audited)	5	150	100	–	200
	Total	4				

#### 4.1.4 Semester 3

Code	Name	CU	LH	PH	TH	CH
CSC 2100	Data Structures and Algorithms	4	45	–	30	60
BSE 2105	Formal Methods	4	45	–	30	60
CSC 2109	Discrete Mathematics	3	30	–	30	45
CSC 2111	Database Management Systems I	3	30	30	–	45
CSC 2113	Software Engineering	4	45	–	30	60
CSC 2114	Artificial Intelligence	3	30	–	30	45
	Total	21				

#### 4.1.5 Semester 4

Code	Name	CU	LH	PH	TH	CH
CSC 2200	Operating Systems	4	45	–	30	60
BSE 2203	Computer Networks & Data Communication	4	45	30	–	60
CSC 2209	Systems Programming	4	45	–	30	60
CSC 2210	Automata, Complexity & Computability	3	45	–	–	45
CSC 2212	Individual Project II	3	–	90	–	45
1 elective						
BSE 2201	Network Applications Development	3	30	30	–	45
CSC 2214	Cryptology and Coding Theory	3	45	–	–	45
	Total	21				

#### 4.1.6 Year 2 Recess Term

Code	Name	CU	LH	PH	TH	CH
CSC 2301	Industrial Training	4	–	120	–	60
	Total	4				

#### 4.1.7 Semester 5

Code	Name	CU	LH	PH	TH	CH
CSC 3103	User Interface Design	4	30	–	60	60
BIT 3102	Enterprenuership and Business	3	30	–	30	45
CSC 3111	Operations Research	3	30	–	30	45
CSC 3112	Principles of Programming Languages	3	45	–	–	45
CSC 3113	Emerging Trends in Computer Science	4	15	90	–	60
2 electives						
CSC 3110	Database Management Systems II	3	45	–	–	45
BIS 3100	Modeling and Simulation	4	45	–	30	60
CSC 3105	Computer Graphics	3	30	–	30	45
CSC 3114	Selected Topics in Computer Science	3	45	–	–	45
CSC 3115	Advanced Programming	3	30	–	30	45
	Total	23				

#### 4.1.8 Semester 6

Code	Name	CU	LH	PH	TH	CH
BIT 3204	Enterprise Network Management	4	45	–	30	60
CSC 3205	Compiler Design	3	45	–	–	45
CSC 3206	Group Project	5	–	135	–	60
2 electives						
CSC 3207	Computer Security	3	45	–	–	45
BIT 3200	Business Intelligence & Data Warehousing	4	45	30	–	60
BSE 3202	Distributed Systems Development	4	45	30	–	60
	Total	15				

## 4.2 Knowledge Areas Covered in the Curriculum

The curriculum is based on 8 broad knowledge areas. These are:-

1. Mathematical Foundations of Computer Science (MFCS)
2. Theoretical Foundations of Computer Science (TFCS)
3. Data Structures, Algorithms and Programming (DSA&P)
4. Organizational Aspects of Software Development (OASD)
5. Study and Optimization of Operational Systems (SOOS)
6. Computer Networks (CN)

7. Research and Development (R&D)

8. Soft Skills (SS)

Graduates from the program are expected to exhibit competent knowledge/skills in all the subject areas above. Each of them, therefore, has to be adequately covered so as to produce good quality graduates. Analysis of the coverage for the different subject areas is done in Section 4.3.

### 4.3 Content Distribution by Knowledge Area

The table below summarizes the distribution of the different course units in the different knowledge areas. The total number of credit units is also summarized.

Knowledge Area								
Sm	DSA&P	OASD	SOOS	CN	R&D	MFCS	TFCS	SS
I	CSC 1106		CSC 1106			CSC 1105	CSC 1104	CSK 1101 CSC 1100
II	CSC 1207 CSC 1209		CSC 1207		CSC 1204 CSC 1208	CSC 1206		
rtI				CSC 1303	CSC 1301			
III	CSC 2100	CSC 2111 CSC 2113 BSE 2105 CSC 2114				CSC 2109		
IV	CSC 2209	BSE 2201 BSE 2203		BSE 2203	CSC 2212		CSC 2200 CSC 2210 CSC 2214	
rtII					CSC 2301			
V	CSC 3115	CSC 3103 CSC 3110	CSC 3111 BIS 3100		CSC 3113	CSC 3111	CSC 3110 CSC 3105 CSC 3112	
VI		BIT 3200 BSE 3202		BIT 3204	CSC 3206		CSC 3207 CSC 3205	
$\Sigma$	21	32	14	16	26	13	29	8

NB: Credits from audited courses are also included

From the table above, the largest proportion of the contact hours are in TFCS, OASD, R& D and DSA&P. This is aimed at having students with (i) hands on skills (ii) ability to research and learn new subject matter on their own and (iii) ability to undertake advanced studies.

## 5 Detailed Curriculum

### 5.1 Semester 1

#### 5.1.1 CSC 1100: Computer Literacy

(a) 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.

(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 and learning pattern

Teaching will be by lectures and laboratory demonstrations/practicals

(e) Indicative content

- General computer organization
- Historical perspectives of computing

- Common Microsoft office packages
  - Office packages in other operating systems
  - Text editors
  - Common Linux commands
  - Using the web
- (f) Assessment method  
The assessment will be in form of tests and assignments (40%) and final written exam (60%)
- (g) Reading list
- (i) Computer Literacy by John Preston, Robert Ferrett and Shelly Gaskin, 2007.
  - (ii) Practical Computer Literacy by Jelne Janrich and Dan Oja, 2001

### 5.1.2 CSK 1101: Communication Skills

- (a) Description  
This course provides students with skills of effective communication. These include verbal, written, and gestural. The course aims at facilitating students appropriately and clearly communicate with others.
- (b) Aims  
The aims of the course are:
- 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) Teaching and Learning Pattern  
Teaching and Learning will be by classroom lectures, demonstration and students practical projects
- (d) Indicative Content

- Writing Skills:  
Thinking critically/ selectively before the writing process; Selecting the relevant details; Organizing 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 e.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 process; Writing examinations and following instructions.

(e) Assessment Method

Assessment will be in form of coursework and tests (40%) and written examination (60%)

(f) Reading List

- (i) 101 ways to improve your communication skills instantly, by Benjie Bough, 4th Edition, 2005
- (ii) The hard Truth About soft skills: Work Place Lessons Smart People wish they had learned sooner, by Peggy Klavs, 2008

### 5.1.3 CSC 1104: Computer Organization & Architecture

(a) Description

This course introduces the logical architecture and organization of computer systems. It highlights the lower end operations in a typical com-

puter as well as the way computers manage their resources during operation. The course opens up a student to be an informed user of the computer rather than a passive recipient of the computer services.

(b) Aims

The aims of the course are:

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

(c) Learning Outcomes

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

(d) Teaching and Learning Pattern

Teaching will be in terms of lectures as well as tutorials

(e) 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.
- Micro Computer Architecture:  
CPU, Memory, I/O Devices and Interfaces, System Bus, Examples of CPU Structures, The Intel / Pentium CPU, The Z-80 or Motorola, Machine Language Instructions, Instruction Formats and Addressing Modes.
- The Processing Elements: Macroinstruction execution, Internal Bus Transfers, Detailed Internal Architecture, Microcontrol, Hard-wired Control, Microprogrammed Control, Reduced Instruction Set Computers.

- I/O Programming:  
Programmed I/O, Interrupt I/O, Polling, Priority Interrupt System, Direct Memory Access, I/O processors.
  - Memory Systems and Memory Management:  
Memory Hierarchy, Main Memory, Auxiliary Memory, Associative Memory, Cache Memory, Virtual Memory, Memory Management Hardware
- (f) Assessment method Assessment will be in form of tests and assignments (40%) and final examination (60%)
- (g) Reading list
- (i) Computer Systems Architecture by M. Morris Mano, Prentice Hall, 1993
  - (ii) Structured Computer Organization by Andrew S. Tanenbaum, Prentice Hall 1984.
  - (iii) Computer Systems Concepts and Design by Glenn B. Gibson, Prentice Hall, 1991
  - (iv) Computer Organization and Architecture by William Stallings, Prentice Hall 2003.

#### 5.1.4 CSC 1105: Numerical Methods

- (a) Description  
The course is to sharpen the students' skills in using numerical approaches to solve mathematical/real life problems. The focus will be on the ability to correctly formulate numerical problems and schemes that solve them. Emphasis will be put on the precision and robustness of the schemes.
- (b) Aims  
The aims of the course are
- To provide a solid basis on the numerical approaches to computational problem solving;
  - To provide students with problem analysis and solving skills to be able to handle typical computational problems in practice.
- (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 and Learning Pattern  
The course will be taught theoretically. The developed schemes can be implemented and run in any programming language.
- (e) Indicative Content
- Numerical solutions to non linear equations;
  - Numerical solutions for systems of linear equations;
  - Fast - Fourier transforms;
  - Numerical differentiation;
  - Numerical integration;
  - Numerical solutions for differential and difference equations.
- (f) Assessment method  
At least 2 (1 hour) tests and 1 assignment (40%) One 3-hour examination (60%)
- (g) Reading lists
- (i) Numerical Analysis by Richard L. Burden and J. Douglas Faires, Wardsworth, 1993.
  - (ii) A.K. Kaw, E.E.Kalu and D.Nguyen(2008) Numerical Methods with Applications

### 5.1.5 CSC 1106: Programming Methodology I

- (a) 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.
- (b) Aims  
The aims of the course are to provide the student with:

- 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.

(c) Teaching and Learning pattern

The course will be taught with a big practical component. Students will be expected to have one supervised practical sessions per week. They will so be given several programming assignments some of which will be marked and contribute to the coursework scores.

(d) 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
- GUI

(e) Assessment method

Assessment will be in form of at least one (practical) assignment and one test (40%), A practical exam - (30%) and a final written examination (30%)

(f) Reading list

- (i) C Programming Language by Brian W. Kernighan, Dennis M. Ritchie; Prentice Hall, 2000.
- (ii) C: A Reference Manual (5<sup>th</sup> Edition) by Samuel P. Harbison; Prentice Hall, 2002.

## 5.2 Semester 2

### 5.2.1 CSC 1204: Research Methodology

(a) Description

The purpose of this course is to acquaint students with types of scientific research relevant for anyone working in the field of computer science. It will enable students to develop capacity to conduct small, simple research projects while at the university.

(b) Aims

The aims of this course unit are to:

- Enable students become competent in understanding the research process;
- Provide skills that will enable students undertake independent research using a variety of appropriate methods, using primary and secondary data, as well as qualitative and quantitative techniques;
- Provide students with skills to produce a research proposal;
- Highlight ethical research practices to students.

(c) Learning outcomes

By the end of the course, students will be:

- Capable in their chosen professional, vocational or study areas to conduct research;
- Able to contribute in an entrepreneurial and innovative way within their business, workplace or community in the field of research;
- Able to operate effectively and ethically in conducting research in groups/teams
- Adaptable and manage change to handle different research situations according to different contexts;
- Aware of research and research methodology in subsequent years of study.

(d) Intellectual, Practical and Transferable skills

At the end of the course, students should have the ability to demonstrate:

- Appreciation of the different functions and applications of scientific research in the field of computer science;

- Basic knowledge of the different research methodologies relevant for computer science;
- Knowledge of which methods to use in what circumstances;
- Knowledge of what a research proposal entails;
- Application of quantitative and qualitative research methods and techniques;
- Judgment of the quality of research proposals as well as the products (articles, papers, theses etc.) of scientific research.

(e) Teaching and Learning Pattern

Teaching will be in form of formal lectures, tutorials and seminars. Classes will be interactive and students are expected to come to class prepared to participate and contribute regularly to class activities and discussions.

(f) Indicative Content

The content of this course will include:

- Introduction to scientific research;
- Formulating and clarifying the research topic and research problem;
- Conducting a literature review;
- Different research approaches;
- Ethics in research;
- Sampling;
- Use of secondary data;
- Collection methods for primary data;
- Analyzing qualitative data;
- Analyzing quantitative data and writing a research proposal and project report.

(g) Assessment method

The course will be assessed by course work and tests (40%) and final examination (60%)

(h) 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.

### 5.2.2 CSC 1206: Computational Mathematics

(a) Description

The course gives the students a strong mathematical base to be able to tackle other computer problems. The course brings together mathematical topics which are 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) Teaching and learning pattern

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

(d) Indicative content

- Calculus
  - Sequences and limits
  - Limits of functions & continuity
  - Advanced differential calculus
- Statistics
  - Discrete Random variables and distributions
  - Continuous distributions
  - Basics of Monte Carlo simulations
- Further topics
  - Mathematics for image processing
  - Mathematics for signal processing

(e) Assessment method

The assessment will be by tests/assignment (40%) and final examination (60%)

(f) Reading list

- (i) Calculus and Analytic Geometry (9<sup>th</sup> Edition) by George B. Thomas, Ross L. Finney Addison Wesley, 1995.
- (ii) Introduction to Statistics by R. E Walpole, 3rd Ed, Prentice Hall, 1982

### 5.2.3 CSC 1207: Programming Methodology II

(a) Description

The course is to give an in depth understanding of Object Oriented programming. It is to cater for Object Oriented programming practices like inheritance, interfaces, exception handling, action handling, security, software reuse and robustness.

(b) Aims

The aim of the course is to

- Move the students' programming skills from basic to advanced
- Avail students with skills to handle non functional program aspects like robustness and security
- Train students to develop complete computer applications

(c) Teaching and learning pattern

This will include lectures, practicals and lab assignments

(d) Indicative content

- The object oriented paradigm
- Classes and objects
- Inheritance and visibility modifiers
- Interfaces and abstract classes
- Graphical user interface and action handlers
- Exception handling
- Working with files
- Working with databases
- Sessions and user management

(e) Assessment method

The assessment will be done by tests and take home assignments (40%), practical examination (30%) and written examination (30%)

(f) Reading list

- (i) Java Software Solution: Foundations of Program Design (6<sup>th</sup> Edition) by John Lewis and William Loftus, Addison-Wesley, 2009.
- (ii) A Programmer's Guide to Java<sup>TM</sup> Certification: A comprehensive Primer by Khalid A. Mughal and Rolf W. Rasmussen, Addison-Wesley, 1999.

#### 5.2.4 CSC 1208: Individual Project I

(a) Description

This course is to give students an experience of developing simple but complete applications. Focus will be put on programming and program documentations as well as realization of the objectives.

(b) Aims

The aims of the course are:

- To practically give students skills of integrating different concepts of programming into a single application.
- To introduce the student to hands on aspects of software development.
- To nurture the student's ability to independently read different sources of literature so as to identify what (s)he can use to develop his/her application.

(c) Teaching and Learning Pattern

Learning will be largely by self study/ research. Students will be given programming Assignments that will be submitted after a specific period of time. The skills which that will be expected in the assignment will be indicated to the student. A member of staff will meet students at the issue of each of the assignment and address any outstanding issues as well as expectations. A minimum of six assignments will be given.

(d) Indicative content

The main content in the course is translation of a real life problem into a working computer program.

(e) Assessment method

Two assignments (each taking two weeks) and three assignments (each taking 3 weeks) will be given. The first two assignments will constitute the coursework (40%) while the last three assignments will constitute the examination mark (60%).

(f) Reading lists

Students can read any literature like books and online tutorials that can help in addressing the problem in the project at hand.

### 5.2.5 CSC 1209 Logic Programming

(a) Description

This course introduces a paradigm where computation arises from proof search in a logic according to a fixed, predictable strategy. It thereby unifies logical specification and implementation in a way that is quite different from functional or imperative programming. This course provides a thorough, modern introduction to logic programming. It introduces the basic concepts and techniques of logic programming followed by successive refinement towards more efficient implementations or extensions to richer logical concepts. It covers a variety of logics and operational interpretations.

(b) Aims

The aim of the course is to provide a basic introduction to the logic programming language, Prolog. It aims at introducing a number of logical systems of importance in computer science.

(c) Learning outcomes

By the end of the subject, students should:

- Be conversant with the syntax and semantics of propositional and predicate logic
- Be familiar with a variety of applications of predicate logic in software verification, databases and knowledge-based systems
- Be able to write specifications in predicate logic expressing state constraints
- Understand the notion of formal proof, and be able to construct simple proofs in a natural deduction proof system for predicate logic
- Be aware that there are inherent expressiveness and computational limitations to the applicability of logical systems, and be familiar with a number of restrictions under which the computational limitations can be overcome
- Be able to write programs in a logic programming language,
- Understand both the top-down and the bottom up operational semantics of logic programs

- Be familiar with a logic for reasoning about sequential programs, and capable of constructing correctness proofs for simple programs
- (d) Teaching and Learning pattern
- The course consists of a traditional lecture component and a project component. The lecture component introduces the basic concepts and techniques of logic programming. The project component will be one or several projects related to logic programming.
- (e) Indicative content
- Introduction
  - Pure logic (relational) programming
  - The Prolog Language
  - Programming in Prolog.
  - Efficient Prolog Programming
  - Combining Logic Programming, Functional Programming, Higher Order, Objects.
  - Review of first order predicate logic and resolution.
  - Fundamental results.
  - Semantics of logic programs.
  - Implementation of logic languages and advanced compilation.
  - Parallelism, concurrency.
  - Other LP/CLP languages
- (f) Assessment method
- Assessment will be by assignments and/or tests (40%) and written examination (60%)
- (g) Reading list
- (i) Logic in Computer Science, Modeling and Reasoning about Systems, M.R. Huth and M.D. Ryan, Cambridge University Press 2000
  - (ii) SWI Prolog Home Page, <http://www.swi-prolog.org/>

## **5.3 Year I Recess Term**

### **5.3.1 CSC 1301: Practical Skills Development**

The course aims at imparting practical skills in areas chosen by the faculty. The students are to be supervised by staff within the faculty. Areas of practical skills development include

- Implementation of projects
- Network and system administration
- Hardware maintenance
- Computer assembly

This can be done within the Faculty of Computing and Information Technology or any other unit in Makerere University. Students will write a report at the end of the course.

### **5.3.2 CSC 1303: Cisco Certified Network Associate (Audited)**

In this course, students will cover the course content of the CCNA international curriculum.

## **5.4 Semester 3**

### **5.4.1 CSC 2100: Data Structures and Algorithms**

(a) 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.

(b) 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.
- (c) Teaching and Learning pattern  
The teaching pattern is by lecture, practical lab work, group discussion and class presentations.
- (d) 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).
- (e) Assessment method  
The assessment will be done by tests/assignment (40%) and final examination (60%)
- (f) Reading List
- (i) Data Structures and Algorithms by Alfred V. Aho, Jeffrey D. Ullman, John E. Hopcroft. Addison-Wesley, 1983
  - (ii) The Design and Analysis of Computer Algorithms by Alfred V. Aho, Addison-Wesley Longman, 1974
  - (iii) Introduction to Algorithms 2nd Ed by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein, McGraw-Hill, 2008

### 5.4.2 BSE 2105 Formal Methods

(a) Description

The course provides students with skills of solving generic formal problems in science. It covers the intellectual and practical skills necessary for problem formalization.

(b) Aims

The aims of the course are:

- To provide students with factual knowledge including the mathematical notations and terminologies used in formalizing scientific problems
- To provide students with fundamental principles including the laws and theorems arising from the concepts covered in this course;
- To be able to apply course material along with techniques and procedures to solve practical problems;
- To provide programming skills by writing numerical programs like Matlab programs, to solve numerical problems.

(c) Teaching and Learning Patterns

Teaching will be by Lectures and practical demonstrations

(d) Indicative Content

- Predicate Logic Specification:  
Foundations; Basic concepts; Verification; Z; Tools and systems; Z animation Miranda and ZANS; Nitpick and the Z Notation.
- Algebraic Specification:  
Foundations; Basic concepts; Verification; Tools and systems; Miranda; The OBJ family of languages; LARCH.

(e) Assessment Method:

Assessment will be in terms of tests and practical assignments (40%) and final written examination (60%)

(f) Reading list

- (i) Z: An Introduction to Formal Methods, by Antoni Diller, 2nd edition, Wiley, (June 1994), ISBN-10: 0471939730
- (ii) Logic in Computer Science: Modeling and Reasoning about Systems, by Michael Huth and Mark Ryan, Cambridge University Press; 2nd Edition (August, 2004), ISBN-10: 052154310X

- (iii) Formal Methods and Models for System Design: A System Level Perspective, by Gupta, R., Le Guernic, P. Shukla, S.K. and Talpin, J.P. (Eds.), 2004, ISBN: 978-1-4020-8051-7

### 5.4.3 CSC 2109: Discrete Mathematics

(a) 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) Teaching and Learning pattern

Teaching and learning will be by lectures and tutorials

(d) 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
- (e) Assessment method  
Assessment will be by assignments and/or tests (40%) and written examination (60%)
- (f) Reading list
- (i) Bobrow, L.S. and Arbib, M.A. Discrete Mathematics: Applied Algebra for Computer and Information Science. Philadelphia, PA: Saunders, 1974.
  - (ii) Dossey, J.A.; Otto, AD.; Spence, L.; and Eynden, C.V. Discrete Mathematics, 3rd ed. Reading, MA: Addison-Wesley, 1997.
  - (iii) Balakrishnan, V.K. Introductory Discrete Mathematics. New York: Dover, 1997.

#### 5.4.4 CSC 2111: Database Management Systems I

- (a) Description  
The course is provide students with a strong foundation in systematic approaches to design and implementation of database applications. Preliminarily operations like requirements gathering and database planning will be covered. The course will also introduce students to developing of application programs that talk to the database. These applications may be online or off line.
- (b) Aims  
The aims of the course are to:
- Provide a background for the evolution of database (management) systems
  - Provide the students with the steps one has to go through when developing good database applications
  - Give hand on experience and knowledge in developing database (driven) applications
- (c) Teaching and Learning patterns  
Teaching will be by lectures, take home reading assignments/class presentations and laboratory practicals
- (d) Indicative content

- Background to databases
  - Evolution of database systems
  - Database organization and architecture
  - Database models
  - The database development life cycle
  - Database design
  - Tuning of operational systems
  - Querying databases
  - SQL/PL SQL
  - Scripting
- (e) Assessment Methods  
Assessment will be in terms of tests and take home assignments (40%),  
A practical examination (30%) and a final written exam (30%)
- (f) Reading list
- (i) Thomas Connolly and Carolyn Begg: Database Systems: A Practical Approach to Design, Implementation and Management. 2nd Edition, Addison-Wesley, 2004.

#### 5.4.5 CSC 2113: Software Engineering

- (a) Description  
This course introduces students to the foundations of software engineering as a discipline. Students are introduced to the evolving role of software engineering, especially with emphasis on software engineering process and process models. Key topics covered include Software configuration management, Requirement analysis, Software Specification, Design methods, Software testing, Software project management techniques; Software project planning, Risk management; Software Quality Assurance; Software reuse; and Computer aided software engineering: CASE tools and application.
- (b) Aims and Objectives
- To introduce software engineering and to explain its importance
  - To set out the answers to key questions about software engineering

- To introduce ethical and professional issues and to explain why they are of concern to software engineers

(c) Learning outcomes

On successfully completing of this unit students will be able to

- Demonstrate competence in handling software engineering projects.
- Know the fundamental software engineering processes and models
- Know what is involved in a typical software engineering project's life cycle.
- Employ good project management principles in handling projects and know why these principles are important in constructing quality software.
- Be competent in using CASE tools in real world projects.

(d) Teaching and Learning Pattern

Teaching and learning is to be implemented through lecture, lab and tutorial sessions. Students are also expected to make presentations of their work.

(e) Indicative Content

- Evolving role of software, software characteristics; Systems and environment; system engineering hierarchy, information and knowledge engineering; Information strategy; Business Area analysis, modeling enterprise and business-level data modeling, system architecture and associated information flow; writing system specification.
- Software Engineering as a layered technology: Software process, software process models. Software configuration management: the SCM process, Identification of objects in software configuration, version control, change control, configuration audit, SCM standards.
- Requirement analysis: Communication techniques, Information gathering tools; organizing and structuring information; analysis principles; Analysis modeling.
- Software Specification: Design process, principles and concepts: Abstraction, refinement, modularity, control hierarchy, structural partitioning, information hiding, functional independence, cohesion, coupling, design heuristics;

- Design methods: data design, architectural design, transform mapping, design optimization, human computer interface design, procedural design and tools; Design documentation.
- Software testing: Testing objectives, Testing principles, Testability, test case designing, white box testing; Basis path testing: Condition testing, data flow testing, loop testing; Black box testing: graph based testing methods, equivalence partitioning, Boundary value analysis, comparison testing; Testing documentation and help facilities; Software testing strategy: unit testing, integration testing, validation testing, system testing.
- Software project management techniques: project metrics, software measurement and metrics, software quality metrics;
- Software project planning: objectives of planning, resources, project estimation and estimation models, project decomposition techniques, make-buy decisions; automated estimation tools.
- Risk management: software risks, risk identification, risk projection, risk mitigation, monitoring and management; Project Scheduling: people and effort relationships, defining tasks, defining task network, scheduling techniques; Software teams and intra-team relationships; role of project manager.
- Software Quality Assurance: Concept of quality, quality control vs. quality assurance, cost of quality, factors that affect quality, quantitative view of quality, quality metrics, defect removal efficiency SQA activities, ISO standards and CMM practices, SEI levels, Software reviews, Formal approaches to SQA, Statistical Quality Assurance. Software reliability, reliability metrics, reliability models, meeting reliability requirements.
- Effective metrics for software process: Measurement principles, attributes of software metrics, metrics for analysis model, metrics for design model, metrics for source code, metrics for maintenance.
- Software reuse: difficulties in reuse, hardware reuse vs. software reuse, reusable artifacts, domain engineering approach, analysis design and construction of reusable components, classification and retrieval of components, economic impact of reuse and reuse metrics.
- Computer aided software engineering: CASE tools and application.

(f) Assessment method

- Continuous assessment through practical exercises and Coursework, together with two scheduled tests (40%)
- Final exam at the end of the Semester and accounts for 60% of the final grade.

(g) Reading lists

- (i) J.F. Peters, W.Pedrycz, Software engineering: An Engineering approach, John Wiley, 2000.
- (ii) R.S. Pressman, Software engineering: A Practitioners Approach, 5th Edition, McGraw Hill, 2005.
- (iii) Sommerville, Software engineering, 8th Edition, Addison Wesley, 2008.
- (iv) D. Ghezzi, M. Jazayeri, D. Mandrioli, Fundamentals of software Engineering, Prentice Hall of India, 2004.

#### 5.4.6 CSC 2114 Artificial Intelligence

(a) 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.

(b) Aims and Objectives

- 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.

(c) Learning outcomes

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

- 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.

(d) Intellectual, Practical and Transferable skills

This course describes and discusses several algorithms and techniques within the fields of artificial intelligence and machine learning. These are algorithms and techniques that have practical applicability in computer science fields. Theoretical and practical understanding of these areas equips the student with the insights and tools required for solving complex and difficult problems, and for implementing them in software.

(e) Teaching and Learning Pattern

Teaching and learning is implemented through lecture, lab and tutorial sessions. Students are expected to make presentations of their assignments for discussion in class.

(f) 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.
- (g) Assessment method  
Assessment will be by continuous assessment through practical exercises and Coursework (40%) and final exam (60%)
- (h) Reading lists
- (i) R. Duda, P.Hart, Pattern Classification and Scene Analysis, Wiley, 1973.
  - (ii) E.A. Feigenbaum, J.Feldman; Computers and Thoughts, AAAI Press, 1995.
  - (iii) N.J. Jilsson, Problem Solving Methods in Artificial Intelligence, McGraw-Hill, 1971.
  - (iv) J.Lloyd, Foundation of Logic Programming, Springer-Verlag, 1993.

## 5.5 Semester 4

### 5.5.1 CSC 2200: Operating Systems

- (a) Description  
Operating Systems course introduces students to software that controls hardware and makes the hardware usable. Its interaction with other computer devices and how it controls other computer processes is explored.
- (b) Aims  
The aims of the course are:
- To provide students with a detailed understanding of how operating systems work.

- To provide students with skills to write basic programs to utilize underlying operating system infrastructures.
- (c) Learning outcomes  
The dominant categories of operating systems are Windows and UNIX (Linux, Mac OS, Solaris, etc). Students of operating systems are expected to have a proper understanding of the differences between these two. Students should be able to understand different design principles for operating systems and various software tools that make operating systems usable.
- (d) Teaching and Learning Pattern  
The teaching pattern is by lectures, lab sessions and group projects.
- (e) Indicative Content
- Operating Systems Structures
  - Processes and threads
  - Thread creation, manipulation and synchronization
  - Deadlock
  - Implementing Synchronization operations
  - CPU scheduling
  - Memory management
  - File systems and file system implementation
  - Monitors
  - Segments
  - Disk Scheduling
  - Networking
  - UDP and TCP
- (f) Assessment method  
The assessment will constitute Practical assignments on at least 5 chapters of the course and written course work (40%) and written Exam (60%)
- (g) Reading list
- (i) Operating Systems: Internals and Design Principles 5th Ed by William Stallings, Prentice Hall, 2005.

### 5.5.2 BSE 2203 Computer Networks & Data Communication

(a) Description

This course examines principles, design, implementation, and performance of computer networks and data communication. The aim is to sharpen student's understanding and skills in computer networking and data communication. Subjects for discussion include: Internet protocols and routing, local area networks, wide area networking, wireless communications and networking, performance analysis, congestion control, TCP, network address translation, multimedia over IP, switching and routing, mobile IP, peer-to-peer networking, network security, and other current research topics and technologies.

(b) Aims

The aims of the course are

- To provide a solid basis on the theoretical and practical understand of data communication networks
- To introduce students to standards and guidelines in computer and data communication networks
- To impart knowledge and skill relevant for the design, implementation and maintenance of modern computer communication networks
- To introduce students to emerging technologies in data communication

(c) Learning outcomes

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

- Describe theoretical concepts of computer communication systems, including standards, protocols, and infrastructure devices e.g. routers and switches.
- Design, install and manage a simple Local Area Network or A campus Wide Area Network
- Describe emerging communication technologies particularly wireless and fiber optic technologies
- Deploy sound network security practices and tools
- Describe emerging research directions in computer communication networks

- To write a few network scripts
- (d) Teaching and Learning Pattern  
The course will be delivered inform of lectures, tutorials, lab experimentation, and group assignments.
- (e) Indicative Content
- Network services and applications: DNS, HTTP, SMTP, peer-to-peer systems
  - Network transport architectures, TCP, UDP, TCP congestion control
  - Routing and forwarding, intra-domain, inter-domain routing algorithms and Mobile IP
  - Link layers and local area networks, Ethernet, WiFi, and mobility
  - Multimedia communications and quality of service
  - Network measurement, inference, and management
  - Network security (ACL, IPSec, etc)
  - Network programming
  - Network experimentation and performance analysis
  - Protocol verification
- (f) Assessment method  
Assessment will be in terms of tests and Assignment (40%) and final examination (60%)
- (g) Reading lists
- (i) James F. Kurose and Keith W. Ross. Computer Networking - A Top Down Approach Featuring the Internet, 3rd edition, Addison-Wesley, 2004, ISBN 0-321-22735-2.
  - (ii) L. Peterson and B. Davie, Computer Networks: A Systems Approach. Morgan Kaufmann Publishers, 1999.

### 5.5.3 CSC 2209: Systems Programming

- (a) 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 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  
Assessment will be in form of tests and practical assignment (40%) and final written examination (60%)
- (f) Reading List
- (i) Advanced programming in the Unix Environment, by W. Richard Stevens, Addison-Wesley 2008

#### **5.5.4 CSC 2210 Automata, Complexity and Computability**

- (a) Description  
The course introduces students to the concept of automata and complexity. It sets a background for more advanced studies like compiler construction and principles of programming languages.
- (b) Aims The aims of the course are:

- To introduce students to the concepts of complexity, automata and computability
  - To prepare students for advanced studies in compiler construction and principle of programming languages
- (c) Teaching and learning pattern Teaching will be in form of class lectures and tutorials
- (d) Indicative content
- Finite state machines and regular languages;
  - Deterministic and non deterministic machines;
  - Equivalence and minimization;
  - Regular expressions and regular grammars;
  - Kleene's theorem.
  - Push-down automata and context free grammars;
  - Normal forms of grammars; Top-down and bottom-up parsing.
  - Turing machines and computability;
  - Church's thesis;
  - NP-Computable problems.
- (e) Assessment method  
Assessment will be in terms of Assignments and tests (40%) and final written examination (60%)
- (f) Reading list
- John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman: Introduction to Automata Theory, Languages, and Computation; Addison Wesley, 2000.
  - Daniel I. A. Cohen Introduction to Computer Theory; John Wiley and Sons, Inc, 1996.

### 5.5.5 CSC 2212 Individual Project II

- (a) Description  
The course is to strengthen the students' hand on skills on integration of the different skills into a programming project. Students are expected to develop a running computer application bigger in size and

more polished than in CSC 1208 Individual Project I. More emphasis will be put on creativity, robustness, data validation, security and completeness.

(b) Aims

The aims of the course are:

- To provide an avenue for students to integrate different subject areas into a single application
- To sharpen the students problem solving skills
- To improve student's ability to read on his/her own as an avenue for solving a certain real life problem

(c) Teaching and Learning patterns

A list of problems to be addressed will be given to students at the beginning of the semester. The expected characteristics of the developed system will be indicated. Each student will pick a problem of his/her own choice and he/she will be free to consult any member of staff.

(d) Indicative Content

The content is to be determined by the problem the student intends to address

(e) Assessment Methods

Assessment will be by demonstrations of students to lecturers as the lecturers award marks. The demonstrations will be in phases as the project progresses.

(f) Reading lists

Students will be required to read any literature related to the project at hand

### 5.5.6 BSE 2201 Network Applications Development

(a) Description

This course gives students theoretical principles and hands on experience of developing network applications. It caters for functional and non functional issues peculiar to network applications. These include security, robustness and performance.

(b) Aims

The aim of the course is to:

- Give students theoretical and practical skills of developing network applications
  - Equip students with security and performance aspects of network based applications
  - Expand the applications of students programming skills
- (c) Teaching and Learning pattern  
Teaching will be in terms of lectures and laboratory programming practical sessions
- (d) Indicative content
- Design principles for network-based applications;
  - Design and development of Java Servlets, JSP, Web services and .NET;
  - Principles of information security in network-based applications;
  - http and https protocols.
- (e) Assessment Method  
Assessment will be in terms of coursework and tests (40%) and final examination (60%)
- (f) Reading List
- (i) Core Servlets and Java Server Pages, Volume 1: Core Technologies by Marty Hall (2nd Edition), Prentice Hall , 2000.

### 5.5.7 CSC 2214 Cryptology and Coding Theory

- (a) Description  
This course provides a foundation for further studies in information security. The course introduces students to the exciting fields of cryptology and coding theory. Fundamentally, it deals with the mathematics that underlies modern cryptology. Cryptology combines the studies of cryptography, the creating of masked messages, and cryptanalysis, the unraveling of masked messages. Coding theory is the study of coding schemes used to detect and correct errors that occur during the data transmission.
- (b) Aims  
The aims of the course are

- To understand the building blocks of crypto systems and error correction
- To gain historical understanding of the evolution of crypto systems.
- To develop tools necessary to crypto analyze crypto systems
- To gain insights in the practical application of cryptology and error correction in the modern information age.
- To understand the goals and trade-offs associated with encryption and error-control coding systems.

(c) Indicative Content

- History of cryptology and coding theory
- Shift registers
- Classical crypto-systems
- Stream ciphers
- Block ciphers
- Information theory
- Crypto analysis techniques
- Introduction to Elliptic curve cryptography
- Basic Algebra
- Coding theory fundamentals
- Linear codes
- Hamming codes
- Secret sharing schemes
- Introduction to Complexity
- Hash functions
- PGP & PKI Diffie-hellman key exchange protocol

(d) Learning outcomes

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

- Deploy sound cryptographic practices and tools
- Discuss the goals and trade-offs associated with encryption and error-control coding systems.

- (d) Teaching and Learning Pattern  
The course will be delivered in form of lectures, tutorials, and group assignments.
- (f) Assessment method  
At least 2 tests and 1 assignment (40%) One 3 hour examination (60%)
- (g) Reading lists
  - (i) Handbook of Applied Cryptography, by A. Menezes, P. van Oorschot, and S. Vanstone, CRC Press, 1996 Trappe & Washington,
  - (ii) Introduction to Cryptography with Coding Theory, Prentice-Hall, 2001 ISBN 0130618144
  - (iii) Introduction to Cryptography with Coding Theory by Trappe and Washington, 2nd edition, Prentice Hall, 2006.
  - (iv) The Code Book by Simon Singh, Doubleday, 1999
  - (v) Introduction to Algebraic Coding Theory with Gap, Sarah Spence Adams, 2005.

## **5.6 Year 2 Recess Term**

### **5.6.1 CSC 2301 Industrial Training**

During Industrial training, students are to go and work in an organization with an IT department. The student is to be under the supervision of one of the workers in the organization. The student is assigned duties in line with the operations of the organization. Staff from Makerere University will make visits to get the students' view of the organization as well as the organization's view about the student. The supervisor will be given a form to evaluate the students and the student will make a report about his experience. The two reports will be used to evaluate the student.

## **5.7 Semester 5**

### **5.7.1 CSC 3103: User Interface Design**

- (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 a 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.

(c) Teaching and Learning Pattern

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

(d) 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

(e) Assessment method

Assessment will be in form of a assignments and tests (40%), practical Exam (30%) and final written exam (30%)

(f) Reading List

- (i) Norman, D. A. The Design of Everyday Things. New York, NY: Doubleday, 1990. ISBN: 0385267746.
- (ii) Nielsen, J. Usability Engineering. Burlington, MA: Academic Press, 1994. ISBN: 0125184069.
- (iii) Mullet, K., and D. Sano. Designing Visual Interfaces: Communication oriented techniques. Prentice Hall, 1994. ISBN: 0133033899.

### 5.7.2 CSC 3110 Database Management Systems II

(a) Description

The course gives students advanced knowledge in the operation of database management systems. Operational aspects like speed and security are to be extensively addressed. Internal operations of database management systems like indexing, query processing and transactions are to be covered. These are to create a strong basis for future advanced studies/research.

(b) Aims

The aim of the course is to equip students with fundamental knowledge of the operation of database systems and how they interact with the operating system.

(c) Teaching and Learning pattern

Teaching and learning will be in form of lectures and class discussions/presentations.

(d) Indicative Content

- Enhanced ER modeling
- Tuning (operational) systems to improve performance
- Advanced scripting
- Data validation
- Relational algebra and Relational calculus
- Low level output representation using relational calculus
- Query processing
- Transactions management
- Indexing and hashing

- (e) Assessment Method  
Assessment will be in terms of tests and assignments (40%) and final examination (60%)
- (f) Reading List
- (i) R. Elmasri, S. B. Navathe: Fundamentals of Database Systems, 4th Edition. Benjamin/Cummings, 2003, ISBN 0-8053-1748-1

### 5.7.3 CSC 3111 Operations Research

- (a) Description  
The course is to introduce students to the broad concepts of operations research. students will learn how to interpret and analyze OR problems, formulate them as problems and use existing techniques to solve them.
- (b) Aims  
The aim of the course is to improve students' problem solving skills by subjecting them to real life problems and guide them through formulation of their solutions. The choice of the cases chosen depends on their applicability in real life computing environment.
- (c) Learning outcomes  
By the end of the course, the student should be able to
- Correctly formalize real life problems into OR problems
  - Adequately solve typical OR problems
  - Make post optimality analysis on OR solutions
- (d) Teaching and Learning Pattern Teaching will be in form of lectures and Tutorials
- (e) Indicative Content
- Linear programming
  - Network Analysis
  - Decision trees
  - Markov processes
- (g) Assessment method Assessment will be by assignments and tests (40%) and final written exam (60%)

(h) Reading lists

- (i) Linear Programming and Network Flows by Mokhtar S. Bazaraa, John J Jarvis and Hanif D Sherali, John Wiley, 2005
- (ii) Operations Research: Applications and Algorithms by Wayne L. Winston. Wadsworth Publishing Company, 1997

#### 5.7.4 CSC 3112: Principles of Programming Languages

(a) Description

The course introduces students to the low level organization and operation of programming languages. It covers semantic and syntactic as well as operational issues in programming languages. The building blocks of programming languages are explored.

(b) Aims

The aims of the course are

- To give students fundamental knowledge in the organization and operation of programming languages
- To make students appreciate the possible future evolutions of programming languages
- To expose students to causes of operational (like performance, security, etc) characteristics of programming languages

(c) Learning outcomes

By the end of the course, students will be able to:

- Understand common language paradigms
- Know the different building blocks of a programming language
- Know how the different blocks of a programming language interact

(d) Teaching and Learning Pattern

Teaching will be largely by lectures, tutorials and class assignments

(e) Indicative Content

- Overview over programming language paradigms
- Common principles: syntax, syntax trees, formal semantics (denotational and operational), variables and binding

- Types: role of types in programming and programming languages, types and their operations: products, sums, functions, recursive types, reference and array types
- Primarily imperative issues: control flow, arrays, pointers and references, parameter-passing mechanisms, scoping
- Type systems: strongly typed languages type checking (static vs. dynamic), type equivalence (by name vs. structural), overloading, coercion, polymorphism, type inference
- Binding: declarations and environments. Block structure: scope and visibility, stack discipline. Bound occurrences: static vs. dynamic binding.
- Encapsulation: information hiding, modules, abstract data types, classes
- Language implementation: parsing, code generation, garbage collection.

(f) Assessment method

Assessment will be by Tests and Assignments (40%) and final written examination (60%)

(g) Reading lists

- (i) Friedman, Wand, and Haynes, Essentials of Programming Languages, MIT Press, 2001

### 5.7.5 CSC 3113 Emerging Trends in Computer Science

(a) Description

The course is to expose provide students with an opportunity to search for knowledge in an area of interest. It is to allow a student do lightweight research and explore the current trends in a certain computer science area.

(b) Aims

The aims of the course are:

- To aid a student get an in depth understanding of the developments in one area of computer science
- To improve the student's research skills
- To develop confidence in the students on the ability to search for knowledge with little guidance.

- (c) Teaching and Learning pattern  
Students will be grouped in theme areas which will be covered by a set of staff. Staff will guide students on the specific themes/topics in the area as well as where to search for information. Staff will also address experiences and hardships found.
- (d) Indicative content  
The content is not specific but will be dependent on the area the student chooses to pursue.
- (e) Assessment Method  
Students will present their findings in a report. The study report will be an outline and explanation of what the student has found out in the state of practice in the area of choice. The write up will be evaluated and the final mark awarded

#### **5.7.6 BIS 3100: Modeling and Simulation**

- (a) Description  
The course gives students theoretical and practical skills in modeling and simulation of dynamic systems with a view of learning their behavior and the sensitivity of that behavior to certain parameters.
- (b) Aims  
The aims of the course are:
- Familiarize students with modeling and simulation techniques that are applicable under varying circumstances
  - Equip students with practical experiences of composing models and running simulations under varying circumstances
  - Equip students with skills of correctly representing simulation results
- (c) Teaching and Learning Method  
Teaching and Learning will be in form of Lectures and laboratory demonstrations
- (d) Indicative Content
- Simulation of operational systems
  - Simulation as a decision making methodology
  - Model development and validation,

- Design of simulation experiments,
  - Generation of appropriate values of random variables,
  - Interactive procedures and interpretation of results.
- (e) Assessment Method Assessment will be in form of tests and assignments (40%) and final examination (60%)
- (f) Reading List
- (i) Business Modeling and Simulation by Les Oakshott, 1997, Trans-Atlantic Publications
  - (ii) System dynamics modeling a practical approach by R.G. Coyle, Chapman & Hall/CRC, 1996.

### 5.7.7 CSC 3105 Computer Graphics

- (a) Description  
The course covers general purpose graphics systems and their use. It gives an in depth knowledge of computer graphics and graphical user interfaces.
- (b) Aims  
The aims of the course are:
- Introduce students to the concepts of graphical representation on computers
  - Teach students the design of good graphical user interfaces
- (c) Teaching and Learning Pattern  
Teaching will be in terms of class lectures and tutorials
- (d) Indicative Content
- Graphics hardware,
  - Geometrical transformations,
  - Surface and volume visualization,
  - Design and implementation of graphical user interfaces.
  - Two dimensional imaging processes.
  - Computer graphics applications.
  - Display system organization;

- Display devices and modes;
  - Display file construction and its structure;
  - Graphic primitive - device initialization,
  - view porting and windowing;
  - Line drawing,
  - simple and symmetrical Digital Differential Analysis (DDA);
  - Arch and circle generating DDA Line; and polygon clipping algorithms;
  - Curve plotting;
  - Transformations- projections and perspective views;
  - Picture segmentation: Graphics standards - PHIGS and GKS.
- (e) Assessment method Assessment will be in terms of assignments and tests (40%) and final exam (60%)
- (h) Reading lists
- (i) Introduction to Computer Graphics by James D. Foley, Andries van Dam, Steven K. Feiner, John F. Hughes and Richard L. Phillips, Addison Wesley, 2003.
  - (ii) Fundamentals of Computer Graphics by Peter Shirley. AK Peters, 2002.

### 5.7.8 CSC 3114 Selected Topics in Computer Science

- (a) Description  
The course is to give an avenue for visiting or local staff to design content for an area that they are researching in but not covered in the curriculum. This could be an upcoming field yet to be incorporated in the curriculum or some outstanding state of practice in a certain computer science area.
- (b) Aims  
The aim of the course is to provide an avenue for exposing students to new interesting areas of computer science which are not (yet) incorporated into the curriculum.
- (c) Teaching and learning pattern  
The teaching and learning pattern is to be designed by the lecturer with approval of the head of department (or another senior staff appointed by him/her)

- (d) Indicative content  
The content is to be determined by the lecturer. The head of department (or senior staff appointed by him/her) may have to be convinced it is sufficient in depth and breadth.
- (e) Assessment method  
The lecturer concerned has to propose the assessment method but has to be comprehensive enough, in the view of the Head of Department, to examine the content.
- (f) Reading List  
The textbooks are to be determined by the lecturer. These depend on the content he/she is going to deliver.

### 5.7.9 CSC 3115 Advanced Programming

- (a) Description  
This course highlights programming practices that are vital in the day today work of a programming professional. While many systems are described by functionalities, some important aspects like security, robustness, and maintainability are ignored. Students are to get an in depth understanding of these concepts as well as exploring the current trends in the programming environment.
- (b) Aims  
The aim of the course is to concretize the student's past programming experience as well as highlighting critical practices in programming that are necessary for a professional programmer
- (c) Learning outcomes  
By the end of the course, the student should be:
- Able to implement non functional but critical aspects of programming like robustness and security
  - Able to develop well documented and well structured software that can easily be maintained
  - Knowledgeable in other programming practices like mobile programming
  - Aware of newer programming paradigms like service oriented and cloud computing

- (e) Teaching and Learning Pattern  
Teaching will be by lectures and lab demonstrations.
- (f) Indicative Content
  - Programming for Security:
  - Programming for Robustness
  - Programming for Maintainability
  - Trends in Programming Paradigms
- (g) Assessment method  
Assessment will be by tests and practical assignments (40%) and final written examination (60%)
- (h) Reading lists
  - (i) Advanced Programming in the UNIX Environment by W. Richard Stevens and Stephen A. Rago Addison Wesley 1992

#### **5.7.10 BIT 3102 Entrepreneurship and Business**

- (a) Course Description:  
This course offers an overview of the entrepreneurial process. Students will develop a business plan using their own business idea. They will learn skills and characteristics of successful entrepreneurs, techniques for evaluating business opportunities, planning tools, selling and marketing basics, financial analysis, record keeping, laws and regulations of Uganda, and step by step procedures for starting a small business.
- (b) Aims:  
A student that undertakes this course should be able to:
  - Identify the talents and skills required to become a successful entrepreneur.
  - Describe the various types of business opportunities and successful entry strategies.
  - Specify the various steps employed in starting a business in Uganda.
  - Identify sources of information and assistance available to those who desire to start their own business.
  - Analyze and research how a selected business markets and sells its products. Be able to prepare a report summarizing the findings of this analysis and research.

- Provide a general company description for the student's start-up business.
- Identify products and services which will be offered in the start-up business.
- Outline the components of a business plan including marketing, operations, finances, and management and organization
- Be able to use the business plan as an integral tool in small business management.

(c) Learning Outcomes: On completion of this course unit, the students will be able to:

- Describe the Ugandan business environment with respect to owning and operating a small business;
- Compare advantages and disadvantages of different types of business start-up and purchase opportunities;
- Conduct research and field work to determine viability of a business idea;
- Perform a competitive analysis within a selected industry;
- Develop and present a detailed business plan including market and legal analysis financial requirements, facilities and management plans and promotional strategies and;
- Determine social and ethical responsibility.

(d) 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.

(e) Indicative content:

- Identification of talents and skills required to be a successful entrepreneur: Personal talents, Technical skills, Educational background
- Overview of business opportunities: Trade association reports, Government reports, Demographic information, Surveys

- Business entry strategies: Steps in starting a business, Purpose of business, Description of business, Starting a business versus buying a business
  - Planning: Setting goals, Operational approach, Laws and regulations affecting a small business in Uganda, CAP and Local bye laws.
  - Sources of information and assistance: Trade association and other non-governmental sources, Local, state, and federal governmental sources, Small business analysis
  - Description of a comprehensive business plan: Reason for preparing a business plan, Types of business plans, Form of business plan, Contents of a successful business plan
  - Outline of a business plan: Preparation of an executive summary, Establishment of company strategy, Development of a marketing plan, Sales strategy, Financial plan
  - Completion of the business plan: Writing the plan, Outside review of the plan, Social and Ethical Responsibility
- (f) Assessment method:  
Assessment will be in terms of tests and practical exercises (40 %) and a final examination (60%)
- (g) Reference Books:
- 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
  - Thoma W. Zimmerer and Norman M. Scarborough (2005) Essentials of Entrepreneurship and Small Business Management. 4th Ed. ISBN 0-13-191856-7

## 5.8 Semester 6

### 5.8.1 BIT 3204 Enterprise Network Management

- (a) Description  
The course is to equip students with knowledge of managing enterprise computer/data networks. This caters for enterprise critical aspects like reliability, security and user management.

- (b) Aims
 

The aims of the course are to equip students with skills to plan, manage and monitor large computer networks.
- (c) Teaching and Learning Pattern
 

Teaching will be in terms of lectures and Tutorials
- (e) Indicative Content
  - The infrastructure for network management.
  - The Internet and ISO models. SNMP, ASN.1.
  - Structure of management information and MIB.
  - Events and managed objects. EMS vs. NMS . NOC,
  - Remote and Web-based management tool.
  - OAM&P Focus:
    - Operation, maintenance, performance monitoring.
    - Introduction of traffic engineering and load balancing.
- (f) Assessment method Assessment will be in form of coursework and tests (40%) and final examination (60%)
- (g) Reading lists
  - (i) Effective management of local area networks by Kornel Terplan, McGraw-Hill, 1992

### 5.8.2 CSC 3205 Compiler Design

- (a) Description
 

In this course unit, students shall understand the complete process of translating a program in a high-level language to machine language. The course gives an introduction to the design and implementation of a compiler with emphasis on principles and techniques for program analysis and translation. It also gives an overview of the tools for compiler construction. Lexical analysis, token selection, transition diagrams, and finite automata. The use of context-free grammars to describe syntax, derivations of parse trees, and construction of parsers. Syntax-directed translation schemes; Intermediate code; Symbol table; Code generation; Detection, reporting, recovery and correction of errors.

(b) Aims

The aim of the course is to allow students to examine how a high-level language program is accepted as input and translated into assembly language or machine language so that the central processing unit receives instructions which it understands and can execute.

(c) Teaching and Learning pattern

The course consists of a traditional theoretical component and a project component. The lecture component introduces the basic concepts of compiler writing. The project component will involve students in writing a compiler for a specified programming language.

(d) Indicative content

- Language translators: Introduction to compilers and interpreters.
- The structure of a compiler: lexical analysis, parsing, semantic analysis.
- Intermediate code generation, register allocation, global optimization.
- Lexical scanning
- Parsing
- Automatic parser construction. FIRST and FOLLOW functions. LL(1) parsers. LR parsers. Conflicts in LR grammars and how to resolve them
- Semantic analysis: Attributes and their computation, tree-traversals, visibility and name resolution. Inherited attributes and symbol tables. Name resolution in block-structured languages
- Type checking: Type systems, varieties of strong typing, overload resolution, polymorphism and dynamic dispatching. Type-checking and type inference, unification
- Run-time or Run-time organization: storage allocation, non-local references, parameter passing, dynamic storage allocation. Exception handling, debugging information
- Intermediate code generation: control structures, expressions, simple register allocation. Aggregates and other high-level constructs
- Global optimization

(e) Assessment method

Assessment will be by assignments and/or tests (40%) and written examination (60%)

(f) Reading list

- (i) Compiler construction by William McCastline Waite, Gerhard Goos Springer Verlag 1994

### 5.8.3 CSC 3206 Group Project

(a) Description

The course is to allow students, in groups, to integrate the knowledge acquired over the previous five semesters into solving a non trivial problem through a computer application. Emphasis will be put on the systematic development methodology, the documentation of the development process, and how well the developed system address the problem to be solved. Non functional attributes like robustness, usability, security and reliability will also be tested.

(b) Aims

The aim of the course is to give students experience in

- Group and collaborative work
- Proper procedures in development of computer systems
- Proper documentation of the software development process
- Development of big not trivial computer projects.

(c) Teaching and Learning pattern

Students will be under the supervision of a member of staff (at least at a rank of Assistant lecturer). The supervisor will guide them in the day today progress of the project. When the supervisor feels the students have addressed the problem at hand, (s)he will sign off their report and recommend them for examination.

(d) Indicative content

The content is to be determined by the students under the guidance of the supervisor.

(e) Assessment Method

Students will submit the signed report for examination to the department. The department will appoint an examination panel of at least 5 people who will evaluate the report as well as testing the system. Students will be required to present their work and answer any questions from the panel. Each member of the panel will award a mark depending on his/her view on the worthiness of the developed application.

#### 5.8.4 CSC 3207 Computer Security

(a) Description

Computer security is a branch of technology concerned with digital security or information security applied to computers. Since the largest part of the computer that users interact with is software, computer security pays big attention to development of secure software.

(b) Aims

The aims of the course are:

- To introduce students to threats faced by computers in the connected digital world.
- To introduce students to techniques that are used to protect computers against various threats.

(c) Teaching and Learning Patterns

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

(d) Indicative content

- Digital security principles
- Hardware based security mechanisms
- Secure operating systems
- Security architecture
- Security by design
- Secure coding (Software Security)
- Access Control Lists
- Security Applications

(e) Assessment method

Assessment will constitute Practical assignments on at least 5 chapters of the course and written course work (20%) and written Exam (60%).

(f) Reading list

- (a) Ross J. Anderson: Security Engineering: A Guide to Building Dependable Distributed Systems, Willey 2001.
- (b) Robert C. Seacord: Secure Coding in C and C++. Addison Wesley, 2005.

### 5.8.5 BIT 3200 Business Intelligence and Data Warehousing

(a) Description

This course covers techniques and software tools that can assist management that deals with large amounts of data in management and business decision making. The course covers the fundamental differences between databases and data warehouses, the techniques of developing data warehouses as well as manipulating them to generate business strategic decisions.

(b) Aims

The aims of the course are

- To give students the understanding on the role and operation of data warehouses
- To equip students with skills of developing data warehouses
- To equip students with skills of maintaining existing data warehouses
- To equip students with skills of manipulating data warehouses to generate information for business decision making

(c) Learning outcomes

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

- Distinguish the roles of a database from that of a data warehouse
- Develop a data warehouse
- Populate and manipulate a data warehouse

(d) Teaching and Learning Pattern

Teaching will be in form of class lectures, tutorials, lab demonstrations as well as class presentations.

(e) Indicative Content

- Data warehouse concepts: partitioning, granularity, record of source, and meta data
- Building viable decision support environments.
- Architect development,
- Data migration and integration,
- Use of operational data stores, and transactional systems.

- (f) Assessment method Assessment will be in form of tests and practical assignments (40%) and final written examination (60%)
- (g) Reading lists
  - (i) Data warehousing fundamentals by Paulraj Ponniah Wiley 2001.

#### **5.8.6 BSE 3202: Distributed Systems Development**

##### 1. Description

This course gives students theoretical and practical skills on development of distributed systems and applications. This include distributed-system-specific challenges like reliability and robustness.

##### 2. Aims

The aim of the course are to equip students with skills of developing distributed systems

##### 3. Teaching and Learning patterns

Teaching will be by class lectures and laboratory practicals/demonstrations

##### 4. Indicative content

- Event-driven software architectures,
- Distributed object computing,
- Development, documentation and testing of distributed applications
- Techniques for reusable, extensible and efficient software systems
- Maintainability and concurrence in distributed systems
- Abstraction based on patterns and object-oriented techniques

##### 5. Assessment Method

Assessment will be in form of tests and (practical) assignments (40%) and final examination (60%)

##### 6. Reading list

- (i) S. Tanenbaum and M. V. Steen, Distributed Systems: Principles and Paradigms, Second Edition, Prentice Hall, 2006.
- (ii) R. Anderson, Security Engineering: A Guide to Building Dependable Distributed Systems, John Wiley & Sons, 2001.

## **6 Resources and Infrastructure**

The Department of Computer Science and the Faculty of Computing and Information Technology have the enough resources and infrastructure to sufficiently run the programme.

### **6.1 Funds**

Fees payable by the students will enable the University to sustain the programme. The fees and the number of students to be admitted are the same as those on the old curriculum.

### **6.2 Staff**

The Faculty of Computing and Information Technology has a big pool of staff who can competently teach the courses. The list of staff members in the Department of Computer Science is in Appendix B. The department also relies on staff from other departments like Information Technology, Information Systems and Networks together with visiting staff from other universities.

### **6.3 Lecture Space**

Initially, the Faculty of Computing and Information Technology housed in a 2,500 square meter building (Block A). In January 2009, a new 12,000 square meter building (Block B) was officially opened. Block B has Six Lecture Theaters each of which can accomodate up to 500 students.

### **6.4 Computer Laboratories**

The old and new faculty buildings have general laboratories (strictly for students practice), teaching laboratories and specialized laboratories. These laboratories are shared among the departments of the faculty and are scheduled by the ICT services unit. In all, the faculty has 6 general laboratories and 4 specialised laboratories. Currently, the faculty has approximately 2000 computers and 5000 students. This leads to a student to computer ration of 1: 2.5 which is adequate for the practical components of the curriculum. More

## **6.5 Software**

On top of the physical computers, students need software for the different practical sessions. Different computers are installed with different software 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.

## **6.6 Library Services**

Makerere University Library supports a book bank system which is operated at the Faculty level (Block B Building). The book bank is stocked with up to date literature. The books in the book bank have been acquired through supplies from the Makerere University Library and purchases by the Faculty for books that are difficult to purchase by the main library. In addition to this facility the Makerere University main library provides access to books, print journals, e-journals, a well stocked reference section and connections to many remote databases. The Online Journals which are also accessed through the main University library provide a range of products from abstracts to full text papers. The University Library has also acquired a wide range of online books which are to access and this is in line with promoting e-learning.

## **7 Quality Assurance**

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 feedback benchmark for possible future revisions in the curriculum

The following activities will be carried out in the process of monitoring and assuring quality in the proposed program.

### **7.1 Feedback from students enrolled**

In the current set up, each class has 4 student representatives (2 day, 2 evening). 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.

At the end of the semester, samples of students 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. The Faculty of Computing and Information Technology is the process of creating a computerized system that will capture and analyze the data. With the computerized system:

- (i) Every student will be required to assess every lecturer teaching him/her, the sample space will therefore be increased
- (ii) No time will be required in the analysis of the results. Staff and faculty management will be able to get the feedback instantly
- (iii) Data will be easily archived and therefore the trend of staff performance in specific areas will be easy to visualize

## **7.2 Class meetings**

The departmental management makes at least 1 meeting 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

## **7.3 Use of ICT in availing lecture materials**

Currently, Makerere University has the blackboard e-learning tool on its Intranet. Students in the Department of Computer Science have adequate access to computers. This creates a good environment 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 blackboard. 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 in turn will, in turn, increase the material covered and taken in by the students.

## **7.4 Peer review**

All members of 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. Staff will be free to advise fellow staff on the content, depth

and presentation of materials. Consequently, for every course, students will access the best possible material in the view of all staff in the department not the course instructor

### **7.5 External examiners' reports**

Like it is everywhere in Makerere University, student results are reviewed every semester by a senior external academician. This is to bring a 'foreign view' of the quality of the program. External examiners write reports on their view of the curriculum/examinations. Some recommendations can be implemented immediately while others have to be implemented 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.

### **7.6 Industrial Training placement reports**

Students will get Industrial placements in the programme. This will help expose them to the world of the industry. Reports from the placements will be used to gauge the effectiveness of the curriculum. They are also to be used to identify and address the deficiencies in the curriculum.

### **7.7 Tracer studies**

The Faculty of Computing and Information Technology 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 Computer Science will use outputs of the tracer studies to gauge the quality of the program and whenever necessary, improve it.

Staff

No	Name (Highest Qualification)	Rank	Comment
1	Venanius Baryamureeba (Ph. D)	Professor	Dean
2	Jose G. Quenum (Ph. D)	Senior Lecturer	
3	John Quinn (Ph. D)	Lecturer	
4	John Ngubiri (Ph. D)	Lecturer	Head of Department
5	Narcis T Rwangoga (M. Sc)	Ass. Lecturer	Ass. Head of Dept
6	Florence Tushabe (M.Sc)	Ass. Lecturer	Ph. D. Student
7	Richard Ssekibuule (M. Sc)	Ass. Lecturer	Ph. D. Student
8	Jonathan Kizito (M. Sc)	Ass. Lecturer	Ph. D. Student
9	Engineer Beinomugisha (M.Sc)	Ass. Lecturer	Ph. D. Student
10	Ronald Azairwe (M.Sc)	Ass. Lecturer	Ph. D. Student
11	Rose Nakibuule (M.Sc)	Ass. Lecturer	Ph. D. Student
12	Doreen Tuheirwe (B.Sc)	Teaching Assistant	M. Sc. Student
13	Joab Ezra Agaba (B.Sc)	Teaching Assistant	M. Sc. Student

Staff and Load Distribution

Staff	Specialisation	Old	Ld	New	Ld	Old	Ld	New	Ld
V. Baryamureeba (PhD, Professor)	Programing, Theoretical computing	MCS 9100	3	MCS 9100	3	MCS 9200	3	MCS 9200	3
J.G.Quenum (PhD, S. Lecturer)	Theoretical computing, Compiler Theory, Dis- tributed Systems	MCS 7117 MCS 8102 MCS 8101	9	MCS 9102 MCN 7105 MCN 7118	9	MCS 7215 MCS 7218 MCS 9202	9	MCN 7207 MCS 9202	6
J. Quinn (PhD, Lecturer)	Image pro- cessing, Mathematical Comput- ing, Image processing, Pattern Recognition	MCS 7116 MCS 8104 MCS 7101	9	MCS CSC 2109 MCS 9101	9	MCS 7216 MCS 7217 MCS 9201	9	MCS 7217 MCS 7224 MCS 9201	9
J. Ngubiri (PhD, Lecturer)	Programing, Disributed Systems, Mathematical Computing, Database Systems, Algorithms	MCS 7115 CSC 3100	9	MCS CSC 2111 MCS 7109	9	MCS 7202 CSC 1201	9	MCS 7202 MCS 7220	6
N.T. Rwangoga (MSc, A. Lecturer)	Software En- gineering, Ar- tificial Intelli- gence, Infor- mation Secu- rity	CSC 3101 CSC 1100	12	CSC 1100 CSC 2112	12	CSC 2204 CSC 2206	12	CSC 1204 CSC 3204	12
F Tushabe (MSc A. Lecturer)	Optimization, Image Pro- cessing, Programing, Foreinsics	CSC 1103 CSC 3102	12	CSC 3111 CSC 3100	12	CSC 2208 CSC 1204	12	CSC 3111 CSC 1206	12
R Ssekibuule (MSc A. Lecturer)	Programing, Security, Distributed Systems	CSC 3103 CSC 2102	12	CSC 3103 CSC 3115	12	CSC 1200 CSC 1202	12	CSC 2209 CSC 3205	12

Staff	Specialisation	Old	Ld	New	Ld	Old	Ld	New	Ld
J. Kizito (MSc A Lecturer)	Programing, Theoretical computing, Compiler Theory, Mathematical Computing	CSC 2100 CSC 3106	12	CSC 1106 CSC 2100	12	CSC 3203 CSC 2201	12	CSC 2210 CSC 1209	12
E. Beinomugisha (MSc A. Lecturer)	Image Pro- cessing, Theoretical Computing, Networked Systems	CSC 2101 CSC 1101	12	CSC 2105 CSC 2114	12	CSC 3200 CSC 2200	12	BSE 2203 BIT 3204	12
R. Azairwe (MSc A. Lecturer)	Programing, Optimiza- tion, Arti- ficial Intelli- gence	CSC 1102 CSC 2103	12	CSC 1105 CSC 2214	12	CSC 1203 CSC 2203	12	BSE 3202 CSC 2214	12
R. Nakibuule (MSc T. Assistant)	Theoretical computing, Computer Graphics, Image pro- cessing	CSC 2104 CSC 1102	12	CSC 1104 CSC 3105	12	CSC 2202 BIT 3204	12	CSC 1207 CSC 2200	12
D. Tuheirwe (BSc T. Assistant)	Programing, Theoretical comput- ing, Health Informatics	DCS 1103 DCS 1105	12	DCS 1103 DCS 1107	12	DCS 1204	12	DCS 1208 DCS 1206	12
J.E. Agaba (BSc T. Assistant)	Programing, Software De- velopment, Database Systems	DCS 1104 DCS 1200	12	DCS 1108	6	DCS 1200 DCS 1202	12	DCS 1205 DCS 1207	12