DEPARTMENT OF SOFTWARE ENGINEERING

Dr. Oluwasola S. Maitanmi
Head of Department

STAFF LIST

S/N	Name	Position	Area of Specialization
1.	Prof. Idowu Sunday A.	Professor	Software Engineering
2.	Prof. Joshua Vincent J.	Professor	Software Engineering
3.	Dr. Maitanmi S. Olusola	Associate Professor	Software Engineering
4.	Dr. Adekola Olubukola D.	Associate Professor	Software Engineering
5.	Dr Eweoya Ibukun O.	Senior Lecturer	Artificial Intelligence
6.	Dr. Ajayi Wumi S.	Lecturer I	Software Engineering
7.	Mrs. Okesola Kikelomo I.	Lecturer I	Bio-informatics
8.	Dr. Adetunji Oluwatofunmi O.	Lecturer I	Software Engineering
9.	Dr. Adigun Taiwo O.	Lecturer II	Computational Biology & Bioinformatics
10.	Dr. Mensah Yaw A.	Assistant Lecturer	Software Engineering
11.	Mr. Adegbola Adesoji A.	Assistant Lecturer	Software Engineering
12.	Mr. Mgbeahuruike, Emmanuel O.	Assistant Lecturer	Artificial Intelligence
13.	Mr. Adebanjo Adedoyin S.	Assistant Lecturer	Software Engineering
14.	Mr. Adeoti Babajide E.	Assistant Lecturer	Software Engineering
15.	Mr. Awoniyi Amos T.	Assistant Lecturer	Software Engineering
16.	Mrs. Adebanjo Olawunmi A.	Assistant Lecturer	Artificial Intelligence

ADJUNCT STAFF

S/N	Name	Position	Area of Specialization
1.	Prof. Akinola Solomon O.	Professor/Adjunct	Information Systems
2.	Dr. Odule Tola	Associate Professor/Adjunct	Information Security
3.	Dr. Kanu Richmond	Associate Professor/Adjunct	Functional Analysis
4.	Dr. Onuiri Ernest	Senior Lecturer/Adjunct	Biomedical Informatics &Machine Learning
5.	Dr. Akinsola Jide	Senior Lecturer/Adjunct	Artificial Intelligence
6.	Dr. Ajayi Olutayo	Senior Lecturer/Adjunct	Artificial Intelligence
7.	Aaron Izang	Lecturer I	Data Analytics and Business Intelligence
8.	Dr. Ayoade Akintayo	Lecturer I	Machine Learning and Cybersecurity
9.	Mr. Ayeni Kenneth	Assistant Lecturer	Software Engineering
10.	Mr. Amusa Afolarin	Assistant Lecturer	Computer Security
11.	Mr. Titiloye Samuel Olalekan	Assistant Lecturer	Software Engineering
12.	Miss Famodimu Oluwasefunmi	Assistant Lecturer	Machine Learning
13.	Mr. Oladipo Sunday	Assistant Lecturer	Artificial Intelligence
14.	Mr. Adelowo Opeyemi	Assistant Lecturer	Cyber security

NON-ACADEMIC STAFF LIST

S/N	Name	Position
1.	Mrs. Fabiyi Oluwatosin	Laboratory Technologist
2.	Mrs. Oladipo Abiodun	Office Manager
3.	Mr. Popoola Temitope	Office Assistant

CORE CURRICULUM AND MINIMUM ACADEMIC STANDARDS (CCMAS)
BSc. (Hons) SOFTWARE ENGINEERING

Overview

The software development industry presents huge opportunities within the context of an expanding global economy that is increasingly becoming digital. With the enormous potentials of this sector of the economy and the ever-increasing need for large and complex software systems, there is great promise to grow a large crop of software engineers as a force for sustainable socio-economic development. In addition to its core Computer Science foundation, Software Engineering also involves human and technical processes, and therefore borrows and adapts from the field of project management as well as from traditional engineering practice.

Philosophy

The philosophy of Software Engineering focuses on producing graduates who have the required knowledge and skills to develop and maintain quality software systems of scale for governments, organisations and businesses that adequately fulfil the functional and non-functional requirements of the systems within time and budget constraints.

Objectives

The specific objectives of the Software Engineering programme for students are to:

1. provide them a solid foundation in computing in such areas as problem solving, algorithm design, data structures and programming basics;
2. demonstrate practical skills in requirements analysis, system design, software architecture, software metrics, verification and validation, and the software engineering process in general for the production of high quality software-based systems;
3. demonstrate expertise in programming in a number of different languages with emphasis on the production of robust, reliable, cost-effective and secure systems that are based on sound design and development principles;
4. train them to be able to effectively and efficiently manage the development of large, complex and critical software; and
5. enable them to have the requisite knowledge and skill base as well as adequate practical exposure and high ethical standards for the limitless professional career opportunities (including self-employment) in the software industry.

Unique Features of The Programme

Special efforts have been made to tailor the programme to the rapidly evolving software industry in Nigeria in particular and Africa in general especially in the following areas:

1. Development of skilled software engineers in mobile applications and web development
2. Rigorous training on how to effectively manage software projects in highly challenging circumstances
3. Grooming of software engineers with specialised skills in Software Engineering but very broad knowledge on the entire computing discipline.

Employability Skills

The critical importance and increasing proliferation of software systems in every aspect of human endeavour make it mandatory for today’s software engineers to have all the necessary employability skills they require in today's competitive world. They include communication, teamwork and collaboration, negotiation and persuasion, problem solving, leadership, organisation, perseverance, motivation, confidence and the ability to work under pressure.

21st Century Skills
Among the 21st Century skills for the programme are:
1. Creative thinking;
2. information literacy;
3. media literacy;
4. flexibility;
5. social skills;
6. Problem solving,
7. Social skills; and
8. Innovation skills.

Basic Admission Requirements and Expected Duration of the Programmes
There are three different pathways by which candidates can be admitted into programmes in the discipline:
1. Unified Tertiary Matriculation Examination (UTME)
2. Direct Entry
3. Inter-University Transfer Mode

Unified Tertiary Matriculation Examination (UTME) Pathway
In addition to appropriate UTME score, a candidate must possess five Senior Secondary Certificate (SSC)-credit passes including English Language, Mathematics, Physics and any other relevant Science subjects in not more than two sittings.

Direct Entry (3-Year Degree Programme)
A minimum of a credit at the University/National Diploma or NCE with other five Senior School Certificate (SSC) credit passes in relevant science subjects three (3) of which must be in English Language, Mathematics, Physics.

Inter-University Transfer Mode
Students can transfer into 200-Level courses provided they have the relevant qualification. Universities are to certify that students meet the minimum requirements for the inter-university transfer.

Minimum duration
The minimum duration of computing programmes is four (4) academic sessions or eight (8) consecutively-run semesters for candidates who enter through the UTME Mode. Direct Entry candidates admitted into the 200 level of their programmes will spend a minimum of three academic sessions or six (6) consecutively-run semesters.

Graduation requirements
To be eligible for the award of the Bachelor degree in any of the Computing degree programme, a student must have:

1. passed all the core courses, university and faculty/school required courses and electives;
2. accumulated a minimum of 120 course units for students admitted through UTME and 90 course units for students admitted to 200 level
3. completed successfully students' industrial training (SIWES), seminar and research project.

To graduate, a student must be found worthy in character throughout the period of his/her studentship and must accumulate the total units prescribed for the programme from Core, Faculty and General Studies courses as well as SIWES, Seminar and Final Year Project.``````````

Requirement for Graduation
Minimum requirement of 156 credits are needed for the award of the B.Sc. degree in Software Engineering. Direct entry candidates earn less than the stipulated number. The distribution of the credit requirement by Level is as follows:

B.SC. SOFTWARE ENGINEERING

LEVEL	**1ST SEMESTER**	**2ND SEMESTER**	**TOTAL**
100	17	20	37
200	22	20	42
300	23	19	42
400	18	17	35
TOTAL	**80**	**76**	**156**

Course Structure: BSc. (Hons) SOFTWARE ENGINEERING

100 Level Courses

COURSE CODE	**COURSE TITLE**	**STATUS Core/Elective**	**SEMESTER**	
			1ST	**2ND**
BU-GST 011	Citizenship Orientation		0	
BU-GST 012	Citizenship Orientation			0
GST111	Communication in English	C	2	-
GST112	Nigerian Peoples and Culture	C	-	2
MTH101	Elementary Mathematics I	C	2	-
MTH102	Elementary Mathematics II	C	-	2
PHY101	General Physics I	C	2	-
PHY102	General Physics II	C	-	2
PHY107	General Practical Physics I	C	1	-
PHY108	General Practical Physics II	C	-	1
STA111	Descriptive Statistics	C	3	-
COS101	Introduction to Computing Sciences	C	3	-
COS102	Problem Solving	C	-	3
BU-SEN-102	Introduction to Web Technology and Development	C	-	3
STA 112	Probability I	C	-	3
BU-GST-105	Use of Library and Study Skill	C	2	-
BU-GST-126	Life and Teaching of Christ the Messiah	C	-	3
BU-GST-112	Health Principles	C	-	1
	International Certification Course			
BU-COS-107	Introduction to Scripting Languages	C	2	-
	TOTAL (37 UNITS)		**17**	**20**

BSc. (Hons) SOFTWARE ENGINEERING
200 Level Courses

COURSE CODE	COURSE TITLE	STATUS Core/Elective	SEMESTER	
			1ST	2ND
BU-GST 021	Citizenship Orientation		0	
BU-GST 022	Citizenship Orientation			0
GST212	Philosophy, Logic and Human Existence	C	-	2
ENT211	Entrepreneurship and Innovation	C	2	-
MTH 201	Mathematical Methods I	C	2	-
MTH 202	Elementary Differential Equations	C	-	2
COS201	Computer Programming I	C	3	-
COS202	Computer Programming II	C	-	3
SEN 201	Introduction to Software Engineering	C	2	-
CSC 203	Discrete Structures	C	2	-
INS204	System Analysis and Design	C	-	3
IFT211	Digital Logic Design	C	2	-
IFT212	Computer Architecture and Organisation	C	-	2
BU-SEN-212	Internet Technology Web Application Development	C	-	3
BU-GST-215	Adventist Heritage	C	3	-
BU-GST-200	Communication in French	C	-	1
BU-GST-221	Introduction to Agriculture	C	1	-
BU-GST-220	Origins and Science	C	-	1
	International Certification Course			
BU-COS-209	Innovations in Web Design and Development	C	2	-
	TOTAL (36 UNITS)		**19**	**17**

BSc. (Hons) SOFTWARE ENGINEERING
200 Level Courses
200 LEVEL DIRECT ENTRY

COURSE CODE	**COURSE TITLE**	**STATUS Core/Elective**	**SEMESTER**	
			1ST	**2ND**
BU-GST 021	Citizenship Orientation		0	
BU-GST 022	Citizenship Orientation			0
GST212	Philosophy, Logic and Human Existence	C	-	2
ENT211	Entrepreneurship and Innovation	C	2	-
MTH 201	Mathematical Methods I	C	2	-
MTH 202	Elementary Differential Equations	C	-	2
COS101	Introduction to Computing Sciences	C	3	-
COS102	Problem Solving	C	-	3
SEN 201	Introduction to Software Engineering	C	2	-
CSC 203	Discrete Structures	C	2	-
INS204	System Analysis and Design	C	-	3
IFT211	Digital Logic Design	C	2	-
IFT212	Computer Architecture and Organisation	C	-	2
BU-SEN 212	Internet Technology Web Application Development	C	-	3
BU-GST 215	Adventist Heritage	C	3	-
BU-GST 200	Communication in French	C	-	1
BU-GST 221	Introduction to Agriculture	C	1	-
BU-GST 220	Origins and Science	C	-	1
	International Certification Course			
BU-COS 209	Innovations in Web Design and Development	C	2	-
	TOTAL (36 UNITS)		**19**	**17**

BSc. (Hons) SOFTWARE ENGINEERING
300 Level Courses

COURSE CODE	COURSE TITLE	STATUS Core/Elective	SEMESTER	
			1ST	2ND
BU-GST 031	Citizenship Orientation		0	
BU-GST 032	Citizenship Orientation			0
GST 312	Peace and Conflict Resolution	C	-	2
ENT 312	Venture Creation	C	-	2
SEN 301	Object-Oriented Analysis and Design	C	2	-
SEN 304	Software Testing and Quality Assurance	C	-	2
SEN 306	Software Construction	C	-	2
SEN 322	Software Engineering Innovation and New Technology	C	-	2
SEN 350	SIWES	C		6
CSC 301	Data Structures	C	3	-
CSC 308	Operating Systems	C	-	3
BU-CSC 333	Database Systems Design, Implementation and Mgt.	C	2	-
BU-CSC 307	Linux System Administration	C	3	-
BU-SEN 303	Introduction to Big Data Engineering	C	2	-
BU-GST 317	Fundamentals of Christian Faith	C	3	-
BU-SEN 313	Mobile Application Development	C	2	
CSC 309	Artificial Intelligence	C	2	-
BU-GST-312	Family Life	C	-	1
	International Certification Course			
BU-COS 325	Introduction to Machine Learning	C	2	-
	TOTAL (42 UNITS)		**21**	**20**

SIWES holds during the second semester of 300L

BSc. (Hons) SOFTWARE ENGINEERING
300 Level Courses
300 LEVEL DIRECT ENTRY

COURSE CODE	**COURSE TITLE**	**STATUS Core/Elective**	**SEMESTER**	
			1ST	**2ND**
BU-GST 031	Citizenship Orientation		0	
BU-GST 032	Citizenship Orientation			0
GST 312	Peace and Conflict Resolution	C	-	2
ENT 312	Venture Creation	C	-	2
COS201	Computer Programming I	C	3	-
COS202	Computer Programming II	C	-	3
SEN 301	Object-Oriented Analysis and Design	C	2	-
SEN304	Software Testing and Quality Assurance	C	-	2
SEN306	Software Construction	C	-	2
SEN322	Software Engineering Innovation and New Technology	C	-	2
SEN350	SIWES	C		6
CSC301	Data Structures	C	3	-
CSC308	Operating Systems	C	-	3
BU-CSC-333	Database Systems Design, Implementation and Mgt.	C	2	-
BU-CSC-307	Linux System Administration	C	3	-
BU-SEN-303	Introduction to Big Data Engineering	C	2	-
BU-SEN-313	Mobile Application Development	C	2	
CSC 309	Artificial Intelligence	C	2	-
BU-GST-317	Fundamentals of Christian Faith	C	3	-
BU-GST-312	Family Life	C	-	1
	International Certification Course			
BU-COS-325	Introduction to Machine Learning	C	2	-
	TOTAL (48 UNITS)		**24**	**23**

SIWES holds during the second semester of 300L

BSc. (Hons) SOFTWARE ENGINEERING
400 Level Courses

COURSE CODE	COURSE TITLE	STATUS Core/Elective	SEMESTER 1ST	SEMESTER 2ND
BU-GST 041	Citizenship Orientation		0	
BU-GST 042	Citizenship Orientation			0
COS 409	Research Methodology and Technical Report Writing	C	3	-
SEN401	Software Configuration Management and Maintenance	C	2	-
SEN410	Software Architecture and Design	C	-	2
SEN490	Final Year Student's Project	C	-	6
INS 401	Project Management	C	2	-
BU-SEN-406	Formal Methods in Software Engineering	C	-	2
BU-SEN-407	Software Measurement and Metrics	C	2	-
BU-SEN-411	Open-Source Systems Development	C	2	-
BU-SEN-417	Human Computer Interaction and Emerging Technologies	C	3	-
BU-SEN-418	Professional Ethics and Practices	C	-	2
CSC 401	Algorithms and Complexity Analysis	C	2	-
*CYB 402	Steganography-Access Methods and Data Hiding	C	-	2
*IFT 410	System Integration and Architecture	C	-	2
BU-GST-400	Religion and Social Ethics	C	-	3
	International Certification Course			
BU-COS-419	Agile Development and Scrum	C	2	-
	TOTAL (37 UNITS)		**18**	**19**

BSc. (Hons) SOFTWARE ENGINEERING
Learning Outcomes and Course Contents

100 Level

GST 111: Communication in English (2 Units C: LH 15; PH 45)

Learning Outcomes

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

1. identify possible sound patterns in English language;
2. list notable language skills;
3. classify word formation processes;
4. construct simple and fairly complex sentences in English;
5. apply logical and critical reasoning skills for meaningful presentations;
6. demonstrate an appreciable level of the art of public speaking and listening; and
7. write simple and technical reports.

Course Contents

Sound patterns in English Language (vowels and consonants, phonetics and phonology). English word classes (lexical and grammatical words, definitions, forms, functions, usages, collocations). Sentence in English (types: structural and functional, simple and complex). Grammar and Usage (tense, mood, modality and concord, aspects of language use in everyday life). Logical and Critical Thinking and Reasoning Methods (Logic and Syllogism, Inductive and Deductive Argument and Reasoning Methods, Analogy, Generalisation and Explanations). Ethical considerations, Copyright Rules and Infringements. Writing Activities: (Pre-writing, writing, post writing, editing and proofreading; brainstorming, outlining, paragraphing. Types of writing, Summary, Essays, Letter, Curriculum Vitae, Report writing, Note making, etc. Mechanics of writing). Comprehension Strategies: (Reading and types of Reading, Comprehension Skills, 3RsQ). Information and Communication Technology in modern language learning. Language skills for effective communication. Major word formation processes. Writing and reading comprehension strategies. Logical and critical reasoning for meaningful presentations. Art of public speaking and listening. Report writing.

GST 112: Nigerian Peoples and Culture (2 Units C: LH 30)

Learning Outcomes

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

1. analyse the historical foundation of the Nigerian culture and arts in pre-colonial times;
2. list and identify the major linguistic groups in Nigeria;
3. explain the gradual evolution of Nigeria as a political unit;
4. analyse the concepts of Trade, Economic and Self-reliance status of the Nigerian peoples towards national development;
5. enumerate the challenges of the Nigerian State towards Nation building;
6. analyse the role of the Judiciary in upholding people’s fundamental rights;
7. identify acceptable norms and values of the major ethnic groups in Nigeria; and
8. list and suggest possible solutions to identifiable Nigerian environmental, moral and value problems.

Course Contents
Nigerian history, culture and art up to 1800 (Yoruba, Hausa and Igbo peoples and culture; peoples and culture of the ethnic minority groups). Nigeria under colonial rule (advent of colonial rule in Nigeria; Colonial administration of Nigeria). Evolution of Nigeria as a political unit (amalgamation of Nigeria in 1914; formation of political parties in Nigeria; Nationalist movement and struggle for independence). Nigeria and challenges of nation-building (military intervention in Nigerian politics; Nigerian Civil War). Concept of trade and economics of self-reliance (indigenous trade and market system; indigenous apprenticeship system among Nigeria people; trade, skill acquisition and self-reliance). Social justice and national development (law definition and classification). Judiciary and fundamental rights. Individual, norms and values (basic Nigeria norms and values, patterns of citizenship acquisition; citizenship and civic responsibilities; indigenous languages, usage and development; negative attitudes and conducts. Cultism, kidnapping and other related social vices). Re-orientation, moral and national values (The 3R's – Reconstruction, Rehabilitation and Re-orientation) Re-orientation Strategies: Operation Feed the Nation (OFN), Green Revolution, Austerity Measures, War Against Indiscipline (WAI), War Against Indiscipline and Corruption (WAIC), Mass Mobilisation for Self-Reliance, Social Justice and Economic Recovery (MAMSER), National Orientation Agency (NOA). Current socio-political and cultural developments in Nigeria.

MTH 101: Elementary Mathematics I (Algebra and Trigonometry) (2 Units C: LH 30)

Learning Outcomes
At the end of the course, students should be able to:
1. explain basic definition of Set, Subset, Union, Intersection, Complements and use of Venn diagrams;
2. solve quadratic equations;
3. solve trigonometric functions;
4. identify the various types of numbers; and
5. solve some problems using Binomial theorem.

Course Contents
Elementary set theory, subsets, union, intersection, complements, Venn diagrams. Real numbers; integers, rational and irrational numbers, mathematical induction, real sequences and series, theory of quadratic equations, binomial theorem. Complex numbers; algebra of complex numbers; the Argand diagram. De-Moivre's theorem, nth roots of unity. Circular measure, trigonometric functions of angles of any magnitude, addition and factor formulae.

MTH 102: Elementary Mathematics II (Calculus) (2 Units C: LH 30)

Learning Outcomes
At the end of the course, students should be able to:
1. distinguish types of rules in Differentiation and Integration;
2. describe the meaning of Function of a real variable, graphs, limits and continuity; and
3. solve some applications of definite integrals in areas and volumes.

Course Contents
Function of a real variable, graphs, limits and idea of continuity. The derivative, as limit of rate of change. Techniques of differentiation. Extreme curve sketching; Integration as an inverse of differentiation. Methods of integration, Definite integrals. Application to areas, volumes.

PHY 101: General Physics I (Mechanics) (2 Units C: LH 30)

Learning Outcomes
At the end of the course, students should be able to:
1. identify and deduce the physical quantities and their units;
2. differentiate between vectors and scalars'
3. describe and evaluate motion of systems on the basis of the fundamental laws of mechanics;
4. apply Newton's laws to describe and solve simple problems of motion;
5. evaluate work, energy, velocity, momentum, acceleration, and torque of moving or rotating objects;
6. explain and apply the principles of conservation of energy, linear and angular momentum;
7. describe the laws governing motion under gravity; and
8. explain motion under gravity and quantitatively determine behaviour of objects moving under gravity.

Course Contents
Space and time. Units and dimension, Vectors and Scalars, Differentiation of vectors. Displacement, velocity and acceleration. Kinematics. Newton laws of motion (Inertial frames, Impulse, force and action at a distance, momentum conservation). Relative motion. Application of Newtonian mechanics. Equations of motion. Conservation principles in physics, Conservative forces, conservation of linear momentum, Kinetic energy and work, Potential energy, System of particles, Centre of mass. Rotational motion. Torque, vector product, moment, rotation of coordinate axes and angular momentum. Polar coordinates. Conservation of angular momentum. Circular motion. Moments of inertia, gyroscopes and precession. Gravitation: Newton's Law of Gravitation, Kepler's laws of planetary motion, Gravitational potential energy, Escape velocity, Satellites motion and orbits.

PHY 102: General Physics II (Electricity & magnetism) (2 Units C: LH 30)

Learning Outcomes
At the end of the course, students should be able to:
1. describe the electric field and potential, and related concepts, for stationary charges;
2. calculate electrostatic properties of simple charge distributions using Coulomb's law, Gauss's law, and electric potential;
3. describe and determine the magnetic field for steady and moving charges;
4. determine the magnetic properties of simple current distributions using Biot-Savart and Ampere's law;
5. describe electromagnetic induction and related concepts and make calculations using Faraday and Lenz's laws;
6. explain the basic physical of Maxwell's equations in integral form;

7. evaluate DC circuits to determine the electrical parameters;
8. determine the characteristics of ac voltages and currents in resistors, capacitors, and Inductors.

Course Contents
Forces in nature. Electrostatics (electric charge and its properties, methods of charging). Coulomb's law and superposition. Electric field and potential. Gauss's law. Capacitance. Electric dipoles. Energy in electric fields. Conductors and insulators. DC circuits (current, voltage and resistance. Ohm's law. Resistor combinations. Analysis of DC circuits. Magnetic fields. Lorentz force. Biot-Savart and Ampère's laws. Magnetic dipoles. Dielectrics. Energy in magnetic fields. Electromotive force. Electromagnetic induction. Self and mutual inductances. Faraday and Lenz's laws. Step up and step down transformers. Maxwell's equations. Electromagnetic oscillations and waves. AC voltages and currents applied to inductors, capacitors, and resistance.

PHY 107: General Practical Physics I (1 Unit C: PH 45)

Learning Outcomes
At the end of the course, students should be able to:
1. conduct measurements of some physical quantities;
2. make observations of events, collect and tabulate data;
3. identify and evaluate some common experimental errors;
4. plot and analyse graphs; and
5. draw conclusions from numerical and graphical analysis of data.

Course Contents
This introductory course emphasizes quantitative measurements, the treatment of measurement errors and graphical analysis. A variety of experimental techniques should be employed. The experiments include studies of meters, the oscilloscope, mechanical systems, electrical and mechanical resonant systems, light, heat, viscosity etc., covered in PHY 101 and PHY 102. However, emphasis should be placed on the basic physical techniques for observation, measurements, data collection, analysis and deduction.

PHY 108 - General Practical Physics II (1 Unit C: PH 45)

Learning Outcomes
On completion, the student should be able to:
1. conduct measurements of some physical quantities;
2. make observations of events, collect and tabulate data;
3. identify and evaluate some common experimental errors;
4. plot and analyse graphs;
5. draw conclusions from numerical and graphical analysis of data; and
6. prepare and present practical reports.

Course Contents
This practical course is a continuation of PHY 107 and is intended to be taught during the second semester of the 100 level to cover the practical aspect of the theoretical courses that have been covered with emphasis on quantitative measurements, the treatment of measurement errors, and graphical analysis. However, emphasis should be

placed on the basic physical techniques for observation, measurements, data collection, analysis and deduction.

STA 111: Descriptive statistics: (3 Units C: LH 45)

Learning Outcomes
At the end of the course, students should be able to:
1. explain the basic concepts of descriptive statistics.
2. present data in graphs and charts.
3. differentiate between measures of location, dispersion and partition.
4. describe the basic concepts of Skewness and Kurtosis as well as their utility function in a given data set.
5. differentiate rates from ratio and how they are use.
6. compute the different types of index number from a given data set and interpret the output.

Course content
Statistical data. Types, sources and methods of collection. Presentation of data. Tables chart and graph. Errors and approximations. Frequency and cumulative distributions. Measures of location, partition, dispersion, skewness and Kurtosis. Rates, ratios and index numbers.

STA 112: Probability I (3 Units C: LH 45)

Learning Outcomes
At the end of the course students should be able to
1. explain the differences between permutation and combination;
2. explain the concept of random variables and relate it to probability and distribution functions;
3. describe the basic distribution functions; and
4. explain the concept of exploratory data analysis.

Course Contents
Permutation and combination. Concepts and principles of probability. Random variables. Probability and distribution functions. Basic distributions: Binomial, geometric, Poisson, normal and sampling distributions; exploratory data analysis.

COS 101: Introduction to Computing Sciences (3 Units C: LH 30; PH 45)

Learning Outcomes
At the end of the course, students should be able to:
1. explain basic components of computers and other computing devices;
2. describe the various applications of computers;
3. explain information processing and its roles in the society;
4. describe the Internet, its various applications and its impact;
5. explain the different areas of the computing discipline and its specializations; and
6. demonstrate practical skills on using computers and the internet.

Course Contents
Brief history of computing. Description of the basic components of a computer/computing device. Input/Output devices and peripherals. Hardware, software and human ware. Diverse and growing computer/digital applications. Information processing and its roles in society. The Internet, its applications and its impact on the world today. The different areas/programs of the computing discipline. The job specializations for computing professionals. The future of computing.

Lab Work: Practical demonstration of the basic parts of a computer. Illustration of different operating systems of different computing devices including desktops, laptops, tablets, smart boards and smart phones. Demonstration of commonly used applications such as word processors, spreadsheets, presentation software and graphics. Illustration of input and output devices including printers, scanners, projectors and smartboards. Practical demonstration of the Internet and its various applications. Illustration of browsers and search engines. How to access online resources.

COS 102: Problem Solving (3 Units C: LH 30; PH 45)

Learning Outcomes
At the end of this course, students should be able to:
1. explain problem solving processes;
2. demonstrate problem solving skills;
3. describe the concept of algorithms development and properties of algorithms;
4. discuss the solution techniques of solving problem;
5. solve computer problems using algorithms, flowcharts, pseudocode; etc.; and
6. solve problems using programming language using C, PYTHON, etc.

Course Contents
Introduction to the core concepts of computing. Problems and problem-solving. The identification of problems and types of problems (routine problems and non-routine problems). Method of solving computing problems (introduction to algorithms and heuristics). Solvable and unsolvable problems. Solution techniques of solving problems (abstraction, analogy, brainstorming, trial and error, hypothesis testing, reduction, literal thinking, means-end analysis, method of focal object, morphological analysis, research, root cause analysis, proof, divide and conquer). General Problem-solving process. Solution formulation and design: flowchart, pseudocode, decision table, decision tree. Implementation, evaluation and refinement. Programming in C, Python etc.

Lab Work: Use of simple tools for algorithms and flowcharts; writing pseudocode; writing assignment statements, input-output statements and condition statements; demonstrating simple programs using any programming language (Visual Basic, Python, C)

BU-COS-107 Introduction to Scripting Languages (2 Units; C: LH=30)
Learning Outcomes
On completion of this course, students should be able to:
1. Describe at least six (6) scripting languages
2. Explain at least four (4) types of scripting languages
3. List at least 5 characteristics of scripting languages

4. Clarify visual scripting and scripting components
5. Summarise four (4) main areas of usage of scripting languages
6. Illuminate web scripting

Course Contents

Origin of Scripting Languages (SLs). Basic concepts of SLs. Scripts and programs. Types of scripting languages. Advantages and disadvantages SL. Development environment for SL. Front-end and Back-end SLs. Characteristics of SLs. Classification of SLs and users. Visual scripting. Scripting components. Applications of traditional SL. Applications of contemporary SLs. Command SLs. Mark-Up SLs. Universal SLs. Applications developed using SLs. Concepts of Web scripting. Dynamic web pages. Dynamically generated HTML.

BU-SEN-102 Introduction to Web Technology and Development (3 Units; C: LH=30; PH=45)

Learning Outcomes

On completion of this course, students should be able to:

1. Differentiate between the concept of the Internet and the Web
2. List at least five (5) various Internet and Web protocols services you know
3. Describe Uniform Resource Locator (URL) as discussed
4. Summarise the concepts of Web browsers and Web Servers and the Types
5. Reproduce the process of HTML and JavaScript in creating Interactive form
6. List two (2) the stages of using CSS to style web pages

Course Contents

Introduction to computer networks. The Internet and the Web. Web Layout. Web Protocols. Use of Browsers and Web Servers. Forms and Data. Use of HTML/XHTML to create forms. Tables. Cascading Style Sheet (CSS) Rules. Web media. Uniform Resource Locator. Fundamental of JavaScript. Using JavaScript to create Interactive pages. Developing simple web forms with HTML/XHTML**.** Using web tools to design web forms. Using CSS for web page styling. Use of CMS Frameworks.

Minimum Academic Standard

Software Laboratory

200 Level

GST 212: Philosophy, Logic and Human Existence (2 Units C: LH 30)

Learning Outcomes
At the end of this course, students should be able to:
1. know the basic features of philosophy as an academic discipline;
2. identify the main branches of philosophy & the centrality of logic in philosophical discourse;
3. know the elementary rules of reasoning;
4. distinguish between valid and invalid arguments;
5. think critically and assess arguments in texts, conversations and day-to-day discussions;
6. critically asses the rationality or otherwise of human conduct under different existential conditions;
7. develop the capacity to extrapolate and deploy expertise in logic to other areas of knowledge, and
8. guide his or her actions, using the knowledge and expertise acquired in philosophy and logic.

Course Contents
Scope of philosophy; notions, meanings, branches and problems of philosophy. Logic as an indispensable tool of philosophy. Elements of syllogism, symbolic logic— the first nine rules of inference. Informal fallacies, laws of thought, nature of arguments. Valid and invalid arguments, logic of form and logic of content — deduction, induction and inferences. Creative and critical thinking. Impact of philosophy on human existence. Philosophy and politics, philosophy and human conduct, philosophy and religion, philosophy and human values, philosophy and character molding, etc.

ENT 211: Entrepreneurship and Innovation (2 Units C: LH 15; PH 45)

Learning Outcomes
At the end of this course, students should be able to:
1. explain the concepts and theories of entrepreneurship, intrapreneurship, opportunity seeking, new value creation, and risk taking;
2. state the characteristics of an entrepreneur;
3. analyse the importance of micro and small businesses in wealth creation, employment, and financial independence;
4. engage in entrepreneurial thinking;
5. identify key elements in innovation;
6. describe stages in enterprise formation, partnership and networking including business planning;
7. describe contemporary entrepreneurial issues in Nigeria, Africa and the rest of the world; and
8. state the basic principles of e-commerce.

Course Contents
Concept of Entrepreneurship (Entrepreneurship, Intrapreneurship/Corporate Entrepreneurship). Theories, Rationale and relevance of Entrepreneurship (Schumpeterian and other perspectives, risk-taking, necessity and opportunity-based

entrepreneurship and creative destruction). Characteristics of Entrepreneurs (Opportunity seeker, risk taker, natural and nurtured, problem solver and change agent, innovator and creative thinker). Entrepreneurial thinking (Critical thinking, Reflective thinking, and Creative thinking). Innovation (Concept of innovation, Dimensions of innovation, Change and innovation, Knowledge and innovation). Enterprise formation, partnership and networking (Basics of business plan, Forms of business ownership, business registration and forming alliances and joint ventures). Contemporary Entrepreneurship Issues (knowledge, skills and technology, intellectual property, virtual office, networking). Entrepreneurship in Nigeria (Biography of inspirational entrepreneurs, youth and women entrepreneurship, Entrepreneurship support institutions, Youth enterprise networks and environmental and cultural barriers to entrepreneurship). Basic principles of e-commerce.

MTH 201: Mathematical Methods I (2 Units C: LH 30)

Learning Outcomes
At the end of the course students should be able to:
1. describe Real-valued functions of a real variable;
2. solve some problems using Mean value Theorem and Taylor Series expansion; and
3. evaluate Line Integral, Surface Integral and Volume Integrals.

Course Contents
Real-valued functions of a real variable. Review of differentiation and integration and their applications. Mean value theorem. Taylor series. Real-valued functions of two and three variables. Partial derivatives chain rule, extrema, Lagrangian multipliers. Increments, differentials and linear approximations. Evaluation of line, integrals. Multiple integrals.

MTH 202: Elementary Differential Equations (2 Units C: LH 30)

Learning Outcomes
At the end of the course, students should be able to:
1. define the following: order and degree of a differential equation;
2. describe some techniques for solving first and second order linear and non-linear equations; and
3. solve some problems related to geometry and physics.

Course Contents
Derivation of differential equations from primitive, geometry, physics, etc. order and degree of differential equation. Techniques for solving first and second order linear and non-linear equations. Solutions of systems of first order linear equations. Finite linear difference equations. Application to geometry and physics.

COS 201: Computer Programming, I (3 Units C1: LH 30; PH 45)

Learning Outcomes
At the end of this course, students should be able to:
1. explain the principles of good programming and structured programming concepts;

2. explain the programming constructs, syntax and semantics of a higher-level language;
3. describe the chosen programming language variables, types, expressions, statements and assignment; simple input and output;
4. describe the programme control structures, functions and parameter passing, and structured decomposition; and
5. develop simple programs in the taught programming language as well as debug and test them.

Course Contents
Essentials of computer programming. Types of programming: Functional programming, Declarative programming, Logic programming, object-oriented programming. Scripting languages, structured programming principles. Basic data types, variables, expressions, assignment statements, and operators. Basic object-oriented concepts: abstraction, objects, classes, methods; parameter passing; encapsulation. Class hierarchies and programme organisation using packages/namespaces. Use of API – use of iterators/enumerators, List, Stack, Queue from API. Searching; sorting; Recursive algorithms. Event-driven programming: event-handling methods; event propagation; exception handling. Introduction to Strings and string processing. Simple I/O; control structures; Arrays. Simple recursive algorithms, inheritance, polymorphism.

Lab work: Programming assignments; design and implementation of simple algorithms e.g. average, standard deviation, searching and sorting. Developing and tracing simple recursive algorithms. Inheritance and polymorphism.

COS 202: Computer Programming II (3 Units C: LH 30; PH 45)

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

1. demonstrate the principles of good programming and structured programming concepts;
2. demonstrate string processing, internal searching, sorting, and recursion;
3. demonstrate the basic use of OOP concepts: classes, objects, inheritance, polymorphism, data abstraction;
4. apply the tools for developing, compiling, interpreting and debugging programs; and
5. demonstrate the use of syntax and data objects, operators. Central flow constructs, objects and classes programming, Arrays, methods, Exceptions, Applets and the Abstract, OLE, Persistence, Window Toolkit.

Course Contents
Review and coverage of advanced object-oriented programming - polymorphism, abstract classes and interfaces; Class hierarchies and program organisation using packages/namespaces; Use of API – use of iterators/enumerators, List, Stack, Queue from API; Searching; sorting; Recursive algorithms; Event-driven programming: event-handling methods; event propagation; exception handling. Applications in Graphical User Interface (GUI) programming.

Lab work: Programming assignments leading to extensive practice in problem solving and program development with emphasis on object-orientation. Solving basic problems using static and dynamic data structures. Solving various searching and sorting algorithms using iterative and recursive approaches. GUI programming.

SEN201: Introduction to Software Engineering (2 units C: LH 30)

Learning Outcomes

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

1. describe the concept of the software life cycle;
2. explain the phases of requirements analysis, design, development, testing and maintenance in a typical software life cycle;
3. differentiate amongst the various software development models;
4. utilise UML for object-oriented analysis and design;
5. describe different design architectures;
6. explain the various tasks involved in software project management; and
7. describe the basic legal issues related to Software Engineering.

Course Contents

Software Engineering concepts and principles. Design, development and testing of software systems. Software processes: software lifecycle and process models. Process assessment models. Software process metrics. Life cycle of software system. Software requirements and specifications. Software design. Software architecture. Software metrics. Software quality and testing. Software architecture. Software validation. Software evolution: software maintenance; characteristics of maintainable software; re-engineering; legacy systems; software reuse. Software Engineering and its place as a computing discipline. Software project management: team management; project scheduling; software measurement and estimation techniques; risk analysis; software quality assurance; software configuration management. Software Engineering and law.

SEN 299: Students Industrial Work Experience Scheme I (3 Units C: PH 135)

Learning Outcomes

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

1. explain how a typical software engineering firm operates;
2. describe the various assignments carried out and the skills acquired during the SIWES period; and
3. submit a comprehensive report on the knowledge acquired and the experience gained during the exercise.

Course Contents

Students are attached to private and public organisations for a period of three months during the second year session long break with a view to making them acquire practical experience and to the extent possible, develop skills in all areas of Software Engineering. Students are supervised during the training period and shall be expected to keep records designed for the purpose of monitoring their performance. They are also expected to submit a report on the experience gained and defend their reports.

CSC 203: Discrete Structures (2 Units C: LH 30)

Learning Outcomes
At the end of this course, the students will be able to:

1. convert logical statements from informal language to propositional and predicate logic expressions;
2. describe the strengths and limitations of propositional and predicate logic;
3. outline the basic structure of each proof technique (direct proof, proof by contradiction, and induction) described in this unit;
4. apply each of the proof techniques (direct proof, proof by contradiction, and induction) correctly in the construction of a sound argument;
5. apply the pigeonhole principle in the context of a formal proof;
6. compute permutations and combinations of a set, and interpret the meaning in the context of the particular application;
7. map real-world applications to appropriate counting formalisms, such as determining the number of ways to arrange people around a table, subject to constraints on the seating arrangement, or the number of ways to determine certain hands in cards (e.g., a full house); and
8. solve a variety of basic recurrence relations.

Course Contents
Propositional Logic. Predicate Logic. Sets. Functions. Sequences and Summation. Proof Techniques. Mathematical induction. Inclusion-exclusion and Pigeonhole principles. Permutations and Combinations (with and without repetitions). The Binomial Theorem. Discrete Probability. Recurrence Relations.

INS 204: Systems Analysis and Design (3 Units C: LH 30; PH 45)

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

1. describe system requirements gathering techniques;
2. explain data modelling technique (entity relationship modelling);
3. explain process modelling technique (data flow diagram);
4. describe system architectural design;
5. describe process and database design; and
6. explain user interface design.

Course Contents
Structured approach to analysis and design of information systems for businesses. Software development life cycle. Structured top-down and bottom-up design. Dataflow diagramming. Entity relationship modelling. Computer aided software engineering. Input and output, prototyping design and validation. File and database design. Design of user interfaces. Comparison of structured and object-oriented design

Lab Work: system requirements gathering techniques; data modelling techniques (entity relationship modelling); process modelling techniques (data flow diagram); use of UML diagrams; system architectural design; user interface design.

IFT 211: Digital Logic Design (2 Units C: LH 15; PH 45)

Learning Outcomes

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

1. explain why everything is data, including instructions, in computers;
2. describe how negative integers, fixed-length numbers, and non-numeric data are represented;
3. convert numerical data from one format to another;
4. describe computations as a system characterised by a known set of configurations with transitions from one unique configuration (state) to another (state);
5. describe the distinction between systems whose output is only a function of their input (combinational) and those with memory/history (sequential);
6. describe a computer as a state machine that interprets machine instructions;
7. articulate that there are many equivalent representations of computer functionality, including logical expressions and gates, and be able to use mathematical expressions to describe the functions of simple combinational and sequential circuits; and
8. design the basic building blocks of a computer: arithmetic-logic unit (gate-level), registers (gate-level), central processing unit (register transfer-level), and memory (register transfer-level).

Course Contents

Introduction to information representation and number systems. Boolean algebra and switching theory. Manipulation and minimisation of completely and incompletely specified Boolean functions. Physical properties of gates: fan-in, fan-out, propagation delay, timing diagrams and tri-state drivers. Combinational circuits design using multiplexers, decoders, comparators and adders. Sequential circuit analysis and design, basic flip-flops, clocking and timing diagrams. Registers, counters, RAMs, ROMs, PLAs, PLDs, and FPGAs.

Lab Work: Simple combinational gates (AND, OR, NOT, NAND, NOR); Combinational circuits design using multiplexers, decoders, comparators and adders. Sequential circuit analysis and design using basic flip-flops (S-R, J-K, D, T flip-flops); Demonstration of registers, counters, RAMs, ROMs, PLAs, PLDs, and FPGAs.

IFT 212: Computer Architecture and Organisation (3 Units C: LH 30; PH 45)

Learning Outcomes:

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

1. explain different instruction formats, such as addresses per instruction and variable length vs. fixed length formats;
2. describe the organisation of the classical von Neumann machine and its major functional units;
3. explain how subroutine calls are handled at the assembly level;
4. describe the basic concepts of interrupts and I/O operations;
5. write simple assembly language program segments;
6. show how fundamental high-level programming constructs are implemented at the machine-language level;
7. compare alternative implementation of data paths;

8. discuss the concept of control points and the generation of control signals using hardwired or micro-programmed implementations

Course Contents
Instruction format and types, memory and I/O instructions, dataflow, arithmetic, and flow control instructions, addressing modes, stack operations, and interrupts. Data path and control unit design. RTL, microprogramming, and hardwired control. Practice of assembly language programming. Memory hierarchy, cache memory, virtual memory. I/O fundamentals. Interrupt structures.

Lab work: Programming assignments to practice MS-DOS batch programming, Assembly Process, Debugging, Procedures, Keyboard input, Video Output, File and Disk I/O and Data Structure. Instruction and arithmetic pipelining, superscalar architecture. Reduced Instruction Set Computers. Parallel architectures and interconnection networks.

BU-COS-209 Innovation in Web Design and Development (2 units; Core; LH=15; PH=45)

Learning Outcomes
On completion of the course, students should be able to:
1. Explain the role of a Back-end developer
2. Describe the relationship between the Front-end and Back-end aspect of web development
3. Develop Server-side (back-end) applications using NodeJS JavaScript runtime
4. Discuss extension of NodeJS applications by adding MongoDB solutions to manage DBs
5. Perform CRUD operations on the Database.
6. Develop asynchronous callbacks or promises to complete asynchronous.

Course Contents
Overview of NodeJS. Installing NodeJS. Creating First NodeJS App. Setting up NodeJS. Exploring NodeJS Modules. Http Servers and Clients. Students First Express Application. Implementing the Mobile-First Paradigm. Data Storage and Retrieval. Interfacing with web development programming languages. Basics concepts of NodeJS. Advanced concepts of NodeJS. Building scalable server-side web applications with NodeJS. NodeJS ecosystem. MongoDB Documents. MongoDB Collections. MongoDB Replica for high availability. MongoDB Sharding for scalability. MongoDB Indexes to improve query speed. REST APIs. GraphQL APIs. DenoJS.

Minimum Academic Standard
Software Laboratory

BU-SEN-212 Internet Technologies and Web Applications Development (3 Units; C: LH=30; PH=45)

Learning Outcomes
On completion of this course, students should be able to:
1. Describe the concept of the Internet Web technologies

2. Define both Client and Server side technologies
3. Demonstrate the knowledge of installation and configuration of application stack namely WAMP, LAMP, XAMP as the case may be to show how Apache server work with PHP
4. Describe the knowledge of the use of basic PHP syntax and language constructs including PHP functions foe working with files
5. Justify with at least five (5) points on how to Connect PHP web application to mysql server
6. Explain the process of access and management of records database server with PHP

Course Contents
Networks Review and TCP/IP Overview. Internet Protocol Stack. Protocol Layering and Data. Application Layer and Client -server Application. Persistent Connections and Non-Persistent connections. Web applications development and Technologies. Programme Libraries and Frameworks. Client and Server–side technologies. Introduction to programming using PHP. Set up and installation of web server. Web applications development using PHP. PHP Functions-System and user functions. Web server and Handling of form data. Object Oriented programming using PHP. Creating and connecting to database in PHP. Managing records database within PHP-Writing MySQL query in PHP. Fetching results and getting data from more than one table. Adding and updating data. jQuery.XML and PHP. Practical real-world applications developments. Creating simple blogs. Web server set-up to work with PHP. Creating web pages with embedded PHP.

Minimum Academic Standard
Software Laboratory

DTS 204: Statistical Computing Inference and Modelling (3 Units C: LH 45)

Learning Outcomes
At the end of the course, the students should be able to:
1. make conclusions based on statistical assumptions, models and results;
2. make inference on statistical outcomes, and real-world implications and how these outcomes are factored into decision-making processes;
3. demonstrate the various considerations that are applied both for communicating statistical solutions to real problems;
4. make conclusions based on statistical models and results by applying a broad range of statistical tools and packages; and
5. demonstrate logical, meaningful skills that bothers not just on the relevance of the data that informed the statistical outcomes, but also on the real-world implications of how these outcomes are factored into decision-making processes.

Course Contents
Population and samples. Asymptotics. Statistical models and methodologies. Random sampling distributions. Elementary time series analysis. Index numbers. Demographic measures. Estimation (point and interval) and tests of hypotheses concerning population mean and proportion (one and two sample cases). Regression and correlation. Programming in Python computer language. Computation of mean,

variance and correlation. Sorting and ranking of data. Data Step Processing. Preparing Data for Analysis. Evaluating Quantitative Data. Sample Size Estimation. Basic statistical computing in regression analysis and the analysis of designed experiments. Introduction to Monte Carlo methods. Use of statistical packages like SPSS, SAS, Minitab, GENSTAT, EPI-INFO, SYSTAT.

Lab work: Practical experiments on statistical models and methodologies. Practical exercises on random sampling distribution methods. Practicals on test of hypothesis, population, mean, proportion, regression and correlation analysis. Exercise on how to sort and data from different data set. Use of SPSS for data analysis and computation.

300 Level

GST 312: Peace and Conflict Resolution (2 Units C: LH 30)

Learning Outcomes

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

1. analyse the concepts of peace, conflict and security;
2. list major forms, types and root causes of conflict and violence;
3. differentiate between conflict and terrorism;
4. enumerate security and peacebuilding strategies; and
5. describe roles of international organisations, media and traditional institutions in peace building.

Course Contents

Concepts of Peace, Conflict and Security in a multi-ethnic nation. Types and Theories of Conflicts: Ethnic, Religious, Economic, Geopolitical Conflicts; Structural Conflict Theory, Realist Theory of Conflict, Frustration-Aggression Conflict Theory. Root causes of Conflict and Violence in Africa: Indigene and Settlers Phenomenon; Boundaries/border disputes; Political disputes; Ethnic disputes and rivalries; Economic Inequalities; Social disputes; Nationalist Movements and Agitations; Selected Conflict Case Studies – Tiv-Junkun; Zango Kartaf, Chieftaincy and Land disputes, etc. Peace Building, Management of Conflicts and Security: Peace & Human Development. Approaches to Peace & Conflict Management (Religious, Government, Community Leaders, etc.). Elements of Peace Studies and Conflict Resolution: Conflict dynamics assessment Scales: Constructive & Destructive. Justice and Legal framework: Concepts of Social Justice; The Nigeria Legal System. Insurgency and Terrorism. Peace Mediation and Peace Keeping. Peace & Security Council (International, National and Local levels) Agents of Conflict resolution – Conventions, Treaties Community Policing: Evolution and Imperatives. Alternative Dispute Resolution, ADR. Dialogue b). Arbitration, c). Negotiation d). Collaboration, etc. Roles of International Organisations in Conflict Resolution. (a). The United Nations, UN and its Conflict Resolution Organs. (b). The African Union & Peace Security Council (c). ECOWAS in Peace Keeping. Media and Traditional Institutions in Peace Building. Managing Post-Conflict Situations/Crisis: Refugees. Internally Displaced Persons, IDPs. The role of NGOs in Post-Conflict Situations/Crisis.

ENT 312: Venture Creation (2 Units C: LH 15; PH 45)

Learning Outcomes

At the end of this course, students, through case study and practical approaches, should be able to:

1. describe the key steps in venture creation;
2. spot opportunities in problems and in high potential sectors regardless of geographical location;
3. state how original products, ideas, and concepts are developed;
4. develop business concept for further incubation or pitching for funding;
5. identify key sources of entrepreneurial finance;
6. implement the requirements for establishing and managing micro and small enterprises;
7. conduct entrepreneurial marketing and e-commerce;

8. apply a wide variety of emerging technological solutions to entrepreneurship; and
9. appreciate why ventures fail due to lack of planning and poor implementation.

Course Contents
Opportunity Identification (Sources of business opportunities in Nigeria, Environmental scanning, Demand and supply gap/unmet needs/market gaps/market research, Unutilised resources, Social and climate conditions, and technology adoption gap). New business development (business planning, market research). Entrepreneurial finance (venture capital, equity finance, microfinance, personal savings, small business investment organisations, and business plan competition). Entrepreneurial marketing and e-commerce (Principles of marketing, customer acquisition & retention, B2B, C2C and B2C models of e-commerce, first mover advantage, e-commerce business models and successful e-commerce companies,). Small business management/family business: Leadership & Management, basic bookkeeping, nature of family business and family business growth model. Negotiation and business communication (Strategy and tactics of negotiation/bargaining, traditional and modern business communication methods). Opportunity discovery demonstrations (business idea generation presentations, business idea contest, brainstorming sessions, idea pitching). Technological solutions (the concept of market/customer solution, customer solution, and emerging technologies, business applications of new technologies- Artificial Intelligence (AI), Virtual/Mixed Reality (VR), Internet of Things (IoT), Blockchain, Cloud Computing, renewable energy, etc. digital business and e-commerce strategies).

SEN 301: Object-Oriented Analysis and Design (2 Units C: LH 15; PH 45)

Learning Outcomes
At the end of this course, students should be able to:
1. explain the concept of the object-oriented approach to modelling;
2. describe the conceptual model of the UML-based software development life cycle;
3. demonstrate how to use the major UML diagrams for object-oriented analysis and design;
4. demonstrate the use of UML-based CASE tools.

Course Contents
Object-oriented approach to information system development, particularly in reference to the earlier stages of analysis and design. Importance of modelling, principles of modelling, object- oriented modelling, conceptual model of the Unified Modelling Language (UML), architecture, software development life cycle. The principles and basic concepts of object orientation and the different aspects of object-oriented modelling as represented by the UML technique. Case study of a typical UML-based CASE tool.

Lab Work: Practical exercises on different requirements specification and design activities; developing problem statements, SRS documents and Use Case Diagrams; designing UML Activity diagrams, UML Class diagrams and State Chart diagrams; drawing partial layered, logical architecture diagram with UML package diagram notation; Designing Component and Deployment diagrams.

SEN 304: Software Testing & Quality Assurance (2 Units C: LH 15; PH 45)

Learning Outcomes

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

1. state the critical importance of software testing in ensuring software quality;
2. explain the difference between validation and verification and their different techniques;
3. describe the concept of quality assurance and differentiate between process assurance and product assurance;
4. describe the different statistical approaches to quality control.

Course Contents

The importance of Software Testing. Understanding Verification and Validation. How to assure it and verify it, and the need for a culture of quality. Avoidance of errors and other quality problems. Inspections and reviews. Testing, verification and validation techniques. Process assurance vs. Product assurance. Quality process standards. Product and process assurance. Problem analysis and reporting. Statistical approaches to quality control

Lab Work: Debugging tools; unit testing – black box and white testing techniques; integration and system testing tools; other testing tools – performance testing, load testing, stress testing, regression testing, security testing; manual testing vs automated testing.

SEN 306: Software Construction (2 Units C: LH 15; PH 45)

Learning Outcomes

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

1. explain the importance of Software Construction and the key construction decisions;
2. describe the key issues in design including key design concepts, levels of design and Abstract Data Types (ADTs);
3. discuss best practices in dealing with routines, fundamental data types and different types of statements; and
4. describe how to ensure software quality through developer testing, debugging and software craftsmanship.

Course Contents

Definition of Software Construction; Its importance; Key construction decisions – choice of programming language, selection of major construction decisions. Design in construction – Key design concepts, levels of design, design heuristics. Abstract Data Types (ADTs). Working Classes. High Quality Routines. The Pseudo Code Programming Process. Fundamental Data Types – Numbers, Characters and Strings, Boolean Variables, Arrays, Tables. Types of Statements – Straight Line Code, Loops, Control Structures; Developer Testing and Debugging. Software Craftsmanship – Layout and Style, Documentation, Personal Character.

Lab Work: Practicals on the most common tools to ensure good software construction. The features include static code analysers to check that code follows coding conventions, special code searching and editing, collaboration support to allow multiple programmers working simultaneously, support for proper code documentation. Practice with IDEs (such as Visual Study Code, NetBeans and Eclipse) on debugging, compilation, running of code, auto completion and version control.

SEN 322: Software Engineering Innovation and New Technology (2 Units C: LH 15)

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

1. explain business models;
2. identify some entrepreneurial opportunities available in Software Engineering;
3. describe business plan and business startup process;
4. explain business feasibility and strategy;
5. explain marketing strategies; and
6. discuss business ethics and legal issues.

Course Contents
Software entrepreneurial process. Principles of software business ownership. Identifying software market opportunities. Entrepreneurial software marketing. Software business communication and negotiation techniques. Feasibility analysis. Entrepreneurial financing. Legal issues. Software business plan development. Risk management.

SEN 399: Students Industrial Work Experience Scheme II (3 Units C: PH 135)

Learning Outcomes
At the end of this training, students should be able to:

1. explain how a typical software engineering firm operates;
2. describe the various assignments carried out and the skills acquired during the SIWES period; and
3. submit a comprehensive report on the knowledge acquired and the experience gained during the exercise.

Course Contents
Students are attached to private and public organisations for a period of three months during the third year session long break with a view to making them acquire additional practical experience in all areas of Software Engineering over and above what is gained in SEN 299. Students are supervised during the training period and shall be expected to keep records designed for the purpose of monitoring their performance. They are also expected to submit a report on the experience gained and defend their reports.

CSC 301: Data Structures (3 Units C: LH 30; PH 45)

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

1. discuss the appropriate use of built-in data structures;
2. apply object-oriented concepts (inheritance, polymorphism, design patterns, etc.) in software design;
3. implement various data structures and their algorithms, and apply them in implementing simple applications;
4. choose the appropriate data structure for modelling a given problem;

5. analyse simple algorithms and determine their efficiency using big-O notation; and
6. apply the knowledge of data structures to other application domains like data compression and memory management.

Course Contents
Primitive types, Arrays, Records Strings and String processing. Data representation in memory, Stack and Heap allocation, Queues, Trees. Implementation strategies for stack, queues, trees. Run time storage management; Pointers and References, linked structures.

Lab work: Writing C+/C++ functions to perform practical exercises and implement using the algorithms on arrays, records, string processing, queues, trees, pointers and linked structures.

CSC 308 Operating System (3 Units C: LH 30; PH 45)

Learning Outcomes
At the end of this course, students should be able to:
1. recognise operating system types and structures;
2. describe OS support for processes and threads;
3. recognise CPU scheduling, synchronisation, and deadlock;
4. resolve OS issues related to synchronisation and failure for distributed systems;
5. explain OS support for virtual memory, disk scheduling, I/O, and file systems;
6. identify security and protection issues in computer systems; and
7. use C and Unix commands, examine behaviour and performance of Linux, and develop various system programmes under Linux to make use of OS concepts related to process synchronisation, shared memory, mailboxes, file systems, etc.

Course Contents
Fundamentals of operating systems design and implementation. History and evolution of operating systems. Types of operating systems. Operating system structures. Process management: processes, threads, CPU scheduling, process synchronisation. Memory management and virtual memory. File systems; I/O systems; Security and protection; Distributed systems; Case studies.

Lab work: Practical hands-on engagement to facilitate understanding of the material taught in the course. All the process, memory, file and directory management issues will be demonstrated under the LINUX operating system. Also UNIX commands will be briefly discussed. Alternatively, hands-on exposure may be through the use of operating systems developed for teaching, like TempOS, Nachos, Xinu or MiniOS. Another possibility is through programming exercises that implement and simulate algorithms taught. Simulation of CPU scheduling algorithms, producer-consumer problem, memory allocation algorithms, file organisation techniques, deadlock algorithms and disk scheduling algorithms.

BU-COS -325 Introduction to Machine Learning (2 Units; Core; LH=15; PH=45)

Learning Outcomes
On successful completion of this course, students should be able to:

1. Discuss three (3) underlying issues and challenges that Machine Learning faces, relating to data, model selection, and model complexity.
2. Discuss five (5) benefits and drawbacks of two (2) Machine Learning techniques.
3. Expound on fundamental mathematical relationships that exist between supervised and unsupervised learning paradigms.
4. Demonstrate the use of three (3) Machine Learning algorithms in a practical situation.
5. develop Python codes that implements three (3) Regression models: Ordinary Linear Regression, Ridge Regression, and Decision Tree regression models

Course Contents
Introduction to Machine Learning. Applications of Machine Learning. Supervised vs Unsupervised Learning. Python libraries suitable for Machine Learning. Linear Regression. Non-linear Regression. Model evaluation methods. Classification. K-Nearest Neighbour. Decision Trees. Logistic Regression. Support Vector Machines. Model Evaluation. Unsupervised Learning. K-Means Clustering. Hierarchical Clustering. Density-Based Clustering. Project - Recommender Systems. Content-based recommender systems. Collaborative Filtering.

Minimum Academic Standard
Software Laboratory

BU-COS 333 **Database Design, Management and Implementation** (3 Units; Core; LH=30; PH=45)

Learning Outcomes
On completion of the course, students should be able to:
1. Describe database design problem statement and how to write one for any given database application.
2. List five (5) methods of generating business rules and requirements for a database application
3. Enumerate how to use Microsoft access to share, import, export data and create database for use in a business environment.
4. Identify and describe how establish entities from mini real-world database problems.
5. Develop a Use design and mapping of entity-relationship diagram(erd) to relational schema
6. Identify functional dependencies and normalizations among relations

Course Contents
Introduction to database and database using Microsoft Access. Database components and environments. File processing system. Database System and Database Management Systems. Database Schema. Database Design and Architecture. Entity Relationship Diagram (ERD). Extended Entity Relationships Diagram (EERD). Relational and Conceptual Models. Mapping ERD and EERD relational Schema. Functional dependencies and Normalizations. First Normal Form. Second Normal Form. Third Normal Form. Structured Query Language (SQL). Creation and Use of

queries in practical applications. Practical real life database applications. Use of Microsoft Access to create Database. Creating forms. Queries and reports. Data sharing among Applications-Importing and exporting data. How to design database using technology tools and techniques such as Visio. Draw.io and others.

Minimum Academic Standard
Software laboratory.

BU-COS-307 Linux System Administration (3 Units; Core; LH=30; PH=45)

Learning Outcomes
On completion of this course, students should be able to:

1. Apply the basic concept and role of Linux as an operating system to daily activities.
2. List 5 fundamentals of system administration using Linux.
3. Explain three (3) roles of a system administrator and skills required to make a good system administrator.
4. Demonstrate a knowledge of Linux shells and use some of the shells.
5. Enumerate 6 basic commands in Linux and use these commands to produce specified action
6. List at least 10 files and directories management in Linux.

Course Contents
Fundamental of Linux System Administration. Roles of a system administrator. Linux Software Architecture. Linux User Interfaces. Installing Linux. Linux Text Editors. Managing Linux File System. File Security. Linux Users and Groups. Basic Password Management. Account Security. Managing Ownerships and Permissions. Installing and Managing Packages in Linux. Processes in Linux. Signal Processing. Networking and Internetworking. Basic commands in Linux. Change of ownership and group ownership of files and directories. Package Management and YUM Server. GUI–based applications for managing the network.

Minimum Academic Standard
Software Laboratory

BU-SEN-303 **Introduction to Big Data Engineering** (3 units; Core; LH=30; PH=45)

Learning Outcomes
On the completion of this course, students should be to:

1. Define big data with respect to the needs of businesses around
2. State three (3) characteristics of big data
3. List at least five (5) importance of big data to Nigeria and other developing countries
4. Enumerate five (5) benefits of big data to medicine
5. Review at least three (3) procedures involved in using Microsoft tableau in creating, editing and visualization.
6. Differentiate between the benefits of python and Microsoft tableau for data creation and visualization using only five concepts

7. Identify five (5) benefits of using Microsoft excel for data creating, editing and visualization.

Course Contents

Introduction to Big data. Characteristics of Big Data. Application of Big data to Business. Overview of Big Data tools. Microsoft Tableau in visualizing data. Installation of Microsoft Tableau. Microsoft Tableau in creating CSV data. Microsoft Tableau in data creation. Editing and visualization. Stages of installing Python programming. CSV files and storage stages in Python. Visualization of problems using Python programming. Introduction to Microsoft Excel. The basics of Microsoft Excel. Microsoft Excel operators. Calculations in Microsoft Excel using all variables. Operators and defined variables. Differenting defined and predefined variables. Microsoft Excel charts. Creating an auto chart board from Microsoft Excel. Multiple sheets in visualization of data in Microsoft Excel.

Minimum Academic Standard

Software Laboratory

BU-SEN-313 Mobile Application Development (2 Units Core; LH=30; PH=45)

Learning Outcomes

On completion of the course, students should be able to:

1. List at least two (2) basic concept of mobile application development.
2. Discuss the different types of mobile application operating systems, their components, and API
3. Explain the process of reading, writing, executing, and debugging react-native program
4. Develop with react-native styling and designing
5. Illustrate redux data architecture
6. Outline and calls react-native API reference

Course Contents

Introduction to mobile application development. Introduction to App design issues and considerations. Introduction to Android Operating System. Introduction to iPhone Operating System. Introduction to Mobile Application Business issues. Introduction to the react-native toolchain. introduction to the react-native ecosystem. Introduction to react-native styling and designing. Introduction to react-native navigation. Introduction to animations in react-native. Introduction to redux data architecture. Introduction to managing hardware platforms in react-native. Introduction to API referencing in react-native. Introduction to IOS-specific components and API. Introduction to Android-specific components and API. Introduction to publishing Apps in react-native. Introduction to building Star Wars app using cross-platform components.

Minimum Academic Standards

Software Laboratory

IFT 342: Network Servers and Infrastructures (2 Units C: LH 15;PH 45)
Learning Outcomes

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

1. analyse IPv6 networking concepts and practices for communications over VPNs;
2. explain the fundamental concept of Virtual Computing, Cloud Computing, VoIP;
3. demonstrate through practical examples how protocols are used to enable communication between computing devices connected;
4. list the opportunities of virtual computing service provision models, such as cloud computing for organisations; and
5. identify, connect and install applications on virtual servers.

Course Contents
IP networking concepts and practices for IPv6 addressing. DHCP and DNS in IPv.6 networks. Secure communication over VPNs, VoIP architecture. Concept of Virtual Computing, Cloud Computing, VoIP. Traffic monitoring and network connectivity between operating systems. Overview of latest technologies of IP networks and understand application-level services used in the internet. Multi-Protocol Label Switching (MPLS). VPN Secure Network Connectivity. VoIP Architecture. Network Neutrality.

Lab Work: Demonstration of IPv6 networks including DHCP and DNS configuration.Basics of VPNs. Simple applications of VPNs. Installation of applications on virtual servers. Monitoring traffic on virtual servers. Working with Multiple Servers. Balancing traffic on servers. Testing the security of VPNs. Illustration of VOIP architecture.

CSC 309: Artificial Intelligence (2 Units C: LH 15; PH 45)
Learning Outcomes
At the end of this course, students should be able to:
1. explain AI fundamentals, concepts, goals, types, techniques, branches, applications, AI technology and tools;
2. discuss intelligent agents, their performance, examples, faculties, environment and architectures, and determine the characteristics of a given problem that an intelligent system must solve;
3. describe the Turing test and the "Chinese Room" thought experiment, and differentiate between the concepts of optimal reasoning/behaviour and human-like reasoning/behaviour;
4. describe the role of heuristics and the trade-offs among completeness, optimality, time complexity, and space complexity;
5. analyse the types of search and their applications in AI and describe the problem of combinatorial explosion of search space and its consequences;
6. demonstrate knowledge representation, semantic network and frames along with their applicable uses;
7. practice Natural Language Processing, translate a natural language (e.g., English) sentence into a predicate logic statement, convert a logic statement into clause form, apply resolution to a set of logic statements to answer a query; and
8. analyse programming languages for AI and expert systems technology, and employ application domains of AI.

Course Contents

Overview of Artificial Intelligence. History of AI. Goals of AI. AI Technique. Types of AI. Branches and applications of AI. Advantages and Disadvantages. Introduction to Intelligent Agents. Agent Performance, Examples of Agents, Agent Faculties, Rationality, Agent Environment. Agent Architectures. Search. General Classes of AI Search Algorithm Problems. Problem Solving by Search. Types of AI Search Techniques and Strategies. Introduction to the types of problems and techniques in AI. Problem-Solving methods. Major structures used in AI programmes. Knowledge Representation. KR and Reasoning Challenges. KR Languages. Knowledge representation techniques such as predicate logic, non-monotonic logic, and probabilistic reasoning. Semantic Network - types of relationships, semantic network inheritance, types and components. Introduction to Frames. Natural Language Processing (NLP). Introduction to natural language understanding and various syntactic and semantic structures. Introduction to Expert Systems - characteristics, components, types, requirements, technology, development. Programming Languages for AI. Introduction to computer image recognition.

Lab work: Group practical in (i) Turing test practical - Students can act out their own version of the Turing test (iii) Facial recognition practical to aid in teaching students how machine learning works with students simulating a facial recognition algorithm. Practical applications of NLP in groups – (i) Question Answering focuses on building systems that automatically answer the questions asked by humans in a natural language (ii) Spam detection application for detecting unwanted e-mails getting to a user's inbox (iii) Sentiment analysis/opinion mining should be used on the web to analyse the attitude, behaviour, and emotional state of the sender, implemented through a combination of NLP and statistics (iv) Practical exercise of machine translation used to translate text or speech from one natural language to another natural language such as the Google Translator (v) Developing a model to provide word processor software for the spelling correction (vi) Developing a model for speech recognition for converting spoken words into text (vii) Implementing a Chatbot to provide the staff/student's chat services. OR
Group Practical exercise on agents and its environment using simulation of a colony of ants foraging for food; model simulating a message between agents; model simulating the flocking behaviour of birds; model to apply standard search algorithm to the classic search problem of missionaries and cannibals, and how to use communicating agents for searching networks. Some computer AI animation exercises for any branch of AI. Practical exercise on simple robots coupling and programming. Group project of building a lawn robot for trimming grasses, or any simple design and implementation of robotics.

400 Level

COS 409: Research Methodology and Technical Report Writing (3 Units C: LH 45)

Learning Outcomes

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

1. describe research, types, approaches, significance of research, research methods, research process, criteria and strategy for good research;
2. discuss the principles of scientific research, scientific investigation, problem formulation, and technique of the research problem;
3. describe the various elicitation methods;
4. develop appropriate data collection instruments;
5. conduct the literature review process; and
6. prepare briefs as well as technical reports and know how to cite referenced works and prepare references and bibliography.

Course Contents

Foundations of Research. Types of Research. Research Approaches. Significance of Research. Research Methods versus Methodology. Research Process. Criteria and Strategy for Good Research. Principles of Scientific Research. Scientific investigation. Problem Formulation and Its Techniques. Developing Research Proposal and Research Plan. Formulation of Research Questions and Hypothesis Testing. Developing Research Proposal and Research Plan. Literature Review. Procedure for Reviewing Related Relevant Studies. Methods for Collection of Primary and Secondary Data. Elicitation Techniques - Questionnaires, Interviewing, Ethnography, etc. Guidelines for Constructing Data Instruments. Methods of AnalysingData in Computing and Related Disciplines.System Design: Architectural design, input design, process design, output design. Use case analysis, sequence diagram, activity diagram, deployment diagram, etc.Types of Reports. Technical Report Writing. Layout and Mechanics of Writing a Research Report. Standard Techniques for Research Documentation. Interpretation and Presentation of Results. How to Cite Referenced Works and Prepare References and Bibliography.

SEN401: Software Configuration Management & Maintenance (2 Units C: LH 15; PH 45)

Learning Outcomes

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

1. state the importance of software configuration management;
2. explain the typical processes in software configuration management; and
3. describe the key issues in software maintenance

Course Contents

Management of the software configuration management process – organisation context for software configuration management, constraints and guidance for software configuration management process. Planning for software configuration management, software configuration management plan, and surveillance of software configuration management. Software configuration identification and software library. Software configuration control – requesting, evaluating and approving software changes, implementing software changes, and deviations and waivers. Software configuration status accounting – software configuration status information and reporting. Software configuration auditing. Key issues in software maintenance – technical issues, management issues, maintenance cost estimation, and software maintenance measurement. Maintenance process – maintenance processes and activities. Techniques for maintenance – program comprehension, re-engineering, reverse engineering, migration, and retirement.

Lab Work: Practical demonstration of software configuration management processes. Working with software configuration management software. Illustration of software maintenance processes and activities. Working with software maintenance software. Illustration of software re-engineering and reverse engineering techniques.

SEN 410: Software Architecture and Design (2 Units C: LH 15; PH 45)
Learning Outcomes
At the end of this course, students should be able to:

1. describe design patterns, frameworks and architectures;
2. explain design of distributed systems and component-based design; and
3. describe the techniques of designing for qualities such as reliability, performance, safety, security and reusability.

Course Contents
An in-depth look at software design. Continuation of the study of design patterns, frameworks, and architectures. Survey of current middleware architectures. Design of distributed systems using middleware. Component based design. Measurement theory and appropriate use of metrics in design. Designing for quality attributes such as reliability, performance, safety, security, reusability, etc. Measuring internal qualities and complexity of software. Evaluation and evolution of designs.

Lab Work: Practical demonstration of the use of design patterns, frameworks and architectures. Practical simulation of distributed systems. Illustration of component-based design. Working with software design software. Use of software metrics measuring software.

SEN 490: Final Year Project (6 Units C: PH 270)

Learning Outcomes
Upon completion of the project, students should be able to:

1. demonstrate technical skills in Software Engineering;
2. demonstrate generic transferable skills such as communication and team work;
3. produce a technical report in the chosen project;
4. defend the written project report; and
5. appreciate the art of carrying out a full-fledged research.

Course Contents
This is a continuation of SEN 497. This contains the implementation and the evaluation of the project. A formal written report (chapters 4-5) has to be approved by the supervisor. A final report comprising chapters 1-5 will be submitted to the department for final grading. An oral presentation is required.

INS 401 Project Management (2 Units C: LH 30)

Learning Outcomes

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

1. describe project management planning;
2. describe project scheduling;
3. explain management of project resources;
4. discuss project procurement, monitoring and execution; and
5. explain project communication and time management;

Course Contents

Introduction to Project Management. The Project Management Lifecycle. Project management and systems development or acquisition. The project management context, technology and techniques to support the project management lifecycle, and Project management processes. Managing Project Teams: Project team planning, Motivating team members, Leadership, power and conflict in project teams, and Managing global project teams. Managing Project Communication and enhancing team communication. Managing Project Scope: Project initiation, how organisations choose projects, activities, and developing the project charter. Managing Project Scheduling: Common problems in project scheduling, and Techniques for project scheduling. Managing Project Resources: Types of resources (human, capital, time), and techniques for managing resources. Project quality and tools to manage project quality. Managing project risk and tools for managing project risk. Managing Project Procurement: Alternatives to systems development, External acquisition, Outsourcing-domestic and offshore, Steps in the procurement process, and Managing the procurement process. Project Execution, Control and Closure: Managing project execution, monitoring progress and managing change, Documentation and communication, and Common problems in project execution; Managing Project Control and Closure: Obtaining information, Cost control, Change control, Administrative closure, Personnel closure, Contractual closure and Project auditing.

BU-COS-419 **Agile Development and Scrum** (2 Units; Core; LH=15; PH=45)

Learning Outcomes

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

1. Differentiate between Agile Scrum and traditional project management methodologies
2. Discuss three (3) foundational principles of the scum methodology
3. Create a diagram that highlights all meetings, roles, and artifacts while outlining the Scrum framework.
4. Explain three (3) responsibilities of each role in scrum – the team members, product owner, scrum-master and stakeholder
5. Explain the purpose of team meetings, and how they support the scrum framework
6. Demonstrate five (5) foundational principles of scrum to monitor and report project progress and status

Course Contents

Project Management Basics. Traditional Versus Agile Project Management. Agile Methodology. Scrum Overview. Scrum Framework and Theory. Three Pillars of Scrum. Scrum Roles. Organizational Influences on Project Management. Managing

the Release Planning. Creating effective Users Stories. Product Backlog and Grooming. Working the Sprint Backlog. Running the Sprint/Iteration. Sprint/Iteration Review. Sprint/Iteration Retrospective. Creating and Collecting Artifacts. Agile Methodologies.
Minimum Academic Standard
Software Laboratory

BU-SEN-406 Formal Methods in Software Engineering (2 Units Core; LH=30; PH=Nil)

Learning Outcomes
On completion of the course, students should be able to:
1. Discuss at least three (3) concepts of formal methods and their relevancy to software engineering.
2. Define propositional logic
3. Illustrate predicate logic in relation to formal methods.
4. Enumerate at least five (5) of the importances of set theory.
5. Explain mathematical proof of a concept.
6. List at least five (5) applications of formal specification

Course Contents
Concepts of Formal methods. Relevance of software engineering. Propositional logic. Proposition. Predicate logic. Equality. Definite description. Set theory. Definitions. Relations. Functions. Sequence and free types. Mathematical proof of a concept. Application of formal specification. Z notation. Schema Operators. Promotion. Preconditions.

Minimum Academic Standard
Software laboratory

BU-SEN-407 Software Measurement and Metrics (2 Units; Core; LH=30; PH=Nil)

Learning Outcomes
On completion of the course, students should be able to:
1. List 5 fundamentals of measurement and experimentation.
2. Identify the importance of measurement in software engineering.
3. Design at least 2 measurement models, with appropriate scale and scale types
4. Illustrate how to measure different internal and external software attributes.
5. Illustrate how measurement supports investigating the use and effectiveness of software engineering tools and techniques.
6. State at least 2 of the software-related empirical investigation tasks.

Course Contents
Introduction to Metrics and Measurement. Representational Theory of Measurement. Measurement and Models. Measurement Scales and Types. Classification of Software Measures. Software Measurement Validation. Principles of Empirical Studies in Software Engineering. Software Engineering Experiments Planning. Software Metrics Data Collection. Software Measurement Data Analysis. Decision Support Metrics. Internal and External Product Attributes. Software Size Measurement. Software Structure Measurement. Introduction to Object-Oriented Structural Attributes and

Measures. Software Quality Measurement. Software Usability Measurement. Software Maintainability Measurement. Software Security Measurement. Software Reliability Measurement and Prediction.

BU-SEN-411 Open Source Systems Development (2 Units; Core; LH=30, PH=Nil)

Learning Outcomes

On completion of this course, students should be able to:

1. Identify three (3) major development platforms and tools that are common for open source projects
2. Describe the correct license, development model, and development community for open source projects, and can initiate a new project or join an existing project.
3. List at least five (5) ethical values and proper behaviour while engaging in OSS projects, to develop the best code and build on the strengths of other contributors.
4. Enumerate at least three (3) practices for long-term sustainability of projects, including how to respect and ncourage diversity.
5. Discuss Github and one (1) other hosting providers and advanced interfaces, such as Gerrit.
6. State five (5) open source projects related to a given development problem

Course Contents

Types of Software. Open source software (OSS). Bundled and Unbundled software. Issues of Open Source technologies. Characteristics of Open Source technologies. Team-based OSS development process. Agile FOSS development process. Licensing and Development growth of OSS. Open Source communities and development processes. Structure of Open Source industry. Business models for FOSS. Implications of OSS technologies. Identifying OSS Projects. Client-based Projects. Community-based Projects. Mobile OSS Development. Web based OSS Development. OSS and Open Systems.

BU-SEN-417 Human Computer Interaction and Emerging Technologies (3 units; Core; LH=30; PH=45)

Learning Outcomes

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

1. List 5 user -centred design methods for conducting formative and summative evaluations.
2. Enumerate at least three (3) models from the field of HCI.
3. Explain the process of interaction by using table and engaging graphical computer interfaces.
4. Evaluate a given research paper on HCI.
5. List at least five (5) fundamental ideas behind Cloud Computing.

Course Contents

Human-computer interaction concepts. HCI Theories and practice. Human computer interaction practices. Basic components of human computer interaction. HCI as an interdisciplinary field. Evaluation of computer-based technologies. Design principles in HCI. Usability and user experience factors. Interaction design. Theory and practice in interface specification. Design and evaluation of interface design. Implementation and evaluation of user interface. Ethnographic study and requirements. Scenario-based design. Current research in HCI. Presentation and critiquing of HCI research. Emerging technologies in HCI. Evolving technologies and organizations. Challenges and opportunities in designing projects that implement new and emerging technologies.

BU-SEN 418 **Professional Ethics and Practices** (2 Units; Core; LH=30; PH=Nil)

Learning Outcomes
On completion of the course, students should be able to:

1. Explain the social, ethical issues and dilemmas in software engineering;
2. Analyze at least three (3) ethical dilemmas and make informed ethical decisions;
3. Enumerate at least five (5) moral problems in terms of facts, values, stakeholders and their interests;
4. List at least 5 professional codes of conduct and ethical standards for software engineers;
5. Apply three (3) options for action in the light of (conflicting) moral values and the relevant facts;
6. Describe at least three (3) ethical theories and frameworks and to make a decision based on the assessment;

Course Contents
Introduction to Ethics and Morality. Introduction to professional ethics and software engineering. Ethical principles and values in software engineering. The IEEE Code of Ethics. ACM Code of Ethics and Professional Conduct. Ethical decision-making frameworks in Software Engineering. Social responsibility and the impact of software engineering. Cyber Ethics. Intellectual property rights and copyright laws. Privacy and security in software engineering. Project management methodologies. Agile software development and ethical considerations. Ethical considerations in software design and architecture. Professional communication for ethical decision-making. Risk Management. Computer Security. Information Security Policy. Case studies of ethical issues in software engineering. Emerging issues and challenges in software engineering ethics.

CSC 401: Algorithms and Complexity Analysis (2 Units C: LH 30)

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

1. explain the use of big-O, omega, and theta notation to describe the amount of work done by an algorithm,
2. use big-O, omega, and theta notation to give asymptotic upper, lower, and tight bounds on time and space complexity of algorithms,
3. determine the time and space complexity of simple algorithms,
4. deduce recurrence relations that describe the time complexity of recursively defined algorithms,
5. solve elementary recurrence relations,
6. for each of the strategies (brute-force, greedy, divide-and-conquer, recursive backtracking, and dynamic programming), identify a practical example to which it would apply,

7. use pattern matching to analyse substrings, and
8. use numerical approximation to solve mathematical problems, such as finding the roots of a polynomial.

Course Contents
Basic algorithmic analysis. Asymptotic analysis of Upper and average complexity bounds. Standard Complexity Classes. Time and space trade-offs in analysis recursive algorithms. Algorithmic Strategies. Fundamental computing algorithms. Numerical algorithms. Sequential and Binary search algorithms. Sorting algorithms, Binary Search trees. Hash tables. Graphs and their representation.

CYB 402: Steganography: Access Methods and Data Hiding (2 Units C: LH 15; PH 45)

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

1. discuss secret writing and different methods and tools used for each;
2. identify why steganography is important, and how it is different from cryptography and encryption;
3. describe the uses and applications of steganography, and how to use steganography methods and work with any of the steganography types and techniques;
4. practice the different steganography techniques for encrypting the data and use data hiding methods, techniques and access methods;
5. develop the information-hiding systems, steganography algorithm and security of a steganographic algorithm;
6. analyse how to detect steganography, finding images, and verifying hidden content; and
7. organise practical experimentation of data hiding tools, investigation techniques and the latest countermeasures.

Course Contents
History of secret writing. An overview of steganography. Introduction to steganography - Definition of steganography. Why is steganography important? Steganography vs. Encryption. Uses of steganography. Problem of steganography. Steganography applications and methods. Steganography types and methods - text steganography, images steganography, video and audio steganography. Steganography techniques. Survey of different steganography techniques for encrypting the data. Information hiding: steganography and steganalysis. Data hiding methods, techniques and access methods. Requirements for data hiding. Steganography and Business - the basics of embedding, different aspects in information-hiding systems. Steganographic algorithm. Security of a steganographic algorithm. Steganography detection, finding images, and verifying hidden content. Research and practical experimentation of data hiding tools. Research on investigation techniques and the latest countermeasures.

Lab work: Practice secret writing using different methods and tools. Learn how to use steganography methods and techniques for encrypting the data. Master data hiding methods, techniques and access methods using case study exercises. Write samples steganography algorithm and secure the algorithm. Detect elements of steganography, finding images, and verifying hidden content in a given text, image, audio and video samples

IFT 410: System Integration and Architecture (2 Units C: LH 30)

Learning Outcomes

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

1. discuss systems integration activities as a part of the development lifecycle;
2. explain and apply key systems integration architecture, methodologies, and technologies;
3. apply integration technologies to implement system integration solutions; and
4. describe Interplay between IT applications roll-out and related organisational processes.

Course Contents

System architecture, testing, evaluation, and benchmarking. Contracts, RFPs, and quality. System integration and deployment. System release. Pilot and acceptance testing and defect repair. System support strategies and user support plans, and enterprise integration approaches, standards, and best practices. Testing and quality assurance. Role of systems architecture in systems integration, performance, and effectiveness. Principles and concepts of DevOps. The interplay between IT applications roll-out and related organisational processes. The concept of Enterprise Architecture. Developing an Enterprise Architecture.