SCHOOL OF SCIENCE AND TECHNOLOGY
PAN-ATLANTIC UNIVERSITY

Programme Manual

B.Sc. Software Engineering

SST • School of Science and Technology • Pan-Atlantic University
52 km Lekki-Epe Expressway, Ibeju-Lekki, Lagos, Nigeria

1.0 THE MISSION OF THE BSC SOFTWARE ENGINEERING PROGRAMME

The undergraduate programme in software engineering is aimed at developing competent, creative, innovative, entrepreneurial and ethically-minded persons, capable of creating value in the diverse fields of Software engineering.

2.0 THE PHILOSOPHY OF PROGRAMME

The overall intention is to nurture individuals who are technically skilled, creative, innovative, professionally competent, enterprising, and zealous for the common good, with the ability to make free and morally upright decisions, and who can thus impart positive values in service to society. The programme will provide intensive practical and theoretical courses, which are designed to prepare the students to work in modern day industries as Software Engineers. Career prospects for graduates of this course include computer software design and development as applied to various fields. The programme will emphasize particularly the need for high ethical standards in the exercise of professional work, training, teaching, and obligations. Hence, the curriculum will be suffused with courses that deal with human values, analytical and critical thinking and the appropriate design and use of computing systems.

The programme is largely driven by the need to facilitate, through adequate theoretical and practical training, the emergence of competent professionals in the area of software engineering. The programme aims to build new generation of software engineers that can favourably compete with peers from any reputable institutions in the world. As a fundamental principle, the programme emphasizes interactions between the industry players, lecturers and students, with the goal of ensuring relevance to the industry as well as driving the innovation needs of the industry. Together with the technical skills and competencies, the programme also places emphasis on a holistic development of the positive character traits of the students. Such traits could be critical success factors in the team work required for professional success in the workplace.

3.0 GUIDING PRINCIPLES FOR THE PROGRAMME

The following basic points are the guiding principles for the programme:

a) The programme will impart an education that is relevant to the needs of the nation and of international standard. The relevance of the programme's content will be ensured by fostering a strong relationship with the industry.
b) The programme will give particular emphasis to teaching and research. The academic staff will be encouraged to engage in research and attend conferences of relevance across the world. This is expected to ensure a continuous improvement in their teaching and maintain its relevance to the needs of the nation.
c) The programme will be concerned with the integral formation of the individual and will lay special emphasis on the development of values and ideals. Professional ethics will permeate all teaching activities in the programme.

4.0 GOALS OF THE PROGRAMME

The aims and objectives of the programme include but are not limited to the following:

a) Create in students the awareness of and enthusiasm for software engineering and its capabilities.

b) Involve the students in an intellectually stimulating and satisfying experience of learning and studying
c) Provide a broad and balanced foundation in software engineering knowledge and practical skills.
d) Develop in students through an education in software engineering a range of transferable applicable skills of information technology to all aspects of human endeavours.
e) Generate in students an appreciation of the importance of computer in an industrial, economic, technological and social context.
f) Provide students with knowledge and skills base for further studies in software engineering or multi-disciplinary studies involving software engineering.
g) To offer an integral formation with emphasis on the development of values and ideals that will help prepare the students to play leadership roles in such industries.

5.0 ADMISSION AND GRADUATION REQUIREMENTS

5.1. Admission and Matriculation Requirements

The entry requirements shall be at least credit level passes in five subjects including English Language, Mathematics, and Physics to form the core subjects with credit in any other two relevant science subjects, at the Senior Secondary School Certificate or its equivalent in not more than two sittings. In addition, an acceptable pass (currently 220) in the Unified Tertiary Matriculation Examination (UTME), with relevant subject combination is required for Admission into 100 Level.

Candidates with two A level passes (graded A-E) at the GCE Advanced Level in relevant subjects (Chemistry, Mathematics and Physics) may be admitted into 200-level. This is in addition to fulfilling the requirement of a minimum of credit level passes in five relevant subjects at SSCE or WASCE/GCE 'O' Level as indicated above. Other tertiary level qualifications such as OND, HND may be considered for direct entry as well.

5.2. The Semester Course System

The undergraduate programme in Software Engineering will be run on the semester course basis, and there will be two semesters in the academic year. Instruction in the programme shall be by courses, and it will be mandatory for students to take an approved combination of courses in any semester.

An evaluation of the courses will be carried out in terms of course units. For this purpose, one course unit is defined as one lecture/seminar/tutorial hour or three hours of practical class per week, for the duration of a semester. Ordinarily, students shall be expected to register for a prescribed number of units in each academic year. This number will be determined by Senate from time to time, based on the recommendation of the School Board.

There shall be four levels of courses in line with the years of study. The levels shall be numbered respectively as 101-199, 201-299, 301-399 and 401-499. Each of these numbers shall be prefixed by a two or three letter subject code.

Students will be required to complete their registration for the courses within the period stipulated by the School. Amendment of this registration will be allowed through the addition or deletion of courses but it must take place within three weeks of the commencement of lectures.

Direct entry and transfer students that enter into the second year of the programme will have to take some compulsory courses from the first year prior to their graduation from the University. However, if the Faculty Board assesses that a student has done any of the courses elsewhere, such a student will be exempted from taking the course.

5.3. Examinations and Grading System

At the end of each semester, students will be examined on all the courses they have registered for and been taught during that period. They shall subsequently be credited with the number of course units assigned to the courses that they pass.

The assessment of students will be based on a combination of continuous assessment (tests, assignments, etc.), class participation and examinations. To be eligible to sit for any examinations, students will be expected to attend a minimum of 80% of the lectures of any course registered for. All courses registered for will be taken into consideration during the computation of results. Students will not be credited for courses that they did not register for even if they are inadvertently allowed to take the examinations and pass them.

Failure to take the examination in a course for which one has registered will attract a score of 0.0, which will have the consequent effect of lowering the student's Grade Point Average.

Special examinations to enable a student graduate may in exceptional circumstances be held by order of Senate. Grades will be awarded based on the scores of the students as follows:

Percent Score	Grade point	Letter Grade
70 -100	5.0	A
60 – 69	4.0	B
50 – 59	3.0	C
45 - 49	2.0	D
0 – 44	0.0	F

For the purpose of description, a score below 2.0 Grade Point (from letter grade D) constitutes a failure. The following qualifications shall be applied to the grades:

A	Excellent
B	Good
C	Fair
D	Pass
F	Failed

To obtain the Cumulative Grade Point Average (CGPA) of the student, the grade point assigned to the mark obtained in each course is multiplied by the units of that course. The total from all the courses is added up to give the total weighted grade point. This total is then divided by the total number of units taken by the student to give the grade point average.

5.4. Retention and Progression

To remain in the School, students will be required to ensure that their CGPA does not fall below 1.5. If a student's CGPA falls below 1.5, the student will be placed on probation. If the student fails to improve and, after one semester of probation, his/her CGPA remains below 1.5, that student will be asked to withdraw. A student on probation will not be permitted to register for more than 18 units.

5.5. Period of Study and Requirements for the Award of a Degree

The normal period of study for an honours degree shall be eight semesters for 100 level entrants and six semesters for 200 level direct entry students. In order to be eligible for graduation, the student must pass all the compulsory and required courses, the total number of which is 130 credit units for UTME admissions. For direct entry entrants, the equivalent is 97 credit units.

Direct entry students will not be required to take 100 level courses unless a given course is a prerequisite for a higher-level course and the student has not done that course or equivalent, as determined by the Faculty Board. The University has also specified that direct entry students must take GST 102 (Introduction to Theology), a 2-credit unit course. Therefore, minimum credit units required for direct entry students to graduate is 99. Transfer students from another school or department into 200 level follow the same criteria as direct entry students.

The determination of the class of degree shall be based on the weighted grade points of all the courses taken, including the courses that are repeated. The award of the degree with honours shall be dependent on the student having obtained a Cumulative Grade Point Average of at least 2.0 in addition to fulfilling other minimum requirements for an honours degree. The following classes of degree are approved for the CGPA indicated:

Class of Degree	Cumulative GPA
First Class	4.5 – 5.0
Second Class (Upper Division)	3.5 – 4.49
Second Class (Lower Division)	2.4 – 3.49
Third Class	1.5 – 2.39

The maximum number of semesters for the award of an honours degree shall be ten semesters. A student who spends more time than this to complete the degree programme will ordinarily not be eligible for an honours classification.

5.6. Curriculum for B.Sc. Degree in Software Engineering in agreement with the NUC CCMAS Standards

NOTE the following legend for the list of courses below:

C = Compulsory Course – A course which every student must compulsorily take and pass in any particular programme at a particular level of study.

E = Elective Course – A course that students take within or outside the faculty (school). Students may graduate without passing the course provided the minimum credit unit for the course had been attained.

LH = Lecture Hours per semester
PH = Practical Hours per semester

B.Sc. SOFTWARE ENGINEERING PROGRAMME STRUCTURE

100 Level

Course Code	Course Title	Unit(s)	Status	Pre-Req	LH	PH
GST 111	Communication in English	2	C		15	45
GST 112	Nigerian Peoples & Culture	2	C		30	0
MTH 101	Elementary Mathematics I	2	C		30	0
MTH 102	Elementary Mathematics II	2	C		30	0
PHY 101	General Physics I	2	C		30	0
PHY 102	General Physics II	2	C		30	0
PHY 107	General Practical Physics I	1	C		0	45
PHY 108	General Practical Physics II	1	C		0	45
STA 111	Descriptive Statistics	3	C		45	0
COS 101	Introduction to Computing Sciences	3	C		30	45
COS 102	Introduction to Problem Solving	3	C		30	45
PAU-SEN 111	Introduction to Software Packages	3	C		30	45
PAU-SEN 112	UI/UX Design Essentials	3	C		30	45
PAU-SEN 114	Linear Algebra for Computer Science	2	C		30	
PAU-SEN 192	Introduction to Christian Theology	2	C		30	
TOTAL		**33**				

200 Level

Course Code	Course Title	Unit(s)	Status	Pre-Req	LH	PH
GST 212	Philosophy, Logic and Human Existence	2	C		30	0
ENT 211	Entrepreneurship and Innovation	2	C		30	0
MTH 201	Mathematical Methods I	2	C		30	0
MTH 202	Mathematical Methods II	2	C		30	0
COS 201	Computer Programming I	3	C		30	45
COS 202	Computer Programming II	3	C		30	45
SEN 201	Introduction to Software Engineering	2	C		30	0
SEN 299	SIWES I	3	C		0	135
CSC 203	Discrete Structures	2	C		30	0
INS 204	System Analysis and Design	3	C		30	45
IFT 211	Digital Logic Design	2	C		15	45
IFT 212	Computer Architecture and Organisation	2	C		15	45

PAU-SEN 211	Computer Hardware and Networking Essentials	3	C		30	45
PAU-SEN 212	Computer Graphics	3	C		30	45
PAU-SEN 214	Computer Security Fundamentals	2	C		30	
PAU-SEN 292	The Nature of Human Beings	2	C		30	
PAU-SEN 293	English for Business Purposes	2	C		30	
TOTAL		**40**				

300 Level

Course Code	Course Title	Unit(s)	Status	Pre-Req	LH	PH
GST 312	Peace and Conflict Resolution	2	C		30	0
ENT 312	Venture Creation	2	C		15	45
SEN 301	Object-Oriented Analysis and Design	2	C		15	45
SEN 304	Software Testing and Quality Assurance	2	C		15	45
SEN 306	Software Construction	2	C		15	45
SEN 322	Software Engineering Innovation and New Technology	2	C		15	45
SEN 399	SIWES II	3	C		0	135
CSC 301	Data Structures	3	C		15	45
CSC 308	Operating Systems	3	C		30	45
PAU-SEN 311	Artificial Intelligence	2	C		15	45
PAU-SEN 312	Compiler Construction	3	C		30	45
PAU-SEN 313	Formal Methods	2	C		30	
PAU-SEN 314	Artificial Intelligence in Software Engineering Processes	2	C		15	45
PAU-SEN 316	Data Management I	3	C		30	45
PAU-SEN 392	Professional and Personal Skills	2	C		30	
TOTAL		**35**				

400 Level

Course Code	Course Title	Unit(s)	Status	Pre-Req	LH	PH
COS 409	Research Methodology and Technical Report Writing	3	C		45	0
SEN 401	Software Configuration Management and Maintenance	2	C		15	45

SEN 410	Software Architecture and Design	2	C		15	45
SEN 497	Final Year Student's Project I	3	C		0	135
SEN 498	Final Year Student's Project II	3	C		0	135
INS 401	Project Management	2	C		30	0
PAU-SEN 408	Ethics and Legal Issues in Computer Science	2	C		30	0
PAU-SEN 411	Machine Learning	2	C		15	45
PAU-SEN 412	Survey of Programming Languages	3	C		30	45
PAU-SEN 413	Human-Computer Interaction	2	E		30	
PAU-SEN 414	Deep Learning	2	E		15	45
PAU-SEN 415	Game Design and Development	2	E		15	45
PAU-SEN 416	Computer Vision and Image Processing	3	E		30	45
PAU-SEN 417	Introduction to Data Science and Engineering	3	E		30	45
PAU-SEN 418	IoT and Edge Computing	2	E		15	45
PAU-SEN 419	Advances in Web, Mobile and Blockchain Development	3	E		30	45
PAU-SEN 420	Data Management II	2	E		15	45
	TOTAL UNITS	**41**				

COURSE DESCRIPTION

100 LEVEL COURSES

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 selfreliance (indigenous trade and market system; indigenous apprenticeship system among Nigeria people; trade, skill acquisition and self-reliance). Social justices 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 3Rs – Reconstruction, Rehabilitation and Re-orientation; Reorientation 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. understand the basic definition of Set, Subset, Union, Intersection, Complements and use of Venn diagrams;
2. solve quadratic equations;
3. solve trigonometric functions;
4. understand 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. understand types of rules in Differentiation and Integration;
2. understand 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

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;
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.

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

Core concepts of computing. Identification of problems. Types of problems (routine problems and non-routine problems). Problem-solving. Methods of solving computing problems. 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 the 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. Programming in any language.

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)

PAU-SEN 111 **Introduction to Software Packages** (3 Units Compulsory; LH=30; PH=45)

Senate-approved relevance

An overarching motivation for this course is to get the software engineering students to aspire to become producers of very matured and useful software systems. By exposing them to a myriad of useful, industry standard software applications, they should be better positioned to set their standards very high when designing and writing software. Besides, a good working knowledge of industry standard software can make them more useful in the workplace.

Overview

In this course, students are introduced to various application software packages. The packages will include general purpose application software as well as some specialised software systems. The general-purpose software will include office productivity tools like word processing, spreadsheets, presentation and basic database management systems. Students will also be introduced to more specialised software like graphic packages, specialised document preparation systems like LaTeX, Markdown editors like GhostWriter, etc. Furthermore, students will be introduced to various forms of software packaging and delivery mechanisms as required by target platforms like Desktop, Mobile, Web, etc.

Objectives:

The objectives of the course are:

1. explain the concept of software packages and their benefits in terms of productivity, efficiency, and convenience;
2. describe popular software packages in different categories and help them identify the right software package for their needs;
3. explain how to use software packages effectively, including installation, setup, interface navigation, and basic operations;
4. describe advanced features and techniques of software packages;
5. describe customization, personalization, and efficient use;
6. explain how to troubleshoot common issues that may arise while using software packages and maintain them effectively, including backups and updates;
7. explain emerging trends and developments in software packages and their impact on the industry;
8. explain how to use specialized tools like LaTeX, markdown editors, etc.;
9. explain how software is packaged and delivered for various platforms.

Learning outcomes

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

1. distinguish between various general-purpose application software;
2. apply word processing operations including some common keyboard shortcut keys;
3. describe the basic features and functions of software packages;
4. use the software packages effectively and efficiently to complete specific tasks, such as creating and formatting documents, spreadsheets, presentations, or graphics;
5. navigate user interfaces, and solve technical issues;
6. use software to collaborate on documents or projects;
7. use specialized software tools;
8. explain how software is packaged and delivered for various platforms.

Course content

Introduction to software packages. Overview of commonly used software packages, such as Microsoft Office Suite. Basic computer skills. File management. Operating systems and hardware requirements. Word processing. Create and edit documents, format text, use tables, and work with images and graphics using LaTeX. Spreadsheets. Create, format, and edit spreadsheets, perform calculations, use formulas and functions, and create charts and graphs. Presentations. Create effective presentations, use slide layouts, add multimedia elements, and present information in a clear and engaging manner using Prezi. Collaboration and sharing. Share files, and manage versions using cloud-based storage and collaboration tools.

Minimum academic standards

Computer Laboratory.

Computer (1:2 students).

Office productivity tools e.g., MS Office 365.

LaTeX software for document presentation.

Prezi presentation app.

All other software to be learnt.

PAU-SEN 112 **UI/UX Design Essentials** (3 Units Compulsory; LH=15; PH=45)

Senate-approved relevance

Teaching UI/UX design fundamentals to software engineering students can help them improve their user-centred design skills, foster interdisciplinary collaboration, encourage innovation, and help the industry meet the demand for skilled UI/UX designers. Graduates will be better equipped to contribute to the digital product development industry and stand out in a competitive job market, which will benefit both students and the university.

Overview

This UI/UX design essentials course is intended to give software engineering students a fundamental understanding of user-centred design principles and practices. User research, wireframing and prototyping, visual design, designing for different devices and platforms, interaction design, information architecture, designing for emotional response, accessibility, and ethics and responsibility in design are all covered in the course. Throughout the course, students will carry out practical classwork that integrate the principles and practices learned in class. Students will have a solid understanding of UI/UX design principles and how to apply them to create effective, user-centred digital products, by the end of the course.

Objectives

The objectives of this course are to:

1. explain the fundamentals of user-centred design practices and principles;
2. explain how to conduct user research, create personas, and design for various platforms and devices;
3. describe the creation of efficient wireframes and prototypes as well as the creation of efficient visual interfaces;
4. illustrate how to use buttons, menus, and gestures in interaction design;
5. illustrate how to create effective information architecture, including content organisation, navigation, and site structure;
6. explain how to design for emotional response as well as the effects of design choices on the environment and society;
7. explain the ethical implications of UI/UX design as well as how to create inclusive and accessible designs;
8. practice creative ways of developing new digital products and solving user problems.

Learning outcomes

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

1. recognize the relationship between user-centred design principles and UI/UX design;
2. carryout UI/UX design decisions, create personas and conduct user research;
3. create efficient prototypes and wireframes that incorporate user feedback;
4. create user interfaces that are simple to use and work well on a variety of platforms and devices;

5. create effective interaction designs, use buttons, menus, and gestures;
6. create a site with a functional navigation and information architecture, including content organisation;
7. consider the emotional response when designing and consider how your choices will affect people and the environment;
8. create new digital products and solve user problems by applying design thinking;
9. create a strong UI/UX design portfolio to show prospective employers.

Course content

Introduction to UI/UX Design Principles. UX Research and Evaluation Tools and Techniques. Wireframing and Prototyping. Usability Testing. Visual Design Principles. Designing For Different Devices and Platforms. Interaction Design. Information Architecture. User Psychology and Engagement. Designing For Emotional Response. Designing For Accessibility. Ethics And Responsibility in Design.

Minimum academic standards

Computer Laboratory.
Computers (1:2 students).
Software e.g., WYSIWYG, Figma, Adobe XD, etc.

PAU-SEN 114 **Linear Algebra for Computer Science** (2 Units Compulsory; LH=30)

Senate-approved relevance

Linear algebra is a branch of mathematics which handles the study of linear functions and equations which are represented through vectors and matrices. Since the 17th Century, there has been a massive development and advances in Linear Algebra as well as in its application to disparate fields in science and engineering. In Computer Science for example, we often talk about linear transformations and to carry out linear transformation of any sort, it is required to understand vector spaces, lines, planes and the idea of mappings which are deeply rooted in Linear Algebra. The solution of linear systems of equations and differential equations as applied in software engineering are easily resolved using Linear Algebra in conjunction with Calculus. Computer Animation, Graphics Design, Game Development etc. all make use of methods in Linear Algebra.

Overview

In this course, we deal with linear transformation from one vector space to another. However, in order to make computation highly tractable, we normally rely on the use of matrices of linear transformation with a proper selection of bases. Thus, this course is concerned with the study of vectors, vector spaces, matrices, linear transformations, determinants and systems of linear equations. The major tool in the study of these components of linear algebra is the mathematics of vectors and matrices which have found applications in many fields, including computer science, engineering, probability and statistics, robotics, optimization, graph theory, genetics, physics, etc.

Objectives

The objectives of the course are to:

1. explain vectors and the various forms in which they appear;
2. illustrate the importance of vectors in computer science;
3. explain matrices and their algebraic structures;
4. illustrate matrices with computer science related problems;
5. describe eigen-values and eigenvectors;
6. illustrate eigen-values and eigenvectors with computer science related problems;
7. explain the idea behind vector spaces and other mathematical concepts tied to it;

Learning outcomes

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

1. explain what vectors are;
2. describe the various mathematical operations performed on vectors as well as solve related problems;
3. explain the nature of matrices and the various algebraic operations performed on matrices as well as solve related problems;
4. solve for the eigenvalues and eigenvectors of square matrices;
5. explain the concept of vector space and its components;

6. describe the rationale behind linear independence, basis, dimension and linear transformation on vector spaces;
7. apply these knowledge areas in computer problem domains e.g., graphics.

Course content

Vectors and its operations. Algebra of matrices. Linear systems and their solutions using matrix-based approaches. Eigenvalues and eigenvectors. Vector space over a real field. Subspaces, linear dependence and independence, basis and dimension. Linear transformations and their representation by matrices. Null spaces.

PAU-SEN 192 **Introduction to Christian Theology** (2 Units Compulsory; LH=30)

Senate-approved relevance

Pan-Atlantic University has a strong Christian identity which is shown in its openness to people of all races and religions. Its mission is "to form competent and committed professionals and encourage them to serve with personal initiative and social responsibility the community in which they work, thereby helping to build a better society in Nigeria and Africa at large". One of the objectives of the university is to also give a well-rounded formation of the human person, which includes some courses in the humanities. The course intends to present the essentials of the Christian faith and morals that are related to contemporary issues that inform the identity of the school and will form the basis of other courses on humanities that the students will be taught. It will also help the students to learn to think deeply about contemporary societal issues.

Overview

Nigeria is a multicultural and multi-religious country. Christians make up roughly half of the population of the country. The purpose of the course is to provide essential background on key aspects of Christian faith and morals, related to contemporary issues, as a means of giving students basic criteria to analyze contemporary situations and form a basis for critical analysis from a Christian perspective.

The course will also help the non-Christians understand some aspects of the Christian faith which fosters societal integration of people of different religions. The purpose of the course is to aid the students to understand the human being in relation to God, the others and the world from a Christian perspective and apply that knowledge to modern-day issues.

Objectives

The objectives of the course are to:

1. state some basic philosophical concepts that are fundamental for theological discussion;
2. discuss some fundamental concepts and mysteries of the Christian faith;
3. explain the place of man in the modern world and correctly judge contemporary issues in the light of the Christian faith;
4. discuss the relationship between law and human conscience;
5. analyse human actions and determine their morality;
6. explain Christian morals and be able to judge situations with clear moral criteria;
7. discuss contemporary moral situations from a Christian perspective.

Learning outcomes

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

1. discuss at least three philosophical arguments about the existence of God;
2. explain the relationship between science and faith using at least three concrete examples;
3. explain three modern theories about the relationship between creation and evolution;
4. analyse human actions to determine their morality based on the three criteria of action, circumstance, and intention;
5. list and explain the Ten Commandments and their implications;
6. explain at least five consequences of mishandling the truth, detraction and defamation;
7. explain five contemporary issues relating to human life and drug use.

Course content

The Existence of God. Divine Revelation. Creation and Evolution. Jesus Christ: both man and God. Eschatology. Human Freedom and Natural Law. Moral Conscience. Factors that determine the Morality of Human Acts. Personal Sin and Responsibility. Influence of the Passions in Human Actions. The Virtues. Introduction to the Ten Commandments. Contemporary human Life issues. Contemporary sexual issues. The morality of Gambling. Contemporary issues regarding handling the truth. Christian Prayer.

Minimum academic standard

A classroom with a projector.

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 30)

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 1 **(2 Units C: LH 30)**

Learning Outcomes

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

1. understand 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: Mathematical Methods II **(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 C: 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.

CSC 203: Discrete Structures **(3 Units C: LH 45)**

Learning Outcomes

At the end of this course, students should 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 **(3 Units C: LH 15; PH 45)**

Learning Outcomes

At the end of this course, student should 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), 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.

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.

PAU-SEN 211 **Computer Hardware and Networking Essentials** (3 Units Compulsory; LH=30; PH=45)

Senate-approved relevance

As reflected in the mission, Pan-Atlantic University seeks to form competent professionals and thus leaves no stone unturned in addressing technical knowledge areas considered relevant to a discipline. In this light, we recognise a good working knowledge of how computers function both from the perspective of hardware as well as networkability, to be important for software engineers who have to program computer systems for effective use. Besides, Computer Science graduates are

also expected to be able to solve basic system problems that arise in the workplace, even where only a good understanding of the functioning of hardware or network systems is required.

Overview

This course is designed to help students become well acquainted with how the computer functions both from hardware and networking perspectives. The treatment of hardware embraces not only classical computer workstation and server systems but also mobile systems. The students will be exposed to the programmable nature of various hardware components and the firmware that drives them; in this way, they will be better positioned to contribute to the requisite firmware development, as they will encounter later in system programming type courses. This course is also designed to help students begin to develop expertise that aligns with the slogan "the network is the computer". Through theory and practical sessions, the students should be in a position to seek industry certification programs in Hardware and Networking essentials, if they so wish.

Objectives

The objectives of the course are to:

1. list and describe all the hardware elements of a computer and how they interconnectedly function;
2. explain the similarities and differences between the Basic Input Output System (BIOS) and Unified Extensible Firmware Interface (UEFI) as software interfaces between an operating system and platform firmware;
3. enumerate the functions of BIOS and UEFI and describe how to update them and manage the options;
4. explain the similarities and differences between Central Processing Unit (CPU) and Graphic Processing Unit (GPU), including their capacity specifications;
5. enumerate different types of Random Access Memory (RAM) and explain their capacity specifications;
6. describe different kinds of USB communication port versions and the differences between them;
7. describe different kinds of display connectors (e.g., HDMI, VGA, USB);
8. enumerate similarities and differences between various types of hardware namely basic computer, mobile phone and AR/VR headset;
9. explain computer networking principles, protocols and practices;
10. describe connectivity models like open systems interconnection (OSI);
11. illustrate basic computer network cabling;
12. develop network and Internet connectivity verification and troubleshooting skills, using network simulation tools like Cisco Packet Tracer.

Learning outcomes

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

1. identify all the hardware elements of a computer and how they interconnectedly function;
2. explain in detail, the specific functions of each of the various components e.g., BIOS, RAM, CPU, GPU, input devices, communication ports like USB ports, storage devices;
3. distinguish between BIOS and other forms of OS firmware interfaces like UEFI;
4. match hardware component specifications with the tasks for which computer hardware is to be deployed;

5. explain similarities and differences between various hardware types namely basic computer, mobile phone and AR/VR headset;
6. distinguish between different kinds of USB communication port versions (e.g., USB-A, USB-B, USB-C);
7. distinguish between different kinds of display connectors (e.g., HDMI, VGA, USB);
8. troubleshoot and optimise firmware settings for various hardware devices, where applicable;
9. explain how computer networking works;
10. explain connectivity models like OSI;
11. implement basic network of computers and connection to the Internet;
12. crimp RJ45 connectors;
13. verify and troubleshoot network and Internet connectivity.

Course content

Overview of computer hardware components and functions. Software interfaces between an operating system and platform firmware (BIOS, UEFI). Processing Units (CPU, GPU). Primary and Secondary Memory. Hardware types. Hardware peripherals. USB connectivity. Display connectivity. Overview of computer networking. Connectivity protocols and models. Network design and cabling. Troubleshooting networks in a simulated environment.

Minimum academic standards

Hardware laboratory with computer systems;
Cables, connectors and crimping tools for networking;
Network simulation tools like Cisco Packet Switcher.

PAU-SEN 212 **Computer Graphics** (3 Units Compulsory; LH=30; PH=45)

Senate-approved relevance

The need for computer graphic programming skills cannot be overemphasised. Computer graphics, which fundamentally refers to visual images produced by computer processing, is applied in a wide variety of systems. The need for it extends from basic graphical user interfaces to complex data visualisation, interaction and manipulation. Besides usage in business intelligence, marketing and entertainment industries, visualisations have become prevalent in science and engineering for modelling complex phenomena as well as in medical imaging. In fact, there is hardly any human computer interaction that does not incorporate some form of computer graphics. Besides, with the emergent spatial Web innovations comes the need for even more computer graphics expertise.

Overview

In this course, students are prepared to have foundational programming knowledge required to drive the development and maintenance of complex computer graphics frameworks like game engines, among others. This course builds on the knowledge of third generation programming language, acquired in previous semesters. The course begins with the creation and use of language in-built graphic primitives and extends to the use of external graphic libraries for more complex graphic systems developments. In addition to still images, motion graphic programming is treated as an animation core. The students are also introduced to the scientific foundation for image rendering along with the importance of GPU usage for higher efficiency.

Objectives

The objectives of the course are to:

1. explain what computer graphics means and the differences between vector and raster graphics;
2. list and describe graphic programming using primitives and libraries, in a third-generation programming language;
3. explain and guide the students on how to create and animate two dimensional (2D) and three dimensional (3D) images, programmatically;
4. explain what rendering is and guide the students on the art of rendering;
5. introduce the use of off-the-shelf software for modelling, rendering and animation of graphic images;
6. describe the various forms of image compression and how image compression is carried out programmatically;
7. describe various conventional image compression formats.

Learning outcomes

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

1. explain what computer graphics means and distinguish between vector and raster graphics;
2. create graphic programs using a third-generation language (preferably C++ or Rust);
3. use graphic primitives available in the in-built graphic library, for coding;
4. program scaling, rotation and translation of graphic objects using in-built graphic library;
5. use external libraries like SDL (Simple DirectMedia Layer), OpenGL and Vulkan for graphic programming;

6. create 2D and 3D images programmatically;
7. animate 2D and 3D images;
8. render graphic objects using external graphic libraries;
9. use GPU (including WebGPU) for rendering;
10. use an industry standard software for a full modelling, rendering and animation workflow experience;
11. explain similarities and differences between lossy and lossless image compression;
12. explain conventional image compression formats;
13. create image compression formats programmatically.

Course content

Introduction to computer graphics concept and types. Language primitives for graphic programming. Still and motion graphics using language primitives. Introduction to external graphic libraries (e.g., SDL, OpenGL and Vulkan). Introduction to rendering. Use of external graphic libraries and GPU for rendering. Introduction to industry standard packages for computer graphics (e.g., blender, Maya). 2D and 3D modelling with industry standard packages. Rendering with industry standard packages. Image compression.

Minimum academic standards

2 Computer workstations per student for lab sessions.
Minimum configuration for computer is Intel core i5 processor with 16GB RAM and equipped with GPU.
Requisite compilers and industry standard packages should be installed in each system.

PAU-SEN 214 **Computer Security Fundamentals** (2 Units Compulsory; LH=30)

Senate-approved relevance

In its mission statement, Pan-Atlantic University seeks "to form competent and committed professionals and encourage them to serve with personal initiative and social responsibility the community in which they work, thereby helping to build a better society in Nigeria and Africa at large". As a means to promote the mission, this course aims to inculcate in students the capacity to understand and analyse security issues relating to the usage, deployment and maintenance of computer systems. Computer systems are the backbone of most organisations. Individuals also have their lives centred now around computers. The emergence of WEB 3.0 has ensured that society is more connected than ever before, through social media adoption and usage on mobile devices.

Overview

This course introduces the most relevant and up to date issues relating to computer security. It identifies the elements of security as related to computer systems and networks. An introduction to computer-based threats and players are explored, with the methodology of executing a risk assessment report also explored. This exploration of risk assessment methods and report writing gives the students an opportunity for practical experience with live production systems and case studies. Security in telecommunication networks, encryption methods and techniques, security models as well as government policies and laws relating to the security of computer systems and networks will also be explored in this course.

Objectives

The objectives of the course are to:

1. explain the concept of computer security;
2. explore the different attitudes to computer security;
3. explain the classes of threats, types of malware and the factors that can cause system breach in computer security;
4. explain cyber stalking, fraud and abuse;
5. explain the preventive measures against cyber fraud;
6. explain the concept of risk analysis;
7. list and describe security control frameworks and best practices;
8. explain the concept of distributed system and its characteristics;
9. describe cloud computing and IoT security;
10. define and classify SQL injection attacks;
11. explain the role of firewalls as part of a computer and network security strategy;
12. list, describe and illustrate the principles of secure coding.

Learning outcomes

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

1. be aware of possible attack modes that can occur in a computer network;
2. execute a risk assessment of a computer system, what security models can be deployed to prevent an attack and what methods can be used to protect and prevent attacks on hardware, software and network systems;
3. identify all types of malwares and how there are deployed to computer systems and networks;

4. explain the unique need for database security, separate from ordinary computer security measures;
5. identify the attack modes and methods deployed by attackers to compromise a computer system;
6. develop a risk assessment report for any computer system and implemented process, while utilising the most current standardised method as defined;
7. explain the role of encryption in protecting user data, while in local storage and also while in transit;
8. explain the role government policies play in the security of computers nationally and globally;
9. describe the relative merits of various choices for firewall location and configurations;
10. distinguish between firewalls and intrusion prevention system;
11. recognize opportunities to apply secure coding principles.

Course content

Fundamental of Computer Security. Risk analysis in Computer Security. Hardware and Software Security Control. Virus. Encryption and Cryptography Techniques. Security models. Telecommunication Security. Distributed System Security. Firewalls. Database Security. Legal issues and current legislation. Government-based security standards. Secure software coding principles.

PAU-SEN 292 **The Nature of Human Beings** (2 Units Compulsory; LH=30)

Senate-approved relevance

The mission of Pan-Atlantic University is "to form competent and committed professionals and encourage them to serve with personal initiative and social responsibility the community in which they work, thereby helping to build a better society in Nigeria and Africa at large". Thus, Pan-Atlantic University is committed to ensuring that its students devote time to courses in humanities and liberal arts that ensure their all-around formation as persons and foster in them the highest regard for human dignity albeit inculcating in them a sense of service for humanity.

Overview

An anthropological crisis is at the root of most of the social and moral problems confronting contemporary society due to a resurgence of doubts as regards the true nature of human beings. This situation calls for intervention in the form of philosophical anthropology to the nagging existential questions such as; Who is a human being? What kind of difference exists between human beings and animals? What is the meaning of life? What is the foundation of human dignity? What is human fulfilment? Is death the end of life? etc.

This course is designed to introduce the students to the philosophical basis of considerations about the human person. It seeks to establish what/who the human person is to bring the students to a due appreciation of the human person and human dignity. The course exposes the students to human potencies and faculties, such as; rationality, understanding, imagination, freedom, Will, and emotions. The students will also be exposed to various conceptions of the human person across different epochs, the idea of death, immortality of the soul, and human destiny.

Objectives

The objectives of the course are to:

1. explain basic concepts in philosophical anthropology;
2. describe human beings as higher animals;
3. describe human rationality with a focus on the Intellect and Will;
4. discuss other aspects of human nature such as; relationality, freedom and responsibility, and human virtues;
5. stimulate conversation and guide the students to arrive at the knowledge of human existence, the meaning of life, and human fulfilment;
6. discuss human dignity and its practical consequences;
7. explore the idea of human death, arguments about the immortality of the human soul, and human destiny.

Learning outcomes

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

1. explain the basic concepts in philosophical anthropology;
2. compare the nature of human beings with the nature of lower animals;
3. explain the operations of the human Intellect and Will;
4. explain human dignity, its foundation, and its practical consequences of human dignity in socio-political, economic, and cultural practices as well as in technology development;
5. demonstrate knowledge of human virtues and their development in the human person;

6. discuss the phenomenon of human death, the immortality of the soul, and how the idea of death shapes one's sense of meaning and value of one's life and that of others.

Course content

Introduction and conceptual clarifications. Human beings as higher animals. Human emotions and emotional Intelligence. The rationality of human beings – intelligence and will. The unity of the human person. Human sexuality. The nature of Human Freedom. Freedom and truth. Freedom and Evil. Human beings as social beings. Expanded view of the social environment – virtual world and metaverse. Human beings as working beings – the objective and subjective dimensions of human work. Human development – the virtues. The dignity of human beings and its practical consequences. Human fulfilment. Existence and the meaning of Life. The phenomenon of Human death.

PAU-SEN 291 **English for Business Purposes** (2 Units Compulsory; LH=30)

Senate-approved relevance

One of the features of a competent professional is the capacity to communicate effectively. English is the most common language of communication in business and professional spheres in Nigeria. A good command of the language's use in professional work, is therefore important, especially in Nigeria, which has a rich variety of languages used in different regions. This course focuses on excellence in writing and business communication, an aspect of education in professional competence that the university aims to achieve among its students. It sets students at an advantage in the professional world as it empowers them to attain English language proficiency according to international standards of excellence.

Overview

English for Business Purposes (EBP) is a key aspect of practical business communicative processes. This course will help the students acquire the knowledge and skills required for their business and work life. This makes teaching them these skills relevant to enable them to perform effectively in their future endeavours.

Additionally, for many different reasons, difficult to engage in modern forms of writing. In this age of new media where business communication is digitally mediated, students need to be proficient in diverse forms of writing to succeed in their business endeavours.

Objectives

The objectives of the course are to:

1. state the importance of English for business purposes for effective communication in the workplace;
2. describe the techniques for writing different business documents;
3. explain how to use different media applications for effective business communication in the 21st century;
4. conduct practical exercises on writing different types of business documents;
5. identify students' communicative challenges for more practical demonstrations.

Learning outcomes

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

1. state at least five importance of English for business purposes for effective communication in the workplace;
2. describe the techniques for writing at least five types of business documents;
3. use at least five different media applications for effective business communication in the 21st century;
4. write at least five different types of business documents for effective communication to a target audience;
5. demonstrate at least three competencies for writing business documents such as memos, business letters, communique, and executive summaries.

Course content

English as a language of global communication. Diversity in writing. Cross-cultural differences in writing. Writing for exporting. Writing reviews. Writing for the Internet. Writing to lead, inspire, and change. Commercial communication. Business documents. Business presentation skills. Giving feedback and performance reviews. Dialogue and protocol in business communication. Interviews and negotiation skills. Language and the new media. Content creation. Learning and memory skills. Effective time management for business executives. Goal setting for business management.

Minimum academic standard

Classroom with projector.

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 peace building 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/boarder 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, objectoriented 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 Systems **(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 structure;
2. describe OS support for processes and threads;
3. recognize CPU scheduling, synchronization, and deadlocks;
4. resolve OS issues related to synchronization 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 programs under Linux to make use of OS concepts related to process synchronization, 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 synchronization; 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.

PAU-SEN 311 **Artificial Intelligence** (2 Units Compulsory; LH=15; PH=45)

Senate-approved relevance

With the emergence of large language models like GPT, among others, Artificial Intelligence (AI) has taken the world by storm. More than ever, it is important to formally train budding software engineers on the rudiments of AI in order to help them better positioned to leverage on it for competitiveness. Besides, they need to be well-positioned to contribute to the future of AI itself.

Overview

This course on the one hand introduces the students to the history, fundamental concepts and techniques of Artificial intelligence, as a discipline. On the other hand, the course takes a practical approach to learning AI, by elucidating its application in a number of areas like intelligent agents, natural language processing, robotics, etc.

Objectives

The objectives of this course are 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.

Learning outcomes

On completion of the 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 content

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.

Minimum academic standards

Computer Laboratory.
Computers (1:2 students).
Lab with Robotics equipment.

PAU-SEN 312 **Compiler Construction** (3 Units Compulsory; LH=30; PH=45)

Senate-approved relevance

The relevance of Compiler Construction to the mission of Pan-Atlantic University is that it equips students with the necessary skills and knowledge to design and develop compilers, interpreters, and other language translation tools. This skill set will enable graduates to develop software programs that can automate processes, increase efficiency, and solve problems faced by organisations in Nigeria and Africa.

By utilising their personal initiative and social responsibility, graduates of this program will be able to develop software solutions that positively impact their communities, thereby contributing to the development of a better society.

Overview

This course covers the fundamental concepts and principles of compiler construction. Students will learn how to design and develop compilers, interpreters, and other language translation tools using different techniques and methods. The course will also cover the process of code optimization and the development of programming language grammars.

Objectives

The objectives of this course are to:

1. explain fundamental concepts in compiler construction;
2. describe the principles of compiler construction;
3. list and explain different compiler construction techniques and methods;
4. describe the design of compilers;
5. describe the design of interpreters;
6. illustrate the development of compilers;
7. illustrate the development of interpreters;
8. explain how to optimise code generated by compilers.

Learning outcomes

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

1. explain fundamental concepts in compiler construction;
2. explain principles of compiler construction;
3. list and describe different compiler construction techniques and methods;
4. design compilers;

5. design interpreters;
6. develop compilers;
7. develop interpreters;
8. implement different compiler construction techniques and methods;
9. develop programming language grammars.

Course content

Introduction to compiler construction. Interpreter design. Lexical analysis. Syntax analysis. Semantic analysis. Intermediate code generation. Code optimization. Code generation. Compiler construction tools and techniques.

Minimum academic standards

Computer Laboratory.
Computers (1:2 students).
Integrated Development Environment (IDE) e.g., Jlex.
Compilers for programming e.g., Java.

PAU-SEN 313 **Formal Methods** (2 Units Compulsory; LH=30)

Senate-approved relevance

Formal Methods is highly relevant to the mission of Pan-Atlantic University, as it equips students with the necessary skills and knowledge to design, specify, and verify computer systems. By ensuring that computer systems are correct, reliable, and safe, graduates of this program can help build a better society in Nigeria and Africa at large.

Furthermore, this course encourages students to think critically, analyse problems, and develop innovative solutions using mathematical techniques. By promoting personal initiative and social responsibility, students will be able to apply their skills to real-world problems faced by organisations and communities, thereby contributing to the development of a better society.

Overview

This course covers the fundamental concepts and principles of Formal Methods. It involves a comprehensive study of mathematical techniques used in the design, specification, and verification of computer systems. Students will learn how to model computer systems using mathematical models, including temporal logic, model checking, and theorem proving.

Objectives

The objectives of this course are to:

1. explain the principles of formal methods;
2. list and describe different formal methods techniques and methods;
3. explain mathematical concepts and logic that underlie formal methods;
4. describe how to design computer systems;

5. describe how to specify computer systems;
6. describe how to verify computer systems;
7. illustrate model checking, theorem proving and program analysis.

Learning outcomes

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

1. explain principles of formal methods;
2. apply different formal methods techniques and methods in designing computer systems;
3. apply different formal methods techniques and methods in specifying computer systems;
4. apply different formal methods techniques and methods in verifying computer systems;
5. develop mathematical models of computer systems using temporal logic, model checking, and theorem proving;
6. analyse and evaluate computer systems using formal methods;
7. analyse and interpret formal models and specifications using mathematical reasoning.

Course content

Introduction to Formal Methods. Mathematical foundations of Formal Methods. Temporal logic and model checking. Theorem proving. Specification Languages. Program analysis. Program verification.

PAU-SEN 314 **Artificial Intelligence in Software Engineering Processes** (2 Units Compulsory; LH=15; PH=45)

Senate-approved relevance

This course is particularly relevant to software engineers who need to learn to leverage on advances in Artificial Intelligence, to enhance productivity. The emergence of large language models like GPT, among others, make it necessary to explore ways to apply AI to make each step in classical software engineering processes, more efficient.

Overview

This course begins with an exploratory literature on AI techniques application in software engineering. It elucidates the need for AI in better ensuring high quality software especially in complex cases. The study of AI applicability spans across IEEE 12207 standard for Information Technology-software life cycle processes. In addition to conceptual examination, the course will involve practical sessions on the use of machine learning models for facilitating various stages in software engineering process.

Objectives

The objectives of this course are to:

1. describe the evolution of AI techniques in software engineering;
2. examine the use natural language processing in software requirement analysis;
3. examine the use of AI techniques in software architecture design;
4. examine the use of AI techniques in coding;
5. examine the use of AI techniques in testing;
6. describe the limitations of AI techniques;
7. describe complementarity between humans and AI.

Learning outcomes

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

1. describe the evolution of AI techniques in software engineering;
2. explain the use natural language processing in software requirement analysis;
3. explain and use of AI techniques in software architecture design;
4. explain and use of AI techniques in coding;
5. explain and use of AI techniques in testing;
6. describe the limitations of AI techniques;
7. describe complementarity between humans and AI.

Course content

History of AI techniques in software engineering. AI techniques in requirements analysis. AI techniques in software architecture design. AI techniques in Coding. AI techniques in testing. Limitations of AI. Human and AI complementarity.

Minimum academic standards

Computer Laboratory.
Computers (1:2 students).

PAU-SEN 392 **Professional and Personal Skills** (2 Units Compulsory; LH=30)

Senate-approved relevance

In order to appropriately resolve many challenges in Africa and the world, there is a need to have professionals who are not only competent but possess soft skills and other qualities which enrich interactions with their colleagues. The goal of our university education extends beyond obtaining technical skills and this course provides an opportunity for preparing students for personal and professional development and for service to the world. The course equips the students with a variety of skills that they need in order to attain the mission of the Pan-Atlantic university to provide holistic formation to students while preparing them for professional life.

Overview

Personal and professional skills contribute greatly to success. This course teaches the meaning of professionalism and its practical implications for the student's chosen career path while exposing the student to ways in which they can be of service to humanity. It teaches skills that lead to personal effectiveness in the professional context and provides practical

strategies for improving self-management and interpersonal relations. It will dwell on a range of skills which are useful in managing ordinary situations and difficult ones in both personal and professional situations. It will teach how to maintain a sense of purpose and direction under pressure and develop the confidence to manage a variety of people and circumstances.

Through this course, the students will learn to make the most of all their capacities. They will be taught to harness their personal talents, energy and time, relative to what is most important, and to channel their resources to achieve what is desirable.

Objectives

The objectives of the course are to:

1. describe the professional and personal skills needed to be effective in professional and personal relationships;
2. identify situations in which soft skills can be acquired for present and future employability;
3. evaluate the most effective practical ways to develop different personal and professional skills;
4. analyse real-life scenarios for applying professional and personal skills in today's world;
5. demonstrate the consequences of the absence or presence of certain skills in relation to personal budgeting and financial responsibility;
6. enumerate effective actions to take in different health emergencies;
7. illustrate what good leadership and team work entail.

Learning outcomes

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

1. describe at least five elements of professionalism and their manifestations within their chosen career path;
2. identify at least five ways in which their chosen career can be of service to communities;
3. identify at least three strengths and five weaknesses associated with each of the four classical types of temperaments;
4. describe five elements of good interpersonal communication differentiating between assertiveness and aggression;
5. describe at least five tips for financial responsibility and making good personal budgets;
6. describe appropriate actions and responses to at least four common medical emergencies;
7. describe two types of good leadership and describe at least five guidelines for good teamwork.

Course content

Professionalism. Job search: interviews, writing applications, CVs, resumes and professional profiles. Professional work as service to the community. Social responsibility. Self-knowledge. Self-esteem & assertiveness. Open-mindedness. Emotional intelligence. Temperaments, character development and personality. Interpersonal communications. Public speaking. Time management. Social etiquette. Cultivating optimal health: mental and physical. Personal budgets and financial responsibility. Leadership and Teamwork.

Minimum academic standards

A classroom with a projector.

PAU-SEN 316: Data Management I (3 Units C: LH 30; PH 45)

Learning Outcomes

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

1. describe the components of a database system and give examples of their use;
2. describe the differences between relational and semi-structured data models;
3. explain and demonstrate the concepts of entity integrity constraint and referential integrity constraint;
4. apply queries, query optimisations and functional dependencies in relational databases;
5. describe properties of normal forms and explain the impact of normalisation on the efficiency of database operations;
6. describe database security and integrity issues and their importance in database design; and
7. explain the concepts of concurrency control and recovery mechanisms in databases.

Course Contents

Information Management Concepts. Information storage & retrieval. Information management applications. Information capture and representation. Analysis and indexing - search, retrieval, information privacy. Integrity and security. Scalability, Efficiency and Effectiveness. Introduction to database systems. Components of database systems. DBMS functions. Database architecture and data independence. Database query language. Conceptual models. Relational data models. Semi-structured data models. Relational theory and languages. Database Design. Database security and integrity. Introduction to query processing and optimisation. Introduction to concurrency and recovery.

Lab work: Practical exercise on information representation, capture, storage and retrieval. Learn how to analyse data and index for easy searching and indexing. Practical on creating database files and models. How to create and use various database designs. How to query the created database. Methods of concurrency and recovery in database. Learn how to secure the database.

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 497: Final Year Project I **(3 Units C: PH 135)**

Learning Outcomes:

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

1. identify researchable project topics in Software Engineering;
2. search and review literature pertinent to identified problem statements;
3. acknowledge and reference sources of information used in the research report;
4. conceptualise and design a research methodology to address an identified problem;
5. determine tools for analysing data collected based on research objectives;
6. write a coherent report on the research conducted;
7. take instruction to accomplish the set goals for the project with the guidance of the research supervisor; and
8. orally present the written project report.

Course Contents

An independent or group investigation to address a Software Engineering problem under the supervision of a lecturer. Before registering, the student must submit a written proposal to the supervisor for review. The proposal should give a brief outline of the project, estimated schedule of completion, and computer resources needed. A formal written report is essential and an oral presentation may also be required. At the end of the semester, the introduction, literature review and methodology employed should be submitted for grading.

SEN 498: Final Year Project II **(3 Units C: PH 135)**

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 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

Minimum Academic Standards

Equipment

The following laboratories together with their software requirements are required for the B.Sc. Software Engineering programme.

Software Engineering Laboratory

The hardware requirements for this laboratory are as follows:

1. PCs with CPU with minimum of 4 GB of main memory, 1 TB HDD (ideally there should be a minimum of one PC per every three students);
2. Multifunctional laser printers (minimum of two) to be networked with the PCs; and
3. External Hard Disk, 500GB (minimum of two).

(Maximum of three students per computer system)

The minimum software requirements are as follows:

Software requirements tools:

1. Requirements gathering tools;
2. UML modelling tools; and
3. Requirements management tools.

Software design tools:

1. UML-based Design tools;
2. Object-oriented Design tools;
3. Design Analysis tools;
4. CASE tools;
5. Process Modelling tools;
6. Project Management tools; and

Software testing tools:

1. Automation Testing tools;
2. Black-box and White-box Testing tools;
3. Unit testing, Integration Testing and System Testing tools;
4. Regression Testing tools;
5. Test Generation tools; and
6. Test Management tools.

Software Construction and Development Laboratory

The hardware requirements the laboratory are as follows:

1. PCs with CPU with minimum of 4 GB of main memory, 1 TB HDD (ideally there should be a minimum of one PC per every three students)
2. Multifunctional laser printers (minimum of two) to be networked with the PCs

3. External Hard Disk, 500GB (minimum of two)

(Maximum of three students per computer system)

The minimum software requirements are as follows:

1. Integrated Development Environments (IDEs) with support for different programming languages
2. Debuggers, Automated Bug-finders, Programme Differencing tools
3. Static and Dynamic Programme Analysis tools
4. Compilation Managers and Build Scripts
5. Software Construction Collaboration tools
6. Software Development Frameworks
7. Cloud Tools

Digital Logic Laboratory

The digital logic or hardware laboratory should provide facilities required for hardware-related practicals. Requirements for the digital logic laboratory include:

1. NAND, NOR, XOR, AND, OR gates
2. Multiplexers
3. Master-slave flip-flops
4. Digi-Designer Logic Board, etc
5. Dual–trace oscilloscope
6. Digital Proto-Board
7. Computer casing
8. Motherboard
9. Microprocessor Emulator System
10. ROMs/RAMs
11. Hard drives
12. CD ROMs
13. Display screens
14. Fans
15. Connectors/Jumpers, etc.

PAU-SEN 411 **Machine Learning** (2 Units Compulsory; LH=15; PH=45)

Senate-approved relevance

In today's world, data is king, and machine learning is the key to unlocking its potential. From finance to healthcare, from e-commerce to autonomous vehicles, machine learning techniques are being applied in a wide range of industries to improve decision-making, automate tasks, and gain insights from large datasets. With the rise of big data and the need to process vast amounts of information, the demand for machine learning skills is only going to increase in the coming years.

Machine learning is also becoming increasingly important in emerging fields such as robotics, the Internet of Things (IoT), and augmented and virtual reality. As the boundaries between the digital

and physical worlds continue to blur, the ability to design and implement intelligent systems will become a critical skill.

Overview

This course provides an introduction to the principles, algorithms, and applications of machine learning, which is a crucial skill for anyone interested in working with data. It is designed to teach students how to develop and evaluate machine learning models and apply them to real-world problems. By the end of the course, students will have a solid knowledge of the major machine learning techniques, including supervised learning, unsupervised learning, and reinforcement learning. The course also covers the essential concepts of feature engineering, model selection, evaluation, interpretation, and optimization.

In the course, students will work on practical assignments and projects that involve solving real-world problems using machine learning techniques.

Objectives

The objectives of the course are to:

1. explain the fundamental concepts of machine learning;
2. describe the historical background of machine learning;
3. state practical applications and potential limitations of machine learning;
4. show how to utilise machine learning techniques to address real-world problems by applying appropriate methods and tools for model development and evaluation;
5. help students develop critical thinking skills and apply them to machine learning problems effectively;
6. show students how to effectively communicate about machine learning concepts and techniques to others;
7. describe the challenges involved in deploying machine learning models in production environments, including working with real-world data, managing model performance, and ensuring reliability and robustness.

Learning outcomes

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

1. explain in details the core principles, algorithms, and techniques of machine learning
2. use machine learning principles, algorithms and techniques to solve real-world problems;
3. develop, implement, and evaluate machine learning models using appropriate tools and frameworks, and critically analyse their performance using relevant metrics;
4. use advanced problem-solving skills and critical thinking to identify and address complex machine learning challenges;
5. communicate about machine learning concepts and techniques, both verbally and in writing, to diverse audiences;
6. evaluate the ethical implications and limitations of machine learning, and be able to assess the potential impact of machine learning on society;
7. collaborate effectively in teams to design, develop, and deploy machine learning solutions to real-world problems, demonstrating strong project management and teamwork skills.

Course content

Introduction to machine learning and its history. Data pre-processing and feature engineering. Supervised learning: linear regression, logistic regression, decision trees, k-nearest neighbours, support vector machines, neural networks. Unsupervised learning: k-means clustering, hierarchical clustering, principal component analysis. Reinforcement learning. Model selection and evaluation: overfitting, underfitting, cross-validation, hyperparameter tuning. Interpretation and visualisation of machine learning models. Optimization techniques. Ethics and limitations of machine learning. Deploying machine learning models in production environments.

Minimum academic standards

Computers with at least 8GB of RAM.
A dedicated graphics processing unit (GPU) with at least 4GB of VRAM (optional, but highly recommended).
Sufficient hard drive space to store large datasets and model parameters.
Python 3.x, Numpy, Pandas, Jupyter Notebook or equivalent IDE.

PAU-SEN 412 **Survey of Programming Languages** (3 Units Compulsory; LH=30; PH=45)

Senate-approved relevance:

Survey of Programming Languages is highly relevant to the mission of Pan-Atlantic University, as it equips students with the necessary skills and knowledge to understand the evolution of computer programming languages, their traits and behaviour, several programming styles and paradigms and how to represent instructions in the major programming languages. By understanding the different structures of programming languages, graduates of this program can help write codes in one language and represent them in other languages thereby leading to more efficient programmers output in the Nigerian software market in particular and Africa at large.

Furthermore, this course encourages students to think logically, represent problems as algorithms and develop innovative solutions using the different programming styles or paradigms learnt. By enhancing good programming skills and attributes, students will be able to apply their skills to real-world problems faced by organisations and communities, thereby contributing to the development of a better society.

Overview

Several Programming Languages (PLs) have been developed and most of them are still in use while some are already obsolete. However, some of these PLs have similarity while some are entirely different from each other. The comparison between the PLs led to the categorization of programming languages into different classes, styles or paradigms. Survey of Programming Languages (SPL) is a course on the fundamental principles of programming languages, introduction to fundamental principles and techniques in programming languages design and implementation. It handles the programming paradigm and historical pattern of programming in addition to a broad comparison of the different programming paradigms. The course elaborates on language structure, syntax and semantics, data type and data structure.

Objectives

The objectives of the course are to:

1. explain good programming styles and ideas;
2. list and explain programming paradigms;
3. describe what makes one language different from another;
4. explain how to choosing appropriate languages;
5. describe how to master the learning of new languages;
6. explain the significance of programming implementation;
7. explain the core rudiments of code representation in several programming languages.

Learning outcomes

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

1. describe how the different types of programming languages evolved;
2. enumerate and explain programming paradigms in detail;
3. explain what makes each language different from the other;
4. describe the different ways of writing algorithms in each language;
5. recognize the different coding environments of each language;
6. compile or interpret codes in each language;
7. express the syntax of each language and its semantics;
8. select ideal programming languages for any given project.

Course content

Overview of programming languages. History of programming languages. Brief survey of programming paradigms (Procedural languages, Object-oriented languages, Functional languages, Declarative – non-algorithmic languages, Scripting languages). Effects of scale on programming methodology. Language Description: Syntactic Structure (Expression notations, abstract Syntax Tree, Lexical Syntax, Grammars for Expressions, Variants of Grammars), Language Semantics (Informal semantics, Overview of formal semantics, Denotation semantics, Axiomatic semantics, Operational semantics). Declarations and types: The concept of types, Declaration models (binding, visibility, scope, and lifetime), Overview of type-checking, Garbage collection. Abstraction mechanisms: Procedures, function, and iterations as abstraction mechanisms. Parameterization mechanisms (reference vs. value). Activation records and storage management. Type parameters and parameterized types. Modules in programming languages. Object oriented language paradigm. Functional and logic language paradigms.

Minimum academic standards

Computer Laboratory.
Computers (1:2 students).
Integrated Development Environment (IDE) e.g., VSCode.
Compilers for programming e.g., JDK, GCC, Jython.

PAU-SEN 413 **Human-Computer Interaction** (2 Units Elective; LH=30)

Senate-approved relevance

Human-Computer Interaction (HCI) is an interdisciplinary field that can prepare software engineering students for careers in a variety of fields where understanding user behaviour and designing effective interactive systems is essential. Teaching HCI to software engineering students will equip them with the skills and knowledge required to create effective, user-centred software. User-centred design is becoming increasingly important as technology becomes more integrated into our lives, and HCI principles are critical for creating usable and effective software. Understanding HCI can assist students in developing critical thinking and problem-solving skills that can be applied to a variety of software development challenges. Furthermore, as new technologies emerge, HCI knowledge will be required to design and develop effective interactive systems.

Overview

This course investigates the principles, theories, and practices that underpin interactive system design and evaluation. It covers fundamental human-computer interaction concepts like user-centred design, interaction design, and usability engineering, as well as advanced topics like natural language processing, machine learning, and virtual and augmented reality. Students will gain a critical understanding of the field and learn how to apply this knowledge to design, develop, and evaluate interactive systems that meet the needs and preferences of their users.

Objectives

The objectives of the course are to:

1. explain the foundational principles of HCI;
2. describe how to design and evaluate interactive systems;
3. illustrate how to use user research, prototyping, usability testing and heuristics for evaluation of interactive systems;
4. explain in detail the role of user experience (UX) design in software development;
5. impact critical thinking and problem-solving skills;
6. elucidate ethical and social issues in HCI, such as privacy, security, bias, and the impact of technology on society;
7. describe emerging trends and technologies in HCI, such as virtual and augmented reality, wearable computing, and natural language processing.

Learning outcomes

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

1. explain and apply fundamental human-computer interaction (HCI) concepts such as user-centred design, usability, and accessibility;
2. design and evaluate interactive systems using methods such as user research, prototyping, usability testing, and heuristic evaluation;
3. explain the importance of user experience (UX) design in software development and apply UX design principles to create successful and interesting software products;
4. recognize and resolve HCI-related problems in software development by applying critical thinking and problem-solving techniques;
5. identify and discuss HCI-related ethical and social issues such as privacy, security, bias, and the impact of technology on society;
6. express the ethical and social implications of interactive systems;

7. describe emerging trends and technologies in human-computer interaction (HCI), such as virtual and augmented reality, wearable computing, and natural language processing, and develop successful and interesting interactive systems.

Course content

Introduction to Human-Computer Interaction. Paradigms of Interaction. User-Centred Design. Interaction Design Basics. Usability Engineering. User Interface Design. Design Rules. Evaluation Techniques. Hierarchical Task Analysis. Adaptation of Human-Computer Interaction in Conventional Software Life Cycle. Collaborative and Distributed Systems. Natural Language Processing. Machine Learning in Human-Computer Interaction. Virtual and Augmented Reality. Ethical Issues in HCI. Emerging Technologies and Future Trends.

PAU-SEN 414 **Deep Learning** (2 Units Elective; LH=15; PH=45)

Senate-approved relevance

As technology continues to evolve, there is an ever-increasing demand for skilled professionals in the field of artificial intelligence. Deep learning, which involves the training of neural networks with large datasets to enable machines to learn and make decisions, is at the forefront of this technological revolution. The applications of deep learning are wide-ranging and include natural language processing, image and speech recognition, autonomous vehicles, and predictive analytics.

Overview

This course is designed to give students a solid understanding of the fundamentals of neural networks and deep learning, as well as the latest developments in the field. Throughout the course, students will explore the biological inspiration behind neural networks, learning about the way that the brain processes information and how that has influenced the development of artificial neural networks.

Students will also learn about the architecture of common network types, including feedforward networks, convolutional networks, and recurrent networks, and the specific applications for which they are best suited. They will gain practical experience in implementing these networks using popular deep learning frameworks such as TensorFlow and PyTorch.

Objectives

The objectives of the course are to:

1. explain the concept of neural networks and deep learning, including their applications, limitations, and underlying principles;
2. describe different network architectures and their applications, and evaluate the performance of different models for specific tasks;

3. illustrate the training of neural networks using popular tools and frameworks, such as TensorFlow and PyTorch, and introduce best practices for model selection, hyperparameter tuning, and optimization;
4. show recent advances and emerging trends in the field, such as generative adversarial networks, autoencoders, and capsule networks, and critically evaluate their potential impact and applications;
5. identify and analyse different network architectures and their specific applications, and develop the ability to choose the appropriate architecture for a given problem;
6. describe up-to-date advancements in neural networks and deep learning, and their implications for the broader field of artificial intelligence and society as a whole;
7. explain ethical considerations and potential biases in the use of deep learning.

Learning outcomes

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

1. explain the principles and concepts of neural networks and deep learning;
2. design and implement various types of neural network architectures for specific applications;
3. train neural networks using backpropagation and gradient descent, as well as apply regularisation and overtraining techniques to improve performance;
4. investigate and apply alternative training methods such as attention, feedback alignment, synthetic gradients, decoupled network interfaces, and transfer learning;
5. evaluate and implement various network architectures, including generative adversarial networks, autoencoders, restricted Boltzmann machines, capsule networks, and spiking networks;
6. apply critical thinking skills to analyse and evaluate the performance of neural network models in different scenarios;
7. recognize and address ethical considerations in the design and deployment of neural network models;
8. communicate effectively about neural networks and deep learning concepts and techniques both verbally and in writing.

Course content

Introduction to Neural Networks and Deep Learning. Biological Brain and Neural Networks. Perceptron and multi-layer perceptron. Network Architectures Overview. Feed-forward networks. Convolutional neural networks. Recurrent neural networks. Memory cells. LSTMs. Neural Network Training. Alternative Training Methods. Decoupled network interfaces. Transfer learning. Generative adversarial networks. Autoencoders. Restricted Boltzmann machines. Capsule networks. Spiking networks.

Minimum academic standards

Computers with at least 8GB of RAM, A dedicated graphics processing unit (GPU) with at least 4GB of VRAM (optional, but highly recommended).
Sufficient hard drive space to store large datasets and model parameters.
Python 3.x, TensorFlow, Keras, Jupyter Notebook or equivalent IDE.

PAU-SEN 415 **Game Design and Development** (2 Units Elective; LH=15; PH=45)

Senate-approved relevance

The global game industry is huge. The world of games is two pronged: game for gaming environment and game for gamification of non-gaming environments like learning. According to Statista.com, projected gaming market volume by 2027 amounts to US$482.30 billion. Gamification market size is also projected by globenewswire.com to be in billions of US Dollars ($58.8 billion) by 2028. These projections reflect the importance attached to games. Game developers are in high demand. So, there is a need to develop skilled persons that can competitively design and develop game systems in professionally and ethically sound ways.

Overview

In this course, students will leverage their existing programming skills to build games with particular emphasis on 3D. The course treats fundamentals of game development using game engines like Unity. In this course, students will learn how to create 3D game environments, characters, and interactions using the game engine's powerful and intuitive tools. Throughout the course, students will have the opportunity to work on their own game projects, applying the concepts and techniques they've learned in each lesson. By the end of the course, students will have a solid understanding of the game engines' capabilities and be able to create their own games from scratch.

Objectives

The objectives of the course are to:

1. explain the concept of game design;
2. explain the concept of gamification design;
3. describe the differences between game and gamification;
4. describe software development process for game development;
5. illustrate game design for game environment;
6. illustrate game design for gamification environment;
7. explain the Math for game development;
8. show how to develop games leveraging on game engines;

Learning outcomes

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

1. explain the concept of game design;
2. explain the concept of and gamification;
3. design games for game environment;
4. design games for gamification environment;
5. describe software development process for game development;
6. apply Math in game development;
7. develop games using an industry standard game engine.

Course content

Game and gamification design concepts. Game genres. Introduction to Game design. Introduction to gamification design. Game audio. Software engineering methods for game design. Getting to know the game engine of choice (e.g., Unity, Unreal). Game engine installation. Programming review for game engine (e.g., C# for Unity, C++ for Unreal). Variables. Relational operators. Assessing Logic. Conditions. Branching. If and Switching branches. Looping. Iteration. Inheritance. Composition. Understanding 3D coordinate space. Object Transforms. Collision detection. Scripting. Parenting. Rotation. Arrays. List. Reference, and Value types. Coroutines. Trigonometry.

Minimum academic standards

1 Computer workstation per student for lab sessions.

Minimum configuration for computer is Intel core i5 processor with 16GB RAM and equipped with GPU.

Requisite compilers and industry standard packages should be installed in each system.

PAU-SEN 416 **Computer Vision and Image Processing** (3 Units Elective; LH=30; PH=45)

Senate-approved relevance

Computer vision and image processing are two closely related fields in software engineering that deal with the analysis and manipulation of digital images. Image processing is the manipulation of digital images using mathematical operations to extract useful information, enhance image features, or remove unwanted artefacts. Examples of image processing tasks include image filtering, edge detection, image segmentation, and image compression. Both image processing and computer vision rely on a combination of mathematical and computational techniques, including signal processing, machine learning, computer graphics, and artificial intelligence. These techniques enable computers to analyse and interpret images and video, providing valuable insights and applications in fields such as medicine, surveillance, robotics, and autonomous vehicles.

Overview

This course introduces the students to computer vision and image processing, which involves the development of algorithms and techniques that enable computers to interpret and understand the content of digital images and video. The course involves introducing students to a range of techniques and algorithms that can be used to process and analyse digital images. This can include tasks such as image enhancement, segmentation, feature extraction, object detection, and image recognition. Computer vision aims to replicate the human visual system's ability to recognize and interpret visual information, such as object detection, facial recognition, scene understanding, and image classification.

Objectives

The objectives of the course are to:

1. explain the fundamentals of digital images;
2. describe how to enhance and manipulate images;
3. explain image segmentation and feature extraction;
4. explain object detection and recognition;
5. describe relevant tools and software for computer vision and image processing;
6. apply deep learning algorithms to image processing;
7. describe how to implement efficient Convolutional Neural Networks (CNN).

Learning outcomes

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

1. explain the properties of digital images, including resolution, colour models, and pixel values;
2. identify techniques for enhancing image quality, adjusting contrast and brightness, and removing noise or artefacts;
3. identify and extract meaningful features from images, such as edges, corners, and textures;
4. detect and recognize objects in images using techniques such as template matching, Haar cascades, and deep learning-based methods;
5. use tools and software commonly used in computer vision and image processing, such as OpenCV, Python libraries such as PIL, Scikit-image, and TensorFlow;
6. apply computer vision and image processing techniques to real-world problems and develop creative solutions;
7. implement efficient CNN.

Course content

Digital Image Fundamentals: Understanding the basics of digital image representation and the properties of images, including colour models, resolution, and pixel values. Image Enhancement: Techniques for improving the quality of digital images, including filters, histogram equalization, and contrast adjustment. Image Segmentation: Dividing an image into meaningful parts or regions, often using techniques such as thresholding, edge detection, and clustering. Feature Extraction: Identifying and extracting important features from images, such as edges, corners, and blobs. Object Detection: Techniques for identifying objects or regions of interest in an image, such as template matching, Haar cascades, and deep learning-based methods. Image Recognition: Using machine learning algorithms, such as convolutional neural networks (CNNs), to classify or recognize objects in images.

Minimum academic standards

Computer Laboratory.
Computers (1:2 students).
Software e.g. OpenCV, Python libraries: PIL, Scikit-image, and TensorFlow.

PAU-SEN 417 **Introduction to Data Science and Engineering** (3 Units Elective; LH=30; PH=45)

Senate-approved relevance

The field of data science has rapidly become an essential component of modern society, as it enables individuals and organisations to derive valuable insights from complex data sets. This course is designed to provide students with a comprehensive understanding of data science concepts and techniques, enabling them to apply data-driven decision making in a range of settings. The course emphasises the practical application of data science techniques, with a focus on real-world data sets and industry-standard tools and software. Upon completion of the course, students will have developed skills that are highly sought after in a wide range of industries and sectors, including finance, healthcare, marketing, and technology. The relevance of this course has been recognised by the Senate as an important component of a modern, future-focused curriculum, with a growing demand for graduates who have expertise in data science.

Overview

Data science has become one of the most sought-after and exciting fields in the modern era. As the world generates an unprecedented amount of data every day, organisations require skilled data scientists to process and analyse that data, to find insights and uncover hidden patterns that can drive innovation, business growth, and informed decision-making. This course is designed to introduce students to the fundamental concepts and techniques of data science, including data collection, pre-processing, processing, analysis, and visualisation. Students will gain practical experience with a variety of tools and techniques and learn how to work with real-world data sets to make data-driven decisions using appropriate statistical and machine learning techniques. Whether they are interested in pursuing a career in data science or simply want to gain the skills to make data-driven decisions in their personal or professional life, this course will provide them with the

knowledge and hands-on experience necessary to succeed. The students will be further exposed to big data engineering, especially as it pertains to data pipelining and ETL for Data Science.

Objectives

The objectives of the course are to:

1. explain the concept of data science, and data science workflow, from data collection to data visualisation and everything in between;
2. demonstrate different data processing techniques, including data cleaning, wrangling, and transformation;
3. describe data analysis and exploration techniques, including exploratory data analysis, statistical inference, and hypothesis testing;
4. show how data visualisation and communication are done, using various tools and technologies;
5. apply different machine learning algorithms and statistical models to real-world data problems;
6. evaluate the effectiveness of various data science approaches and methodologies;
7. describe ethical considerations and challenges in working with data, including data privacy and bias;
8. describe and illustrate infrastructure engineering for big data pipelining.

Learning outcomes

Upon completion of this course, students will be able to:

1. apply data pre-processing techniques to clean and transform raw data into a format suitable for analysis;
2. perform exploratory data analysis to identify patterns and relationships in data, and communicate insights through effective visualisation;
3. apply appropriate statistical methods to make inferences and draw conclusions from data;
4. develop and implement machine learning models to solve real-world problems, and evaluate their performance using appropriate metrics;
5. use advanced techniques such as feature engineering, ensemble methods, and deep learning to build more complex models;
6. deploy and communicate data-driven insights and models to stakeholders in a clear and effective manner;
7. apply ethical considerations in all stages of the data science process, including data collection, analysis, and communication;
8. describe and illustrate infrastructure engineering for big data pipelining.

Course content

Introduction to Data Science: Overview of Data Science, Data Science Activities, Sources of Data. Use Cases and Performance Evaluation: Data Science Use Cases (DSUCs), Performance Evaluation. Data Pre-processing: Transmission of Data, Data Quality, Cleansing, and Transformation, Data Visualization. Data Processing: Stages of Data Processing, Methods and Types of Data Processing, Output Formats of Processed Data. Machine Learning Techniques: Principal Component Analysis, Cluster Analysis, Linear Regression, Time-Series Forecasting, Transformation Approaches. Artificial Intelligence Techniques: Support Vector Machines, Artificial Neural Networks, Further Approaches.

Minimum academic standards

Computers with at least 8GB of RAM, A dedicated graphics processing unit (GPU) with at least 4GB of VRAM (optional, but highly recommended).
Sufficient hard drive space to store large datasets and model parameters.
Python 3.x, Numpy, Pandas, Jupyter Notebook or equivalent IDE.

PAU-SEN 418 **IoT and Edge Computing** (2 Units Elective; LH=15; PH=45)

Senate-approved relevance

According to statista.com, the number of IoT devices connected to the Internet was reported to have surpassed non-IoT devices in 2020. The install base of IoT has further been projected to reach 30.9 billion by 2025, which will imply a sharp rise from 13.8 billion estimated for 2021. These figures reflect the growing importance attached to IoT in different sectors of society as the implementation becomes increasingly more feasible with 5G network rollout. For a present-day computer scientist, a neglect of this subject matter could constitute a significant knowledge gap. Closely related to IoT growth is the growing perceived need to bring data processing and AI engines closer to the devices rather than centralise these engines in the cloud premises, a phenomenon that has become conventionally referred to as edge computing.

Overview

This course has been designed to help the students acquire working knowledge of three of the interrelated technology domains (IoT, cloud and edge computing) that are shaping the future of solutions, in a highly connected world. The course is approached primarily from the perspective of a computer scientist with emphasis on requisite programming skills. The course involves practical lab sections where the students will design, develop and deploy solutions in real devices as well as the cloud. They will also learn what it means in practice to provision infrastructure as service in the cloud and deploy scalable solutions using tools like Kubernetes.

Objectives

The objectives of the course are to:

1. explain the notions of Internet of Things (IoT), cloud computing and edge computing as well as the differences between them;
2. describe the evolution of IoT and edge computing;
3. explain the concept of embedded systems;
4. describe and illustrate how to implement embedded systems using microprocessor boards like Raspberry Pi and third generation programming language like Python, JavaScript, C/C++ and Rust;
5. describe and illustrate how to install and configure operating systems designed for IoT devices and edge computing;
6. describe and illustrate how to implement software solutions on edge devices;
7. describe the various forms of cloud computing and how to deploy scalable solutions to the cloud;

8. describe and illustrate how to orchestrate IoT related services using software tools like Node-Red, IFTTT (If This Then That).

Learning outcomes

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

1. describe IoT, cloud computing and edge computing as well as how they are related;
2. describe the evolution of IoT;
3. explain why edge computing is gaining prominence even though IoT and cloud computing integration has long been established;
4. explain in detail what an embedded system is;
5. implement embedded systems using microprocessor board (e.g., Raspberry Pi) and a third-generation programming language like Python, JavaScript, C/C++ and Rust;
6. install and configure standard operating system (e.g., Ubuntu core, Windows 10 IoT Core, KataOs) on microprocessor boards (e.g., Raspberry Pi);
7. implement solutions (e.g., machine learning model) on edge devices (e.g., Raspberry Pi with relevant sensor);
8. explain cloud provisioning types e.g., IaaS, PaaS and SaaS;
9. deploy solutions to the cloud;
10. use cloud development operation tools like Kubernetes;
11. orchestrate IoT related services using software tools like Node-Red, IFTTT.

Course content

Comparative description of IoT, cloud computing and edge computing. Embedded systems concept. Embedded system implementation on microprocessor board. Using standard operating systems in embedded systems. Developing edge solutions. Notion of cloud computing. Provisioning of cloud infrastructure. Orchestrating IoT related services.

Minimum academic standards

1 Computer workstation per 5 students.
1 Raspberry Pi with in-built Wi-Fi per 5 students.
1 each of MicroSD memory card, keyboard, micro-USB power supply, HDMI cable, camera module, per Raspberry Pi.
Sensor devices for environmental input illustration.

PAU-SEN 419 **Advances in Web, Mobile and Blockchain Development** (3 Units Elective; LH=30; PH=45)

Senate-approved relevance

The Web which was invented decades ago continues to be relevant today and is solidly projected into the future as it transits through the hypothetical versions tagged Web1, Web2 and Web3. Closely associated with this transition to Web3 is the Blockchain as the underlying data structure. Complementary to the Web and Blockchain platforms are mobile devices (phones, AR/VR sets) which put solutions and access to platforms in the hands of the mobile user. We consider an atomic

treatment of these three dimensions in a course, to be important for expertise in full-stack solutions development, needed now and in the future.

Overview

This course involves a deep dive into full-stack development embracing both Web2 and Web3. In order to inspire creativity, the students will be taken through the innovative history of the Web, from stage to stage. The course should equip them with the requisite skills required to become reputable solutions providers, at the service of global demands, while at the same time positioned to solve our local problems. The course should also expose the students to the academic questions in Web3 and blockchain evolution, with the potential to pursue post-graduate research in these areas.

Objectives

The objectives of the course are to:

1. describe in detail, the historical evolution of the Web from inception to Web3;
2. explain in scientific detail, what blockchain is and the benefits and drawbacks;
3. describe blockchain oracles and their importance in Web3 solutions stack;
4. illustrate the design, development and deployment of server-side software applications for Web2;
5. explain what it takes to build a blockchain;
6. describe and illustrate the development of blockchains;
7. illustrate the design, development and deployment of decentralised applications (dApps) for blockchains;
8. illustrate the design, development and deployment of mobile phone applications;
9. illustrate the design, development and deployment of progressive Web applications;
10. describe development for VR/AR devices.

Learning outcomes

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

1. explain the historical evolution of the Web from inception to Web3;
2. explain in detail, blockchain and related blockchain oracles (e.g., IPFS);
3. identify the respective solutions stack for Web2 and Web3;
4. design, write and deploy simple to complex server-side software applications for Web2, in third-generation languages;
5. create blockchains using third-generation languages like Rust;
6. design, write and deploy decentralised applications for blockchains;
7. design and develop native mobile applications;
8. create simple to complex progressive Web applications;
9. create and deploy applications for selected AR/VR devices.

Course content

The evolution of the Web. Blockchain and related oracles. Web2 solutions stack. Web3 solutions stack. Server-side applications for Web2. Creating blockchains. Creating decentralised applications for blockchains. Mobile applications. Progressive Web applications. AR/VR applications.

Minimum academic standards

1 Computer workstation per two students for lab sessions.
Minimum configuration for computer is Intel core i5 processor with 16GB RAM.
Requisite compilers, mobile simulators and industry standard packages should be installed in each system, for development.
Networked systems for decentralised blockchain and oracles illustrations.
Mobile phones (personal phones recommended).
Two VR headsets.

PAU-SEN 420: Data Management II (2 Units E: LH 15; PH 45)

Learning Outcomes

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

1. explain the principles and best practices of managing data with efficiency and effectiveness;
2. demonstrate knowledge of SQL and NoSQL;
3. explain data warehouse concepts, methodologies and tools; and
4. explain data mining architecture and applications.

Course Contents

Rational Databases: Mapping conceptual schema to relational schema; Database Query Languages (SQL) and NoSQL, Concept of functional dependencies & multi-valued dependencies. Transaction processing; distributed databases, XML and semantic Web. Data warehousing. Introduction to data science. Introduction to Data Warehouse, OLTP Systems; Differences between OLTP Systems and Data Warehouse: Characteristics of Data Warehouse; Functionality of Data Warehouse: Advantages and Applications of Data Warehouse. Advantages, Applications: Top- Down and Bottom-Up Development Methodology: Tools for Data warehouse development: Data Warehouse Types. Introduction: Scope of Data Mining: What is Data Mining. How Data Mining Works, Predictive Modelling: Data Mining and Data Warehousing: Architecture for Data Mining: Profitable Applications: Data Mining Tools.

Lab work: Practical exercises on basic R commands and data structures for manipulating data; how to read data from multiple formats in and out of R, using loops, conditional statements, and functions to automate common data management tasks. Exercises on how to clean and manage multiple complex datasets, manipulate textual data, basic web scraping techniques, for both standard web pages and the Twitter API. Work on techniques and hardware necessary to manage large datasets efficiently. Practical exercise on managing multiple data sets by example; working with text data; converting long- and wide-format data; and dealing with messy data. R Programming Fundamentals for data I/O and packages, looping and conditional statements, and functions.

PAU-SEN 408: Ethics and Legal Issues in Computer Science (2 Units C: LH 30)

Learning Outcomes

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

1. state laws and regulations related to ethics;
2. identify and explain relevant codes of ethics for computing practice;
3. identify social and ethical issues in different areas of computing practice;
4. review real-life ethical cases and be able to develop ethical resolutions and policies; 5. explain the consequences of ignoring and non-compliance with ethical provisions; and
6. develop a sound methodology in resolving ethical conflicts and crisis.

Course Contents

Addresses social, ethical, legal and managerial issues in the application of Computer Science to the information technology industry. Through seminars and case studies, human issues confronting Computer Science graduates will be addressed. Topics include managerial and personal ethics, computer security, privacy, software reliability, personal responsibility for the quality of work, intellectual property, environment and health concerns, and fairness in the workplace.

SCHOOL OF
SCIENCE AND
TECHNOLOGY
PAN-ATLANTIC UNIVERSITY