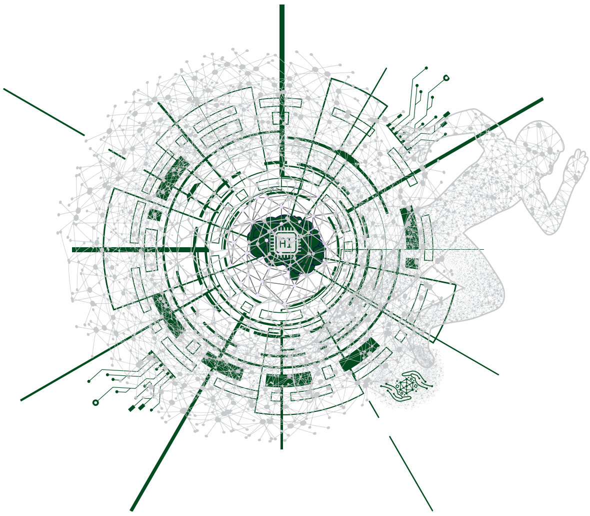
ZIAUDDIN UNIVERSITY
EXAMINATION BOARD

HSSC A
Computer Science
Syllabus



For exams in 2026 & onwards

INTRODUCTION TO ZUEB

The Ziauddin University Examination Board (ZUEB) is not only an awarding body but also a solution-driven educational organization dedicated to upholding the highest standards of academic excellence. ZUEB believes in excellence, integrity, and innovation in education. Established with a vision to foster a robust educational environment, ZUEB is committed to nurturing intellectual growth and development that meets international standards in an effective manner. The Ziauddin University Examination Board (ZUEB) was established through the Government Gazette No. XLI on June 6th, 2018. Its purpose is to ensure high quality, maintain global standards, and align the syllabi with national integrity within Pakistan's examination system. ZUEB manages student appeals, regulates assessments, and reviews policies to maintain high standards.

WHY CHOOSE HSSC-A AT ZUEB?

Ziauddin University Examination Board (ZUEB) offers the HSSC-A (Higher Secondary School Certificate Advance) program, designed for students from international educational backgrounds. This program provides a structured, affordable, and academically strong pathway for learners to align with Pakistan's education system. It allows students to fulfil national curriculum requirements, including Urdu, Islamiyat, Pakistan Studies, or Sindhi, with academic integrity and flexible learning options. ZUEB believes no student should be left behind due to financial limitations or cross-system transitions, and HSSC-A serves as a bridge between past efforts and future ambitions. It is the trusted choice for higher education in Pakistan.

HSSC-ADVANCE COMPUTER SCIENCE

HSSC-Advance Computer Science at ZUEB is a foundation for exploring the digital world and computational systems, designed for students aspiring to pursue higher education in software engineering, information technology, data science, and related fields. The course offers a rigorous, concept-driven curriculum aligned with both national and international standards, covering key topics such as programming, algorithms, data structures, computer hardware, databases, networking, and cybersecurity. Students develop a strong grasp of computational principles and practical applications, while enhancing their logical reasoning, problem-solving, and analytical skills, ensuring they are both examination-ready and future-ready.

Aligned with national and international standards, HSSC-Advance Computer Science at ZUEB equips students with a comprehensive understanding of modern computing, digital systems, and emerging technologies. Designed for students aiming for careers in computer science, software development, artificial intelligence, and information systems, the course builds essential skills in computational thinking, coding, and system design.

Whether you are preparing for admission into top universities for computer science and technology, or planning a career in software engineering, AI, or IT-related research, HSSC-Advance Computer Science ensures you are academically prepared and nationally aligned, with a flexible, student-focused learning approach. Explore more on what HSSC-Advance Computer Science offers ZUEB HSSC-Advance Official Page.

Syllabus Overview

No.	Content	XII	XIII	AO	Exam Details
1	Information and Data Representations	✓	✓	1, 2 and 3	Combination of written exam papers (externally set and marked) and a practical demonstration of skills. XII **Paper 1:** Short answers & structured questions Duration: 1 hour 30 minutes Weighting: 50% **Paper 2:** Short answers & structured questions Duration: 2 hours Weighting: 50% XIII **Paper 1:** Short answers and structured questions Duration: 1 hour 30 minutes Weighting: 25% **Paper 2:** Short answers and structured questions Duration: 2 hours Weighting: 25% **Paper 3:** Short answers and structured questions **Duration:** 1 hour 30 minutes Weighting: 25% **Paper 4 (Practical Skills):** Short answers and structured questions Duration: 2 hours Weighting: 25%
2	Communication and Internet Technologies	✓	✓	1, 2 and 3	
3	Hardware and Virtual Machines	✓	✓	1, 2 and 3	
4	Processor Fundamentals	✓	✓	1, 2 and 3	
5	System Software	✓	✓	1, 2 and 3	
6	Security, Privacy and Data Integrity	✓	✓	1, 2 and 3	
7	Ethics and Ownership	✓	✓	1, 2 and 3	
8	Databases	✓	✓	1, 2 and 3	
9	Computational Thinking, Algorithm Design and Problem-Solving	-	✓	1, 2 and 3	
10	Data Types and Structures	✓	✓	1, 2 and 3	
11	Programming	✓	✓	1, 2 and 3	
12	Software Development	-	✓	1, 2 and 3	
13	Artificial Intelligence (AI)	-	✓	1, 2 and 3	

Description of Assessment Objectives

- **AO1:** Demonstrate a clear understanding of the fundamental principles and concepts of computer science, including abstraction, logic, algorithms, and data representation.

- **AO2:** Apply knowledge of computer science principles and concepts to analyse and interpret problems in computational terms.

- **AO3:** Design, implement, and evaluate computer-based solutions to problems, making informed and justified decisions throughout the process.

Weighting of Assessment Objectives

Assessment Objectives	P1 (%)	P2 (%)	P3 (%)	P4 (%)
AO1	40	35	40	20
AO2	35	40	30	40
AO3	25	25	30	40

1: Information and Data Representations

Aim:
The aim of this content is to equip learners with both theoretical and practical understanding of how data is collected and converted between different bases.

	The learner will:	SLO #	Assessment Criteria - The learner can:	Cognitive levels
1	Understand data representation in the context of binary and character sets	1.1.1	**Transform** positive integers between binary, hexadecimal, and denary.	AO2
		1.1.2	**Interpret** how character sets are used in computer systems.	AO1
		1.1.3	**Examine** how binary data is used in computer systems.	AO1
2	Understand ways in which multimedia is represented through graphics and sound	1.2.1	**Examine** how a bitmap image is represented and stored on a computer.	AO1
		1.2.2	**Describe** how a vector graphic is	AO1
		1.2.3	**Compare** the suitability of bitmap images and vector graphics for a specific task.	AO3
		1.2.4	**Discuss** the process of digitising an analogue sound wave.	AO1
		1.2.5	**Assess** the impact of altering the sample rate and resolution on a sound wave.	AO3
3	Understand the principles of data compression	1.3.1	**Evaluate** the purpose of data compression.	AO3
		1.3.2	**Distinguish** between lossy and lossless data compression.	AO1
4	Be able to demonstrate the practical application	1.4.1	**Explore** methods for converting a number from one base to another.	AO1

	of information and data representations	1.4.2	**Execute** calculations involving binary addition and subtraction.	AO2
		1.4.3	**Use** ASCII, extended ASCII, and Unicode to represent textual data.	AO2
		1.4.4	**Examine** lossy and lossless data compression.	AO1
		1.4.5	**Support** the use of a method in various given scenarios.	AO3
		1.4.6	**Review** appropriate methods of file organisation and file access for a given problem.	AO3
		1.4.7	**Choose** and design a suitable user-defined data type for a given problem.	AO3
		1.4.8	**Transform** binary floating-point numbers to denary and vice versa.	AO2
		1.4.9	**Standardize** floating-point numbers.	AO2
		1.4.10	**Explore** how sound, image, or text can be compressed using run-length encoding.	AO1
5	Understand the concepts of user-defined data types	1.5.1	**Explore** the reasons why user-defined data types are necessary.	AO1
		1.5.2	**Explain** and utilize composite and non-composite data types.	AO2
6	Understand the principles of file organisation and access	1.6.1	**Identify** the various methods of file organisation and file access.	AO1
		1.6.2	**Explain** and apply hashing algorithms.	AO2
7	Understand floating-point numbers, representation, and manipulation	1.7.1	**Outline** the format of binary floating-point real numbers.	AO1
		1.7.2	**Identify** the consequences of binary representation being an approximation of the real number in certain cases.	AO1
		1.7.3	**Explain** how binary representations can lead to rounding errors.	AO1

2: Communication and Internet Technologies

Aim:
The aim of this content is to help learners develop a theoretical understanding of communication and networks, including the internet.

	The learner will:	SLO #	Assessment Criteria - The learner can:	Cognitive levels
1	Understand networks including the internet (introduction to types of network, hardware, and data transmission)	2.1.1	**Investigate** the purpose and benefits of networking devices.	AO1
		2.1.2	**Explore** the characteristics of a LAN and a WAN.	AO1
		2.1.3	**Evaluate** whether a given network is a LAN or a WAN.	AO3
		2.1.4	**Explain** the use, benefits, and drawbacks of cloud computing.	AO1
		2.1.5	**Examine** the characteristics of a client-server and peer-to-peer network.	AO1
		2.1.6	**Explain** the benefits and drawbacks of a client-server and peer-to-peer network.	AO1
		2.1.7	**Support** the choice of a client-server or peer-to-peer network in a given scenario.	AO3
		2.1.8	**Investigate** the characteristics, benefits, and drawbacks of different network topologies.	AO1
		2.1.9	**Contrast** wired and wireless networks.	AO2
		2.1.10	**Categorize** the benefits and drawbacks of both wired and wireless connections.	AO2

		2.1.11	**Examine** the purpose of hardware components that support a LAN.	AO1
		2.1.12	**Recommend** the appropriate components to create a LAN.	AO3
		2.1.13	**Clarify** the role and function of a router in a network.	AO1
		2.1.14	**Identify** collisions in data transmission and explain how Ethernet detects and avoids collisions.	AO2
		2.1.15	**Contrast** the internet and the WWW.	AO2
		2.1.16	**Identify** the hardware required to communicate over the internet.	AO1
		2.1.17	**Evaluate** the use of IP addresses in the transmission of data over the internet.	AO3
		2.1.18	**Compare** the benefits of a URL over an IP address.	AO2
		2.1.19	**Investigate** the role of a DNS in converting a URL to an IP.	AO1
2	Understand different communication protocols and their purposes	2.2.1	**Identify** why a protocol is essential for communication between computers.	AO1
		2.2.2	**Investigate** how protocol is implemented as a stack, with each layer having its own functionality.	AO1
		2.2.3	**Depict** the TCP/IP protocol suite.	AO2
		2.2.4	**Summarize** the purposes of these protocols: HTTP, FTP, POP3, IMAP, SMTP, BitTorrent.	AO1
3	Understand the principles of circuit and packet switching	2.3.1	**Explore** the purpose, benefits, and drawbacks of circuit switching and packet switching.	AO1
		2.3.2	**Evaluate** the use of packet and/or circuit switching in a scenario.	AO3

3: Hardware and Virtual Machines

Aim:
The aim of this content is to empower learners to conduct both theoretical and practical analysis of hardware, virtual machines and their applications.

	The learner will:	SLO #	Assessment Criteria - The learner can:	Cognitive levels
1	Understand the purpose of computers and their components.	3.1.1	**Identify** the differences between primary and secondary storage.	AO1
		3.1.2	**Outline** the items stored in secondary storage.	AO1
		3.1.3	**Differentiate** between RAM and ROM.	AO1
		3.1.4	**Contrast** SRAM with DRAM.	AO2
		3.1.5	**List** the differences between PROM, EPROM, and EEPROM.	AO1
		3.1.6	**Examine** the principal operations of a range of hardware devices.	AO1
		3.1.7	**Explore** the purpose and use of buffers in a range of devices.	AO1
		3.1.8	**Investigate** the uses of sensors and identify appropriate sensors for a scenario.	AO3
		3.1.9	**Differentiate** between a monitoring and control system.	AO2
		3.1.10	**Explore** the use and function of a monitoring and control system in a given situation.	AO3

		3.1.11	**Discover** and define the functions of: NOT, AND, OR, NAND, NOR, and XOR (EOR) truth tables.	AO1
		3.1.12	**Examine** Reduced Instruction Set Computers (RISC) and Complex Instruction Set Computers (CISC) processors.	AO1
		3.1.13	**Highlight** the importance and use of pipelining and registers in RISC processors.	AO1
		3.1.14	**Explore** the four basic computer architectures (SISD, SIMD, MISD, and MIMD).	AO1
		3.1.15	**Discuss** the characteristics of massively parallel computers.	AO1
		3.1.16	**Discuss** the concept, benefits, and limitations of a virtual machine.	AO1
2	Be able to demonstrate the practical application of hardware and virtual machines.	3.2.1	**Utilize** the NOT, AND, OR, NAND, NOR, and XOR logic gate symbols to create the truth table for each of the logic gates.	AO2
		3.2.2	**Build** a logic circuit and logic expression.	AO3
		3.2.3	**Develop** truth tables for logic circuits, including half adders and full adders.	AO3
		3.2.4	**Explain** the function and design a truth table for a flip-flop (SR, JK).	AO2
		3.2.5	**Apply** Boolean algebra to manipulate Boolean expressions.	AO2
		3.2.6	**Predict** the use of, and apply a Karnaugh map (K-map).	AO3

4: Processor Fundamental

Aim:
The aim of this content is to equip learners with the ability to perform both theoretical and practical analysis of CPU architecture, assembly language, and bit manipulation.

	The learner will:	**SLO #**	**Assessment Criteria - The learner can:**	**Cognitive levels**
1	Understand processor fundamentals.	4.1.1	**Explain** the Von Neumann model for a computer system.	AO1
		4.1.2	**Examine** the purpose and role of each register in the Von Neumann model.	AO1
		4.1.3	**Assess** the purpose and role of the components within the processor.	AO2
		4.1.4	**Determine** how the different ports enable connection to peripherals.	AO2
		4.1.5	**Outline** the stages of the Fetch-Execute cycle.	AO1
		4.1.6	**Explain** the purpose of interrupts.	AO1
		4.1.7	**Demonstrate** how interrupts are handled in the Fetch-Execute cycle.	AO3
		4.1.8	**Investigate** the relationship between assembly language and machine code.	AO2
		4.1.9	**Explain** the stages of the assembly process for a two-pass assembler.	AO1
		4.1.10	**Classify** assembly language instructions.	AO2

		4.1.11	**Outline** the different modes of addressing.	AO1
		4.1.12	**Examine** the impact of a shift on a binary number.	AO2
2	Be able to demonstrate the practical application of processor fundamentals.	4.2.1	**Execute** assembly language instructions to dry run a program.	AO3
		4.2.2	**Carry** out shifts on a binary number.	AO2
		4.2.3	**Implement** bit manipulation to check values in registers.	AO3

5: System Software

Aim:
The aim of this content is to empower learners to develop both a theoretical understanding and practical skills related to operating systems and language translators.

	The learner will:	SLO #	Assessment Criteria - The learner can:	Cognitive levels
1	Understanding the fundamentals of system software.	5.1.1	**Explain** why a computer system requires an Operating System.	AO1
		5.1.2	**Explain** the key management tasks carried out by the Operating System.	AO1
		5.1.3	**Support** the need for utility software.	AO2
		5.1.4	**Explore** the purpose and function of typical utility software.	AO2
		5.1.5	**Explain** the purpose of program libraries and the benefits of using a library (including DLL).	AO1
		5.1.6	**Outline** the purpose of an assembler, compiler, and interpreter.	AO1
		5.1.7	**Analyze** the benefits of using a compiler and/or interpreter in a given situation.	AO2
		5.1.8	**Identify** the features found in an IDE.	AO1
		5.1.9	**Clarify** how an OS can maximize the use of resources.	AO2
		5.1.10	**Demonstrate** the ways in which the user interface hides the complexities of the hardware from the user.	AO3

		5.1.11	**Outline** how processes are managed by the OS.	AO1
		5.1.12	**Explain** the use of virtual memory, paging, and segmentation for memory management.	AO1
		5.1.13	**Explore** how an interpreter can execute programs without producing a translated version.	AO2
		5.1.14	**Investigate** the various stages involved in the compilation of a program.	AO2
2	Be able to demonstrate the practical application of system software	5.2.1	**Apply** Backus-Naur Form (BNF) and syntax diagrams to express the grammar of a language.	AO3
		5.2.2	**Utilize** Reverse Polish Notation (RPN) to evaluate expressions.	AO3

6: Security, Privacy, and Data Integrity

Aim:
The aim of this content is to equip learners with a theoretical understanding of, and practical applications for, data security and data integrity.

	The learner will:	SLO #	Assessment Criteria - The learner can:	Cognitive levels
1	Understand the fundamentals of security, privacy, and data integrity	6.1.1	**Distinguish** between the security, integrity, and privacy of data.	AO1
		6.1.2	**Explain** the threats to data and computer systems.	AO1
		6.1.3	**Investigate** how threats can be prevented or restricted.	AO2
		6.1.4	**Assess** the methods to secure data.	AO3
		6.1.5	**Outline** different validation routines.	AO1
		6.1.6	**Explain** how verification can ensure data is identical to the original.	AO1
		6.1.7	**Support** how data can be verified during data entry and transfer.	AO3
		6.1.8	**Explore** the key terms associated with encryption.	AO1
		6.1.9	**Investigate** the use of encryption, including symmetric and asymmetric encryption.	AO3
		6.1.10	**Explain** the purpose and use of SSL and TLS.	AO1
		6.1.11	**Explain** how digital certificates are applied.	AO1

7: Ethics and Ownership

Aim:
The aim of this content is to equip learners with a theoretical understanding of, and practical applications for, copyright and artificial intelligence.

	The learner will:	SLO #	Assessment Criteria - The learner can:	Cognitive levels
1	Understand the applications of ethics and ownership	7.1.1	**Explain** the need for ethics and ethical behavior.	AO1
		7.1.2	Investigate the impact of acting ethically and unethically.	AO2
		7.1.3	**Identify** ways a person can act ethically or unethically in a given situation.	AO1
		7.1.4	**Outline** the key features of a range of software licenses.	AO1
		7.1.5	**Assess** the need for Artificial Intelligence (AI).	AO3
		7.1.6	**Evaluate** the benefits and drawbacks of AI.	AO3

8: Databases

Aim:
The aim of this content is to equip learners with the ability to perform both theoretical and practical analysis of CPU architecture, assembly language, and bit manipulation.

	The learner will:	SLO #	Assessment Criteria - The learner can:	Cognitive levels
1	Understand database concepts and database management systems	8.1.1	**Recognize** the limitations of a file-based approach.	AO1
		8.1.2	**Outline** the features of a relational database that address the limitations of a file-based approach.	AO1
		8.1.3	**Analyze** the normalization process of a database.	AO2
		8.1.4	**Explain** how a DBMS overcomes the limitations of a file-based approach.	AO1
		8.1.5	**Explore** the features and software tools of a DBMS.	AO2
2	Be able to demonstrate a practical application of databases	8.2.1	**Develop** entity-relationship (E-R) diagrams to document a database design.	AO3
		8.2.2	**Rebuild** a normalized database design based on a given database description.	AO3
		8.2.3	**Assist** with DDL and DML commands written in SQL.	AO3
		8.2.4	**Write** SQL scripts to perform DDL and DML tasks.	AO3

9: Computational Thinking, Algorithm Design, and Problem Solving

Aim:
The aim of this content is to equip candidates with both a theoretical understanding and practical knowledge of computational thinking skills and algorithms.

	The learner will:	SLO #	Assessment Criteria - The learner can:	Cognitive levels
1	Understand theoretical concepts of computational thinking, algorithm design and problem solving	9.1.1	**Explore** the purpose of and need for abstraction.	AO2
		9.1.2	**Investigate** the purpose of and need for decomposition.	AO2
		9.1.3	**Select** appropriate identifier names.	AO2
		9.1.4	**Explain** how stepwise refinement can be used to express an algorithm to a level of detail suitable for programming.	AO1
		9.1.5	**Analyze** a linear and binary search.	AO2
		9.1.6	**Analyze** an insertion sort and a bubble sort.	AO2
		9.1.7	**Explore** linked lists, stacks, queues, and binary trees.	AO2
		9.1.8	**Explain** the use of Big O notation to specify time and space complexity.	AO1
		9.1.9	**Evaluate** algorithms based on criteria such as time taken and memory used.	AO3

		9.1.10	**Identify** the essential features of recursion.	AO1
		9.1.11	**Contrast** the use of recursion and iteration.	AO2
		9.1.12	**Assess** what a compiler must do to translate recursive programming code.	AO3
2	Be able to demonstrate the practical application of computational thinking algorithm design and problem solving	9.2.1	**Create** an abstract model of a system.	AO3
		9.2.2	**Decompose** a problem into its sub-problems.	AO3
		9.2.3	**Construct** programs in pseudocode using input, process, and output.	AO3
		9.2.4	**Develop** pseudocode using assignment, sequence, selection, and repetition (including logic statements).	AO3
		9.2.5	**Derive** pseudocode from a structured English description and a flowchart.	AO3
		9.2.6	**Design** algorithms to implement a binary and linear search.	AO3
		9.2.7	**Develop** algorithms to implement insertion and bubble sorts.	AO3
		9.2.8	**Develop** algorithms to locate items in a linked list and a binary tree.	AO3
		9.2.9	**Develop** algorithms to insert items into a stack, queue, linked list, and binary tree.	AO3
		9.2.10	**Develop** algorithms to delete an item from a stack, a queue, and a linked list.	AO3
		9.2.11	**Investigate** how an ADT can be implemented using a built-in data type and another ADT, and create algorithms to implement this.	AO3
		9.2.12	**Design** and trace recursive algorithms.	AO3

10: Data Types and Structures

Aim:
The purpose of this content is to help learners develop both theoretical understanding and practical skills in data types, records, arrays, files, and abstract data types (ADTs).

	The learner will:	**SLO #**	**Assessment Criteria - The learner can:**	**Cognitive levels**
1	Understand the concepts of data types, records, arrays, files, and abstract data types	10.1.1	**Apply** and use appropriate data types for a problem solution.	AO3
		10.1.2	**Identify** a suitable data structure (1D or 2D array) to use for a given task.	AO2
		10.1.3	**Justify** why files are needed.	AO1
		10.1.4	**Show** how a queue, stack and linked list can be implemented using arrays.	AO3
		10.1.5	**Evaluate** how a stack, queue and linked list are examples of ADTs.	AO3
		10.1.6	**Explore** that an ADT is a collection of data and a set of operations on those data.	AO2
2	Be able to demonstrate the practical knowledge of data types and structures	10.2.1	**Apply** a record structure to hold a set of different data types under one identifier.	AO3
		10.2.2	**Use** the technical terms associated with arrays.	AO1
		10.2.3	**Develop** pseudocode for 1D and 2D arrays.	AO3
		10.2.4	**Construct** pseudocode to process array data.	AO3

		10.2.5	**Write** pseudocode to handle text files that consist of one or more lines.	AO3
		10.2.6	**Implement** a stack, queue and linked list to store data.	AO3

11: Programming

Aim:
The purpose of this content is to equip learners with both theoretical understanding and practical competence in programming and structured programming.

	The learner will:	SLO #	Assessment Criteria - The learner can:	Cognitive levels
1	Understand the concepts of programming	11.1.1	**Justify** the purpose of the one loop structure when solving problems.	AO2
		11.1.2	**Evaluate** the terminologies associated with procedures and functions.	AO2
		11.1.3	**Describe** what is meant by a programming paradigm.	AO1
		11.1.4	**Investigate** the terminology associated with OOP such as attributes, objects, methods.	AO2
		11.1.5	**Assess** the importance of exception handling.	AO2
		11.1.6	**Illustrate** when to consider the constructor of an algorithm in terms of its appropriateness.	AO2
2	Be able to demonstrate the practical application of programming	11.2.1	**Apply** a section of code that is repeated multiple times.	AO3
		11.2.2	**Construct** pseudocode from a given design presented as either a program flowchart or structured English.	AO3
		11.2.3	**Create** pseudocode statements for: ▪ the declaration of variables and constants ▪ the assignment of values to variables and constants ▪ expressions involving any of the arithmetic or logical operators, input from the keyboard, and output to the console.	AO3

		11.2.4	**Apply** pseudocode to produce: ▪ an IF structure including ELSE and nested IF statements ▪ a CASE statement ▪ a count-controlled loop ▪ a post-condition loop ▪ a pre-condition loop	AO3
		11.2.5	**Implement** parameters in a procedure and a function.	AO3
		11.2.6	**Develop** efficient pseudocode.	AO3
		11.2.7	**Construct** low-level code that uses various addressing modes.	AO3
		11.2.8	**Produce** imperative programming code that uses constructs, procedures and functions.	AO3
		11.2.9	**Create** low-level code that uses various addressing modes.	AO3
		11.2.10	**Develop** imperative programming code that uses constructs, procedures and functions.	AO3
		11.2.11	**Design** program code to solve problems by creating appropriate classes and making use of OOP techniques.	AO3
		11.2.12	**Modify** and construct program code to solve problems by writing appropriate facts and rules.	AO3
		11.2.13	**Implement** code to perform file-processing operations.	AO3
		11.2.14	**Apply** program code to use exception handling.	AO3

12: Software Development
Aim: *The purpose of this content is to provide learners with theoretical knowledge and practical experience in the software development lifecycle, including program design, testing, and maintenance.*

	The learner will:	SLO #	Assessment Criteria - The learner can:	Cognitive levels
1	Understand the program development lifecycle	12.1.1	**Evaluate** the purpose of a development life cycle.	AO3
		12.1.2	**Assess** the need for different development life cycles depending on the program being developed.	AO3
		12.1.3	**Review** the principles, benefits and drawbacks of each type of life cycle.	AO3
		12.1.4	**Outline** the analysis, design, coding, testing and maintenance stages in the program development life cycle.	AO1
		12.1.5	**Discuss** how faults in programs can be exposed and avoided.	AO2
		12.1.6	**Justify** the need for continuing maintenance of a system and the differences between each type of maintenance.	AO3
		12.1.7	**Examine** an existing program and make amendments to enhance functionality.	AO3
2	Be able to demonstrate the practical application of software development	12.2.1	**Apply** a structure chart to decompose a problem into sub-tasks and express the parameters passed between the various modules, procedures or functions which are part of the algorithm design.	AO3

		12.2.2	**Develop** a state-transition diagram to document an algorithm.	AO3
		12.2.3	**Construct** a state-transition diagram to document an algorithm.	AO1
		12.2.4	**Identify** the different types of errors.	AO3
		12.2.5	**Correct** identified errors.	AO3
		12.2.6	**Implement** different methods of testing and appropriate data for each method.	AO3
		12.2.7	**Select** appropriate data for a test plan.	AO2
		12.2.8	**Explore** the need for a test strategy and test plan, and their likely contents.	AO2

13: Artificial Intelligence

Aim:
The purpose of this content is to help learners gain theoretical knowledge and practical experience in artificial intelligence, focusing on graphs and their applications.

	The learner will:	**SLO #**	**Assessment Criteria - The learner can:**	**Cognitive levels**
1	Understand artificial intelligence graphs and applications	13.1.1	**Examine** how graphs can be used to aid Artificial Intelligence.	AO1
		13.1.2	**Evaluate** how artificial neural networks help with machine learning.	AO2
		13.1.3	**Review** the use of Deep Learning, Machine Learning and Reinforcement Learning and the reasons for using these methods.	AO3
		13.1.4	**Justify** the reasons for using Deep Learning, Machine Learning and Reinforcement Learning.	AO3
		13.1.5	**Assess** back propagation and regression methods in machine learning.	AO3
2	Be able to demonstrate the practical application of Artificial Intelligence	13.2.1	**Apply** A* and Dijkstra’s algorithms to perform searches on a graph.	AO2
		13.2.2	**Develop** a game using sequence/selection/loops using variables/constants/maths symbols/input/output.	AO3

Use of Calculators

- Calculators are **not permitted** in any examination paper.

Programming Languages

ZUEB accepts solutions written in the following programming languages:

- Python
- C family of languages (e.g., C, C++, C#)
- Java
- Visual Basic
- PHP
- Delphi