Lanterna Education

IBDP COMPUTER SCIENCE

COURSE OUTLINE

INTRODUCTION

The objectives for all group 4 subjects reflect those parts of the aims that will be assessed. Wherever appropriate, the assessment will draw upon a range of contexts and identify the social, moral and economic effects of science and technology.

In the Course outline I have taken the support from the IBDP Subject guide and set the course content weekly to complete the course in a period of two year (240 hours).

SYLLABUS OUTLINE (HIGHER LEVEL):

Syllabus content	Recommended Teaching Hours
CORE The topics that must be studied, including some practical work, are: • Topic 1: System fundamentals • Topic 2: Computer organization • Topic 3: Networks • Topic 4: Computational thinking, problem-solving and programming	80
HL EXTENSION The topics that must be studied, including some practical work, are: • Topic 5: Abstract data structures • Topic 6: Resource management • Topic 7: Control	45
CASE STUDY Additional subject content introduced by the annually issued case study	30

Syllabus content	Recommended Teaching Hours
OPTION • Students study one of the following options: • Option A: Databases • Option B: Modelling and simulation • Option C: Web science • Option D: Object-oriented programming (OOP)	45
INTERNAL ASSESSMENT Practical application of skills through the development of a product and associated documentation	30
GROUP 4 PROJECT	10
TOTAL TEACHING HOURS	240

PAPER 1 APPROACHES

The purpose: to assess the student's ability to demonstrate the following objectives in relation to the syllabus:

Assessment objective 1: know and understand

Assessment objective 2: apply and use

Assessment objective 3: construct, analyze and evaluate

Section A (30 minutes approximately) consists of several compulsory short answer questions testing mainly objectives 1 and 2. The maximum mark for this section is 25. Some of the questions are common to this paper and HL paper 1, section

Section B (60 minutes approximately) consists of three compulsory structured questions that may be subdivided. The maximum mark for this section is 45. Some questions may be common to this paper and HL paper 1, section B.

The number of marks for each part will be given on the paper and is linked to the command term used. This will indicate to students the depth of the response required.

Approach	Level	Weeks	Hours
Topic 1.1: Systems in organisations	SAQ and ERQ only	4	10
Topic 1.2: System design basics	SAQ and ERQ only	4	10
Topic 2: Computer Organisation	SAQ and ERQ only	3	06
Topic 3: Networks	SAQ and ERQ only	4	10

PAPER 1 APPROACHES

Approach	Level	Weeks	Hours
Topic 4: Computational thinking, problem-solving and programming	SAQ and ERQ only	15	45
Topic 5: Abstract Data Structures	Mainly ERQ	18	23
Topic 6: Resource Management	SAQ and ERQ only	3	08
Topic 7: Control	SAQ and ERQ only	5	14

PAPER 2 OPTIONS

(HL STUDENTS STUDY 2; SL STUDENTS STUDY 1).

The purpose: assess the student's ability to demonstrate the following objectives in relation to the option chosen:

Assessment objective 1: know and understand

Assessment objective 2: apply and use

Assessment objective 3: construct, analyze and evaluate

Section A consists of common questions (to the SL/HL core) for the option chosen. The maximum mark is 45.

Section B consists of questions for the option chosen relating to the HL extension. The maximum mark is 20.

Students are not expected to construct code in the following options:

- databases
- modelling and simulation
- web science.

Students will be expected to interpret and/or construct code in Java in the OOP option.

Option	Level	Weeks	Hours
Database	SAQ and ERQ both	15	45
Modelling and Simulation	SAQ and ERQ both		
Web Science	SAQ and ERQ only		
Object-oriented programming	SAQ and ERQ only		

SUMMARY

The course can be modified to suit students' preferences such as selection of favourite topic for the paper 2 or more than one HLE can be taught giving the students more choice. I have provided support to many students across the globe, especially in the internal assessment. Being a part of IB moderation teams, best practice has been applied to give the best understanding and learning experience to the students.

I would be happy to discuss these modifications with you should you need to. Or, if any school wants to see a course with all the content put back in, then I can also remove the modifications.

To be clear: The modifications presented here are only slight and benefit student learning. If we put them back, the course becomes rushed with less time on each unit. However, there are areas of discussion whereby the course can be modified to suit specific needs and interests.

Best wishes

Praveen Kumar Sharma (M.Tech., B.Ed.)

June 2023

YEAR 1:

APPROACHES AND IA

Week: Number of weeks out of 33 for Year 1

Content: Course content copied and pasted directly from the Guide.

Aims: Suggested lesson aims for the week (3/4 per week).

Abbreviations used: Theory of knowledge (TOK), Middle Years Programme (MYP), social/ethical issues (S/E) and the international dimension (INT).

Assessment: Students do not need to be assessed every lesson. Formative assessment = practice in extended time frames (e.g. 30 minutes for an SAQ rather than 20 minutes); Summative assessment = IB Grading with IB Exam expectations (e.g. writing an SAQ in 20 minutes and graded with IB Rubrics).

Week	Content	Aims	Assessment
1	Introduction to Computer Science and Assessment	What is computer Science? What is IBDP CS? How will I be assessed? Exam technique: How to write an SAQ; How to write an ERQ.	N/A

YEAR 1:

APPROACHES AND IA

Topic 1.1—Systems in organizations (10 hours)			
2	• Identify the context for which a new system is planned. • Describe the need for change management. • Outline compatibility issues resulting from situations including legacy systems or business mergers. • Compare the implementation of systems using a client's hardware with hosting systems remotely. • Evaluate alternative installation processes. • Discuss problems that may arise as a part of data migration.	• The extent and limitations of a new system should be appreciated. Organizational issues related to the installation of new systems such as user roles, underlying technologies. • Students should understand there are a number of factors that need to be managed to ensure change is successful. S/E The way that change is managed can have significant effects on employers and employees. • INT, S/E When organizations interact, particularly on an international basis, there may be issues of software compatibility and language differences. • The benefits and drawbacks of SaaS (Software-as-a-Service) should be considered. S/E, INT The remote host may be in a different time zone and this can have significant effects on end-users. • Students should be aware of the methods of implementation/ conversion. Parallel running, pilot running, direct changeover and phased conversion. • S/E Training issues may require organizations to restructure their workforce.	N/A

Topic 1.1—Systems in organizations (10 hours)			
3	• Discuss problems that may arise as a part of data migration. • Suggest various types of testing. • Describe the importance of user documentation. • Evaluate different methods of providing user documentation. • Evaluate different methods of delivering user training.	INT These include incompatible file formats, data structures, validation rules, incomplete data transfer and international conventions on dates, currencies and character sets. • Types of testing can include: user acceptance testing, debugging, beta testing. Students should be aware that there are programs that can test other programs, thereby automating parts of the testing process and reducing costs. • S/E The quality of user documentation can affect the rate of implementation of the new system. • The quality of user documentation can affect the rate of implementation of the new system. Examples should include methods such as: help files, online support and printed manuals. • Examples should include self-instruction, formal classes, remote/ online training. S/E The quality of the delivery of user training	Formative SAQ assessment (30 -40 minutes)
4	Identify a range of causes of data loss. • Outline the consequences of data loss in a specified situation. • Describe a range of methods that can be used to prevent data loss. • Describe strategies for managing releases and updates.	Causes include malicious activities and natural disasters. • S/E Malicious activity may be a result of activities by employees within the organization or intruders. S/E Loss of medical records, cancellation of a hotel reservation without the knowledge of the traveller. • These should include failover systems, redundancy, removable media, offsite/online storage. • Students should be aware of a variety of ways in which updates and patches are made available and deployed. This includes automatic updates received on a regular basis online. S/E, INT Performance issues related to the inability to install updates may hinder end-users and reduce compatibility between systems in geographically diverse locations.	Summative ERQ and SAQ (50 minutes)

Topic 1.2— System design basics (10 hours)			
5	• Define the terms: hardware, software, peripheral, network, human resources. • Describe the roles that a computer can take in a networked world. • Discuss the social and ethical issues associated with a networked world. • Identify the relevant stakeholders when planning a new system. • Describe methods of obtaining requirements from stakeholders. • Describe appropriate techniques for gathering the information needed to arrive at a workable solution.	• Roles include client, server, email server, DNS server, router and firewall. • Develop an appreciation of the social and ethical issues associated with continued developments in computer systems. • Who is a relevant stakeholder? The role of the end-user must be considered when planning a new system. • Including surveys, interviews, direct observations. The need for effective collaboration to obtain appropriate information from stakeholders. • Examining current systems, competing products, organizational capabilities, literature searches.	N/A
6	• Construct suitable representations to illustrate system requirements. • Describe the purpose of prototypes to demonstrate the proposed system to the client. • Discuss the importance of iteration during the design process. • Explain the possible consequences of failing to involve the end-user in the design process. • Discuss the social and ethical issues associated with the introduction of new IT systems.	• Examples include: system flow charts, data flow diagrams, structure chart. • The need to effectively collaborate to gather appropriate information to resolve complex problems. To develop logical and critical thinking to develop proposed systems. • MYP Design cycle. • The failure to involve the end- user may lead to software that is not suitable for its intended use, which may have adverse effects on user productivity. • Develop an appreciation of the social and ethical issues associated with continued developments in specified computer systems.	SAQ Practice (Formative)

7	• Define the term usability. • Identify a range of usability problems with commonly used digital devices. • Identify methods that can be used to improve the accessibility of systems. • Identify a range of usability problems that can occur in a system. • Discuss the moral, ethical, social, economic and environmental implications of the interaction between humans and machines.	• S/E This includes ergonomics and accessibility. • S/E Students should be aware of usability issues in a range of devices including PCs, digital cameras, cell phones, games consoles, MP3 players and other commonly used digital devices. • S/E Examples include touch screen, voice recognition, text-to-speech, Braille keyboard. • S/E These should be related to the systems. Systems include ticketing, online payroll, scheduling, voice recognition, systems that provide feedback. • AIM 8 Raise awareness of the moral, ethical, social, economic and environmental implications of using science and technology.	ERQ Practice (Formative)
Topic 2.1 Computer organization (6 hours) (10 hours)			
8	• Outline the architecture of the central processing unit (CPU) and the functions of the arithmetic logic unit • (ALU) and the control unit (CU) and the registers within the CPU. • Describe primary memory. • Explain the use of cache memory. • Explain the machine instruction cycle. • Identify the need for persistent storage.	• Students should be able to reproduce a block diagram showing the relationship between the elements of the CPU, input and output and storage. The memory address register (MAR) and memory data register (MDR) are the only ones that need to be included. • Distinguish between random access memory (RAM) and read-only memory (ROM), and their use in primary memory. • Students should be able to explain the effect of cache memory in speeding up the system as well as being able to explain how it is used. • This should include the role of data bus and address bus. • Persistent storage is needed to store data in a non-volatile device during and after the running of a program. Consequences of data loss.	N/A

9	• Describe the main functions of an operating system. • Outline the use of a range of application software. • Identify common features of applications. • Define the terms: bit, byte, binary,denary/decimal, hexadecimal. • Outline the way in which data is represented in the computer. • Define the Boolean operators: AND, OR, NOT, NAND, NOR and XOR. • Construct truth tables using the above operators. • Construct a logic diagram using AND, OR, NOT, NAND, NOR and XOR gates.	• This is confined to a single-user operating system. Technical details are not needed. For example, memory management should be described but how this is handled in a multitasking environment is not expected. • Application software should include word processors, spreadsheets, database management systems, email, web browsers, computer-aided design (CAD) and graphic processing software. • Including toolbars, menus, dialogue boxes, graphical user interface (GUI) components. Students should understand that some features are provided by the application software and some by the operating system. • To include strings, integers, characters and colours. This should include considering the space taken by data, for instance the relation between the hexadecimal representation of colours and the number of colours available. • LINK Introduction to programming, approved notation sheet. • For example, Maria won't go to school if it is cold and raining or she has not done her homework. Not more than three inputs are used. LINK Thinking logically. • Problems will be limited to an output dependent on no more than three inputs. The gate should be written as a circle with the name of the gate inside it. For example: • OR • LINK Thinking logically, connecting computational thinking and program design, introduction to programming.	SAQ Practice (Formative)

3.1 Networks (9 hours)

10-12	• Identify different types of networks. • Outline the importance of standards in the construction of networks. • Describe how communication over networks is broken down into different layers. • Identify the technologies required to provide a VPN. • Evaluate the use of a VPN. • Define the terms: protocol, data packet. • Explain why protocols are necessary. • Explain why the speed of data transmission across a network can vary. • Explain why compression of data is often necessary when transmitting across a network. • Outline the characteristics of different transmission media. • Explain how data is transmitted by packet switching. • Outline the advantages and disadvantages of wireless networks. • Describe the hardware and software components of a wireless network. • Describe the characteristics of wireless networks. • Describe the different methods of network security. • Evaluate the advantages and disadvantages of each method of network security.	Examples include local area network (LAN), virtual local area network (VLAN), wide area network (WAN), storage area network (SAN), wireless local area network (WLAN), internet, extranet, virtual private network (VPN), personal area network (PAN), peer-to-peer (P2P). • S/E, INT Globalization has been accelerated by the technical advances linked to network development. • INT Standards enable compatibility through a common “language” internationally. • Awareness of the OSI seven layer model is required, but an understanding of the functioning of each layer is not. • S/E use of a VPN has led to changes in working patterns. • Including data integrity, flow control, deadlock, congestion, error checking. • S/E, INT Compression has enabled information to be disseminated more rapidly. • Characteristics include: speed, reliability, cost and security. Transmission media include: metal conductor, fibre optic, wireless. • S/E wireless networks have led to changes in working patterns, social activities and raised health issues. • Include: WiFi; Worldwide Interoperability for Microwave Access (WiMAX); 3G mobile; future networks. S/E, INT Connectivity between different locations. • Include encryption types, userID, trusted media access control (MAC) addresses. S/E Wireless networks have led to concerns about the security of the user’s data.	ERQ and SAQ Practice (Formative)

Topic 4—Computational thinking, problem-solving and programming (45 hours) 4.1 General principles (10 hours)			
13-15	• Identify the procedure appropriate to solving a problem. • Evaluate whether the order in which activities are undertaken will result in the required outcome. • Explain the role of sub-procedures in solving a problem. • Identify when decision-making is required in a specified situation. • Identify the decisions required for the solution to a specified problem. • Identify the condition associated with a given decision in a specified problem. • Explain the relationship between the decisions and conditions of a system. • Deduce logical rules for real-world situations. • Identify the inputs and outputs required in a solution. • Identify pre-planning in a suggested problem and solution. • Explain the need for pre-conditions when executing an algorithm. • Outline the pre- and post-conditions to a specified problem. • Identify exceptions that need to be considered in a specified problem solution. • Identify the parts of a solution that could be implemented concurrently.	• This includes identifying the steps and putting them in the correct order. Such as recipes, block-arrow-block- arrow. Connecting computational thinking and program design, introduction to programming. • Links to problems presented to the student in other areas of the syllabus. Thinking ahead, thinking concurrently. Connecting computational thinking and program design, introduction to programming. MYP Technology, step-by-step instructions. • Constructing procedures that can then be referred to by their identifier. Abstraction, connecting computational thinking and program design, introduction to programming. • Links to procedural thinking— alternative procedures. LINK Connecting computational thinking and program design, introduction to programming. • Different actions are taken based on conditions. LINK Connecting computational thinking and program design, introduction to programming. Applying thinking skills to identify and resolve a specified complex problem. • Testing conditions, iteration. Identifying and constructing the conditions—AND, OR, NOT relationships—Boolean tests. LINK Connecting computational thinking and program design, introduction to programming. • IF … THEN … ELSE LINK Connecting computational thinking and program design, introduction to programming. • LINK Connecting computational thinking and program design, introduction to programming.	One SAQ and One ERQ Summative Assessment: 90 minutes.

13-15	• Describe how concurrent processing can be used to solve a problem. • Evaluate the decision to use concurrent processing in solving a problem. • Identify examples of abstraction. • Explain why abstraction is required in the derivation of computational solutions for a specified situation. • Construct an abstraction from a specified situation. • Distinguish between a real-world entity and its abstraction.	• Gantt charts. Pre-ordering. Pre-heating an oven. Home/locker/ knapsack. Caching/pre-fetching. Building libraries of pre-formed elements for future use. LINK Thinking procedurally, thinking concurrently. Connecting computational thinking and program design, introduction to programming • For example, cooking a dish for a meal. All ingredients available before starting to cook. A place to eat the food. • For example, identify the pre- conditions for calculating the end-of-year bonus when not all employees have worked for the company for the whole year. LINK Connecting computational thinking and program design, introduction to programming. • Could include computer systems or real-life situations. LINK Thinking ahead, thinking procedurally. Connecting computational thinking and program design, introduction to programming. • For example, building a house, production lines, division of labour. Students will not be expected to construct a flow chart or pseudocode related to concurrent processing. • LINK Thinking ahead, thinking procedurally. Connecting computational thinking and program design, introduction to programming. • Selecting the pieces of information that are relevant to solving the problem. LINK Thinking ahead. • Students should be aware of the concept of objects, for example, the use of collections as objects in the design of algorithms. LINK • Databases: tables, queries • Modelling and simulation: an abstraction of reality • OOP: classes, sub-classes • Web science: distributed applications • There is no need to use code. Levels of abstraction through successive decomposition. A school can be decomposed into faculties. A faculty can be decomposed into departments. LINK Thinking ahead, thinking procedurally. Connecting computational thinking and program design, introduction to programming. • TOK The map as an abstraction of the territory.	One SAQ and One ERQ Summative Assessment: 90 minutes.

4.2 Connecting computational thinking and program design (22 hours)			
16	Describe the characteristics of standard algorithms on linear arrays.	These are: sequential search, binary search, bubble sort, selection sort.	N/A
17	• Outline the standard operations of collections. • Discuss an algorithm to solve a specific problem.	• Demonstrate thinking skills to represent a possible solution to a specified complex problem. Using flow charts to solve problems in real-life contexts, patterns and sequences, logic, algorithms. • Suitable algorithms may include both standard algorithms and novel algorithms. Suitable may include considerations of efficiency, correctness, reliability, and flexibility. Students are expected to suggest algorithms that will actually solve the problem successfully. LINK General principles of computational thinking, introduction to programming. • Students should understand and explain the difference in efficiency between a single loop, nested loops, a loop that ends when a condition is met or questions of similar complexity. • Students should also be able to suggest changes in an algorithm that would improve efficiency, for example, using a flag to stop a search immediately when an item is found, rather than continuing the search through the entire list. • Examination questions will involve specific algorithms (in pseudocode/ flow charts), and students may be expected to give an actual number (or range of numbers) of iterations that a step will execute.	N/A

18	• Analyse an algorithm presented as a flow chart. • Analyse an algorithm presented as pseudocode.	• Examination questions may involve variables, calculations, simple and nested loops, simple conditionals and multiple or nested conditionals. This would include tracing an algorithm as well as assessing its correctness. Students will not be expected to construct a flow chart to represent an algorithm in an externally assessed component. • Examination questions may involve variables, calculations, simple and nested loops, simple conditionals and multiple or nested conditionals. • This would include tracing an algorithm as well as assessing its correctness.	N/A
19-22	• Construct pseudocode to represent an algorithm. • Suggest suitable algorithms to solve a specific problem. • Deduce the efficiency of an algorithm in the context of its use. • Determine the number of times a step in an algorithm will be performed for given input data.	• MYP Mathematics: using flow charts to solve problems in real-life contexts, patterns and sequences, logic, algorithms. MYP Technology: design cycle (inputs, processes, outputs, feedback, iteration). AIM 4 Demonstrate thinking skills to represent a possible solution to a specified complex problem. • Suitable algorithms may include both standard algorithms and novel algorithms. Suitable may include considerations of efficiency, correctness, reliability and flexibility. Students are expected to suggest algorithms that will actually solve the problem successfully. LINK General principles of computational thinking, introduction to programming. • Students should understand and explain the difference in efficiency between a single loop, nested loops, a loop that ends when a condition is met or questions of similar complexity. • Students should also be able to suggest changes in an algorithm that would improve efficiency, for example, using a flag to stop a • search immediately when an item is found, rather than continuing the search through the entire list. • Examination questions will involve specific algorithms (in pseudocode/ flow charts), and students may be expected to give an actual number (or range of numbers) of iterations that a step will execute.	N/A

4.2 Introduction to programming (13 hours)			
23	• State the fundamental operations of a computer. • Distinguish between fundamental and compound operations of a computer. • Explain the essential features of a computer language.	• These include: add, compare, retrieve and store data. Complex capabilities are composed of very large numbers of very simple operations. • For example, “find the largest” is a compound operation. • For example, fixed vocabulary, unambiguous meaning, consistent grammar and syntax.	N/A
24	• Explain the essential features of a computer language. • Explain the need for higher level languages. • Outline the need for a translation process from a higher level language to machine executable code.	• For example, fixed vocabulary, unambiguous meaning, consistent grammar and syntax. • For example, as the human needs for computer systems have expanded it is necessary to abstract from the basic operations of the computer. It would take far too long to write the type of systems needed today in machine code. • For example, compiler, interpreter, virtual machine.	N/A
25	Define the terms: variable, constant, operator, object. • Define the operators =, ≠, <, <=, >, >=, mod, div. • Analyse the use of variables, constants and operators in algorithms. • Construct algorithms using loops, branching.	• LINK Approved notation sheet. • For example, identify and justify the use of a constant as opposed to a variable in a given situation. **MYP Mathematics:** forms of numbers, algebra—patterns and sequences, logic, algorithms. • Teachers must ensure algorithms use the symbols from the approved notation sheet. **LINK** Approved notation sheet. **MYP Mathematics:** using flow charts to solve problems in real-life contexts, logic, algorithms. **MYP Technology:** design cycle (inputs, processes, outputs, feedback, iteration). **LINK** Connecting computational thinking and program design.	N/A

26	• Describe the characteristics and applications of a collection. • Construct algorithms using the access methods of a collection. • Discuss the need for sub-programmes and collections within programmed solutions. • Construct algorithms using pre- defined sub-programmes, one- dimensional arrays and/or collections.	• **LINK** General principles of computational thinking, connecting computational thinking and program design. • **LINK** Connecting computational thinking and program design. • Show an understanding of the usefulness of reusable code and program organization for the individual programmer, team members and future maintenance. • **LINK** General principles of computational thinking, connecting computational thinking and program design. • Students will only be required to analyse flow charts in the externally assessed components. • Students will be expected to write and analyse pseudocode in the externally assessed components. • **S/E, AIM 8** Appreciate the implications of using available code from sources such as online forums.	N/A
Topic 5—Abstract data structures (23 hours) **5.1 Abstract data structures (23 hours)**			
27-28	• Identify a situation that requires the use of recursive thinking. • Identify recursive thinking in a specified problem solution. • Trace a recursive algorithm to express a solution to a problem. • Describe the characteristics of a twodimensional array. • Construct algorithms using twodimensional arrays. • Describe the characteristics and applications of a stack.	• Suggested practical activity: snowflakes and fractals, towers of Hanoi. • **LINK** Binary trees. • Students will be required to state the output of the recursive algorithm. For example, trees. **LINK** Binary trees. • **LINK** One-dimensional arrays and basic algorithms. • **LINK** Pseudocode information. • Characteristics: • Last in, first out (LIFO). Examples of the applications of stacks may include running recursive processes, return memory addresses. **LINK** Recursive thinking; connecting computational thinking and program design	Practicing recursive thinking and binary tree in P1 format (formative)

		Access methods: • push • pop • isEmpty. **LINK** Connecting computational thinking and program design.	N/A

Internal Assessment Introduction

Internal assessment is an integral part of the course and is compulsory for both SL and HL students. It enables students to demonstrate the application of their skills and knowledge, and to pursue their personal interests, without the time limitations and other constraints that are associated with written examinations. The internal assessment should, as far as possible, be woven into normal classroom teaching and not be a separate activity conducted after a course has been taught. The internal assessment requirements at SL and at HL are the same.

However, these requirements contribute to a different percentage of the overall mark. Students are required to produce a solution that consists of a cover page, the product and the documentation. The focus of the solution is on providing either an original product or additional functionality to an existing product for a client. The internal assessment component (solution), as well as being practical and productive, forms an important part of the assessment of the computer science course. It is imperative, therefore, that the teacher provides appropriate guidance to students.

30	Criterion A: Planning	• The following key questions should be considered. Who is the client/adviser? Is the choice of client/adviser appropriate? Why is the product being developed? • Rationale for proposed solution • Success Criteria	Selection of the client.

31	• Criterion B: Solution overview • Criterion C: Development	• Constructing RoT (Record of task). • Constructing Design Overview • The product must be compatible with the information in criterion A and criterion B. The student must present a list of the techniques used in developing the product. T**he techniques** may include algorithmic thinking, data structures, software tools and user interface. This list need not be exhaustive but should illustrate how the major components of the product were developed. **The student** must provide evidence of algorithmic thinking	Samples will be discussed.
32	• Criterion D: Functionality and extensibility of product • Criterion E: Evaluation	• **Functionality of the product** The student must use the video to demonstrate the product functioning. This evidence will be supported, where possible, by the product on the CD-ROM/DVD or USB. • **Extensibility of product** The student should design the product so that it can be maintained by another party and/or be further developed. • **Evaluation of the product** The evaluation of the product should refer directly to the success criteria in criterion A, feedback from the client/ adviser, as well as any other appropriate feedback obtained. • **Recommendations for the future development of the product** The student will use the feedback and the evaluation of the specific performance criteria to recommend possible future developments to the product.	Samples will be discussed.
33	IA Mopping Up	**To mop any areas that need it.**	N/A.

HAVE ANY QUESTIONS?

Contact us in any convenient way

UK COMPANY
Cottesbrooke Park,
Heartlands,
Daventry, NN11 8YL

SWEDISH COMPANY
Linnégatan 87D,
115 23 Stockholm, Sweden

CONTACT US
+44 20 4571 0455
lanterna.com
info@lanternaeducation.com