An Roinn Oideachais
agus Scileanna

Computer Science

Curriculum Specification

LEAVING CERTIFICATE
Ordinary and Higher Level

UPDATED 2023

Contents

Introduction 2

Senior cycle 3

The experience of senior cycle 4

Computer Science 6

Rationale 6

Aim 6

Objectives 6

Related learning 7

Early childhood 7

Primary school 7

Junior cycle 8

Senior cycle 8

Further study 8

Society and community 9

Education for sustainable development 9

Structure of Leaving Certificate Computer Science 10

Strand 1: Practices and principles 10

Strand 2: Core concepts 10

Strand 3: Computer science in practice 10

Key skills of senior cycle 12

Information processing 13

Critical and creative thinking 13

Communicating 13

Working with others 14

Being personally effective 14

Teaching and learning 15

Applied learning tasks 15

Differentiation 16

Differentiation through the learning outcomes of the specification 16

Differentiation in teaching and learning 17

Differentiation in assessment 17

Time allocation 17

Strands and learning outcomes 18

Assessment 24

Assessment for certification 24

Structure of assessment for certification 25

End-of-course examination 25

End-of-course assessment criteria 26

Coursework assessment 27

Coursework assessment criteria 27

Assessment programming language 28

Reasonable accommodations 28

Appendices 29

Appendix A: Glossary of action verbs used 29

Appendix B: Glossary of core concepts 32

1 Introduction

Computer science is the study of computers and algorithmic processes. Leaving Certificate Computer Science includes how programming and computational thinking can be applied to the solution of problems, and how computing technology impacts the world around us.

The specification is constructed into 3 strands, whose learning outcomes are interwoven. The 3 strands are:

1. Practices and principles
2. Core concepts
3. Computer science in practice

Students will learn:

- The practices and principles of computer science, such as computational thinking, computers and society, and creative design
- How to analyse problems in computational terms and understand concepts such as abstraction, logic, algorithms, computer systems, data representation and evaluation
- Programming languages and how to read, write, test and modify computer programs
- The process of designing computational artefacts such as web pages, digital animations, simulations, games, apps and robotic systems
- The ethical, historical, environmental and technological aspects of computer science, and how it impacts the social and economic development of society.

The role of programming in computer science is like that of practical work in the other subjects—it provides motivation, and a context within which ideas are brought to life. Students learn programming by solving problems through computational thinking processes and through practical applications such as applied learning tasks.

The Leaving Certificate Computer Science specification is designed for all students. It applies to many aspects of students' lives and is therefore relevant to a wide range of student interests. It is situated within the context of senior cycle education.

2

Senior cycle

The objectives of Leaving Certificate Computer Science are well aligned with the vision and principles of senior cycle education. Learners in senior cycle are approaching the end of their time in school and are focusing on the directions they would like to take in their future lives. Senior cycle plays a vital role in helping learners to address their current needs as resourceful, confident, engaged and active young adults and in preparing them for life in a changing economic and social context (Figure 1).

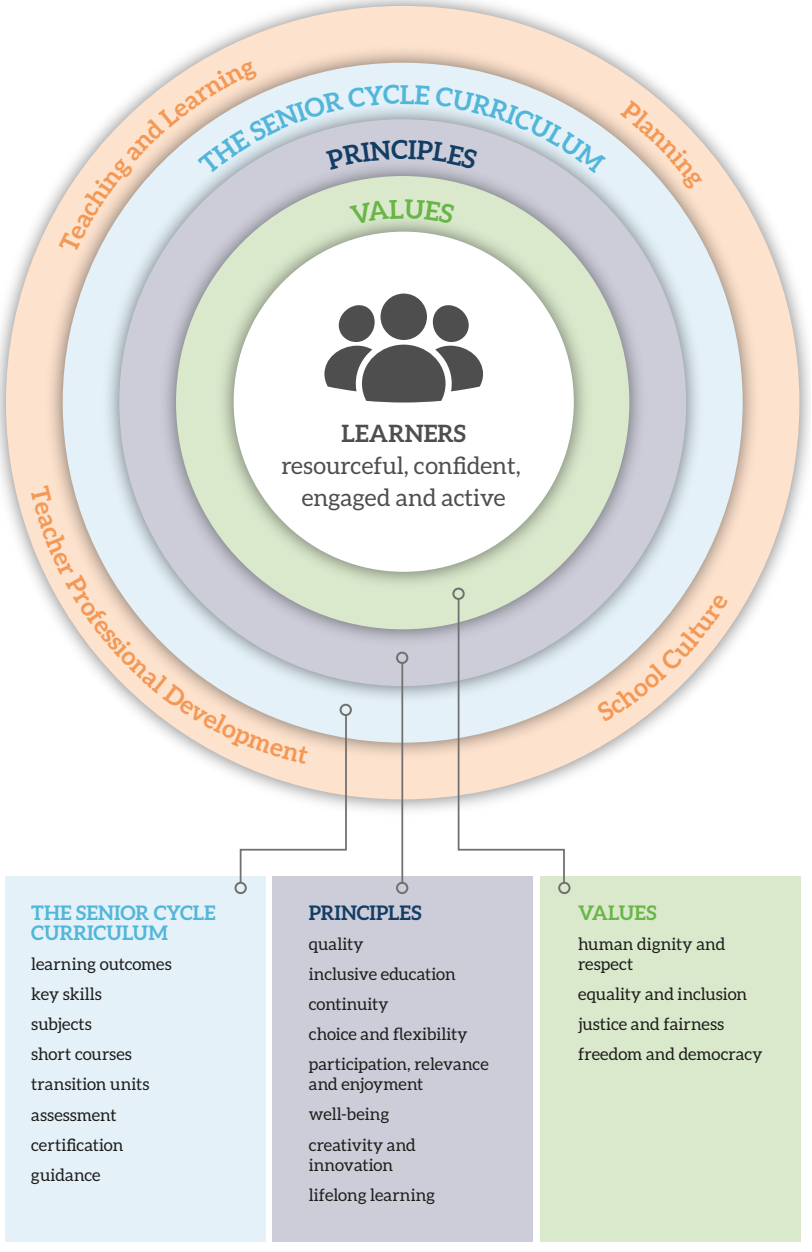

Figure 1: Overview of senior cycle

Senior cycle is founded on a commitment to educational achievement of the highest standard for all students, commensurate with their individual abilities. To support students as they shape their own future, there is an emphasis on the development of knowledge and deep understanding; on students taking responsibility for their own learning; on the acquisition of key skills; and on the processes of learning. The broad curriculum, with some opportunities for specialisation, supports continuity from junior cycle and sets out to meet the needs of students, some of whom have special educational needs, but all of whom share a wide range of learning interests, aptitudes and talents.

Curriculum components at senior cycle promote *a balance between knowledge and skills*, and the kinds of learning strategies relevant to participation in, and contribution to, a changing world where the future is uncertain.

The experience of senior cycle

The vision of senior cycle (Figure 2) sees the learner at the centre of the educational experience. That experience will enable students to be resourceful, to be confident, to participate actively in society, to build an interest in learning, and develop an ability to learn throughout their lives.

This vision of the learner is underpinned by the values on which senior cycle is based and it is realised through the principles that inform the curriculum as it is experienced by students in schools. The curriculum, made up of subjects and courses, embedded key skills, clearly expressed learning outcomes, and supported by a range of approaches to assessment, is the vehicle through which the vision becomes a reality for the learner.

At a practical level, the provision of a high-quality educational experience in senior cycle is supported by:

- effective curriculum planning, development, organisation and evaluation
- teaching and learning approaches that motivate and interest students, that enable them to progress, that deepen and apply their learning, and that develop their capacity to reflect on their learning
- professional development for teachers and school management that enables them to lead curriculum development and change in their schools
- a school culture that respects students, that encourages them to take responsibility for their own learning over time, and that promotes a love of learning.

Senior cycle education is situated in the context of a broader education policy that focuses on the contribution that education can make to the development of the learner as a person and as a citizen. It is an education policy that emphasises the promotion of social cohesion, the growth of society and the economy, and the principle of sustainability in all aspects of development.

RESOURCEFUL

they show their imagination, intelligence, intuition and other talents through

curiosity

enquiry

open-mindedness

reflection

connecting learning

innovation

problem solving

creativity

CONFIDENT

they develop their physical and mental well-being and

become self-aware

have high self-efficacy

engage with ethics, values and beliefs

welcome opportunities

can cope with setbacks

can effect positive change

LEARNERS
resourceful, confident, engaged and active

ENGAGED

they participate in the social, community, national and international dimensions of their lives by

showing respect for others

forming and sustaining caring relationships

making informed decisions

building practical know-how

taking interest in and responsibility for their social and physical environment

developing moral/ethical and political understanding

making lifestyle choices that are sustainable

contributing to their own material wellbeing and the material wellbeing of society

ACTIVE LEARNERS

they pursue excellence in learning to the best of their ability and develop a love of learning by

seeking and using knowledge, and understanding how knowledge is created

experiencing passion for, rigour in and commitment to learning

developing intellectual and critical thinking skills

exercising autonomy and independence in learning

managing their learning and making learning choices

setting and achieving learning goals

pursuing learning qualifications

Figure 2: Vision of senior cycle

3 Computer Science

Rationale

The accelerated expansion of computing technologies and artificial intelligence into all our lives means students need to understand the principles of computer science now, more than at any other time. It is necessary for all students to understand the ethical and social role of computers in society. Computer science is the foundation of this change and so the study of Computer Science at Leaving Certificate has become highly relevant to almost all aspects of modern life, and to every career choice. Computer Science is the foundation, information technology is the application and digital literacy is the implication of computing technologies. Computational thinking is one of the most fundamental skills acquired through the study of computer science. It shares the characteristics of other sciences, such as problem solving, abstraction and logical reasoning. However, computational thinking involves the implementation of solutions using automation, programming and computer systems. Students studying this subject will gain both thinking and practical skills that are valuable well beyond the computer science classroom and are applicable in many contexts.

Aim

Leaving Certificate Computer Science aims to develop and foster the learner's creativity and problem solving, along with their ability to work both independently and collaboratively. Students will apply the fundamental practices and concepts of computer science and develop an appreciation of the diverse role of computing technology in society and the environment in which they live. Studying computer science will nurture students' interests and passions and empower them to engage confidently and actively with the world.

Objectives

The objectives of Leaving Certificate Computer Science are to enable students to:

- develop an understanding of how computing technology presents new ways to address problems; and to use computational thinking to analyse problems and to design, develop and evaluate solutions
- read, write, test, and modify computer programs
- develop an understanding of how computers work; the component parts of computer systems and how they interrelate, including software, data, hardware, communications, and users
- appreciate the ethical and social implications relating to the use of computing technology and information and identify the impact of technology on personal life and society
- understand how information technology has changed over time and the effects these changes may have on education, the workforce, and society
- evaluate the accuracy, relevance, appropriateness, comprehensiveness, and bias of online information sources
- work independently and collaboratively, communicate effectively, and become responsible, competent, confident, reflective, and creative users of computing technology.

4

Related learning

Leaving Certificate Computer Science builds on the knowledge, attitudes and broad range of transferable skills that stem from the student's educational experience at early childhood, primary and post-primary junior cycle. Computational thinking is a problem-solving methodology that can be automated and transferred across all disciplines. It allows us to solve problems, design systems, and understand the power and limits of human and machine intelligence. Computational thinkers use a set of core concepts to process and analyse data and create real and virtual artefacts.

Students who can think computationally are better able to conceptualise, understand and use computer-based technology, and so are better prepared for today's world and the future.

Early childhood

Aistear, the early childhood curriculum framework, celebrates early childhood as a time of wellbeing and enjoyment where children learn from experiences as they unfold. The theme of *Exploring* and *Thinking* is about children making sense of the things, places and people in their world by interacting with others, playing, investigating, questioning, and forming, testing and refining ideas. The theme of Communicating is about children sharing their experiences, thoughts, ideas, and feelings with others with growing confidence and competence, in a variety of ways and for a variety of purposes.

Primary school

The curriculum area of Social, Environmental and Scientific Education (SESE) at primary school provides opportunities for children to actively explore and investigate the world around them from a human, social and cultural perspective. A scientific approach to investigations fosters the development of important skills, concepts and knowledge through which children can observe, question, investigate, understand and think logically about living things and their environments, materials, forces, everyday events and problems. The knowledge and skills acquired may be applied in designing and making activities in which children perceive a need to create or modify elements of their environments. Through their investigations, children develop informed, critical and scientific perspectives that acknowledge the importance of founding judgements on a respect for facts, accuracy and reason.

Computer science builds on language skills developed at primary level. Through language, students learn to use appropriate sequencing, tenses, and vocabulary to tell and retell stories and personal and procedural narratives of increasing complexity. They learn to use topic-specific language to give information, to explain and to justify their ideas and to predict and reflect upon actions, events and processes relating to real and imaginary contexts. Language skills developed at primary level will help students of computer science to appreciate the importance of the correct use of language, and appreciate just how powerful words and language are in the context of social media.

The Primary School Mathematics Curriculum aims to provide children with a language and a system through which to analyse, describe, illustrate, model and explain a wide range of experiences, make predictions, and solve problems. Leaving Certificate Computer Science builds on these skills, as it supports students to think and communicate quantitatively and spatially, to solve problems, and to recognise situations where mathematics can be applied.

Junior cycle

Many of the Statements of Learning at junior cycle relate to Leaving Certificate Computer Science, especially those statements focused on problem solving, design, communication, and understanding the role and contribution of technology in society. In addition, the key skills required for successful learning by students across the curriculum at junior cycle are relevant for Leaving Certificate Computer Science.

Many junior cycle subjects and short courses have close links with computer science, particularly mathematics, science, CSPE, and the short courses in coding and digital media literacy.

Senior cycle

Many senior cycle subjects have close links with computer science. Computational thinking is a thought process (or a human thinking skill) that uses analytic and algorithmic approaches to formulate, analyse and solve problems. Whilst the alignment of computer science with the STEM subjects is obvious, the strategies learned in computer science also relate to learning in other subjects. For example, computer science shares similarities with language learning, as aspects such as pattern recognition, syntax, textual analysis, and argument formation are relevant to both fields of study. Computer science provides a context for students to develop metacognitive skills which will support them as they take responsibility for their own learning.

Further study

Students live in a technologically-rich world. Leaving Certificate Computer Science will provide students with the knowledge and skills that will help them to understand current computer technology and prepare them for emerging technologies. A foundation in this discipline will introduce students to the excitement and opportunities afforded by this growing and dynamic field, as well as preparing them for a range of rewarding careers.

Leaving Certificate Computer Science incorporates a broad range of transferable and trans-disciplinary skills such as problem solving, logical thinking, and creative design. It also promotes skills of synthesis, evaluation, communication, time management, organisation, and teamwork. These skills and capabilities provide support for further study and learning beyond formal education, including learning in areas such as computer programming, database analysis, computer science, computer engineering, software engineering, information technology and game development.

Society and community

Leaving Certificate Computer Science includes the study and discussion of current events and emerging technologies, which will stimulate student interest and curiosity and help them connect what they are learning in class with real-world events or situations. Exploring the benefits and drawbacks of current and future computing technologies, and most importantly their impact on people and societies, will help students develop and refine their understanding of how to use computing technology and information ethically. Additionally, students will explore the role that adaptive technology can play in the lives of people with special needs and how access to, and engagement with computing and technology is of ever-increasing importance to societies, democracies and human progress.

Community links are a valuable resource for schools and students participating in Leaving Certificate Computer Science. These links can take the form of participation in industry and local business mentoring/career programmes and university mentoring programmes, leading and participating in local coding clubs or school coding clubs, and collaborating with local community groups to use technology to solve a local problem.

Education for sustainable development

The *National Strategy on Education for Sustainable Development 2014-2020* highlights the need to integrate education for sustainable development (ESD) into the curriculum from pre-school to senior cycle. The National Strategy aims to ensure that education contributes to sustainable development by equipping learners with the relevant knowledge (the 'what'), the key dispositions and skills (the 'how') and the values (the 'why') that will motivate and empower them throughout their lives to become informed, active citizens who act for a more sustainable future.

Computer science supports education for sustainable development by integrating the key skills of senior cycle throughout its strands. Many of the contexts used to explore the knowledge and understanding of computer science provide opportunities to discuss the practical and ethical aspects of computing, and to consider the use of computers and related technology from a societal perspective.

In strand 1, the practices and principles of computer science are encountered in a context-based approach related to social, professional, and scientific contexts. Students will appreciate how the use of computing technology impacts on communities. In strand 2, students learn how solutions can be designed that exploit the power of computers. They will consider ethical dilemmas and contexts relating to the use of computers, including how the resources used in the product lifecycle—water, fuel, and electricity—can increase energy efficiency by changing systems and ways of working. In strand 3, as students build computational artefacts, they appreciate the possibilities of how computing technology can provide ways to protect natural resources. For example, students will learn how modelling can be used to optimise systems to improve efficiency and reduce the damaging impact of energy-consuming infrastructures and systems. Throughout the course, students will apply the fundamental practices and concepts of computer science and develop an appreciation of the diverse role of computing technology in society and the environment in which they live.

5 Structure of Leaving Certificate Computer Science

There are three strands in the Computer Science specification: Practices and principles, Core concepts and Computer science in practice. All three strands are interwoven and should be studied concurrently at different stages of the course and should not be studied in a linear order. Skills and knowledge learned in strands 1 and 2 are applied to collaborative learning tasks outlined in strand 3. In that way, the applied learning tasks provide further practical context. Student application in the strand 3 learning tasks should increase in complexity and sophistication, thus developing and deepening the skills and knowledge learned in strands 1 and 2.

Strand 1: Practices and principles

The overarching practices and principles of computer science are the behaviours and ways of thinking that computer scientists use. This strand underpins the specification and is fundamental to all learning activities. By becoming familiar with, and fluent in, the practices and principles that underpin good practice, students develop their ability to manage themselves and their learning across the subject.

Strand 2: Core concepts

The core concepts of computer science represent the major areas in the field of computer science: abstraction, data, computer systems, algorithms and evaluation/testing. Students engage with the core concepts theoretically and practically in this strand. As their skills and knowledge develop, they engage in the applied learning tasks outlined in strand 3. Conceptual and practical classroom-based learning is combined with experimental computer-based learning throughout the two years of the course.

Strand 3: Computer science in practice

Computer Science in practice provides multiple opportunities for students to apply the practices and principles and the core concepts. Students work in teams to carry out four applied learning tasks over the duration of the course, each of which results in the creation of a real or virtual computational artefact[1]. These artefacts should relate to the students' lives and interests. Where possible, the artefacts should be beneficial to the community and society in general. Examples of computational artefacts include programs, games, web pages, simulations, visualisations, digital animations, robotic systems, and apps.

The four applied learning tasks explore the four following contexts: Interactive information systems, Analytics, Modelling and simulation, and Embedded systems. The tasks provide opportunities for students to develop their theoretical and procedural understanding as they grapple with computer science practices, principles and core concepts in increasingly sophisticated applications.

---

[1] A computational artefact is anything created by a human using a computer..

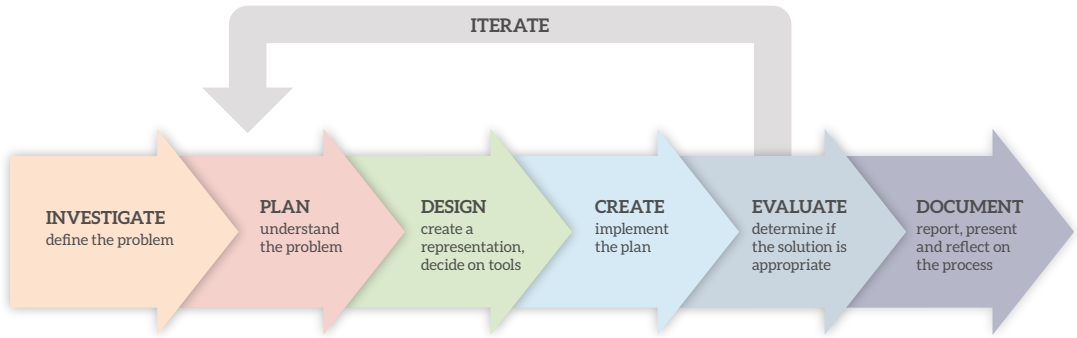

Figure 3: Overview of a design process

The output from each task is a computational artefact and a concise individual report outlining its development. In the report, students outline where and how the core concepts were employed. The structure of the reports should reflect the design process shown above in Figure 3. Initial reports could be in the form of structured presentations to the whole class. As students progress, reports should become detailed and individual. Reports are collected in a digital portfolio along with the computational artefact and must be verified as completed by both the teacher and the student. The (separate) externally-assessed coursework will be based on all learning outcomes, with those of strand 3 being particularly relevant.

Strand 1: Practices and principles	Strand 2: Core concepts	Strand 3: Computer science in practice
▸ Computers and society ▸ Computational thinking ▸ Design and development	▸ Abstraction ▸ Algorithms ▸ Computer systems ▸ Data ▸ Evaluation/Testing	▸ Applied learning task 1 - Interactive information systems ▸ Applied learning task 2 - Analytics ▸ Applied learning task 3 - Modelling and simulation ▸ Applied learning task 4 - Embedded systems

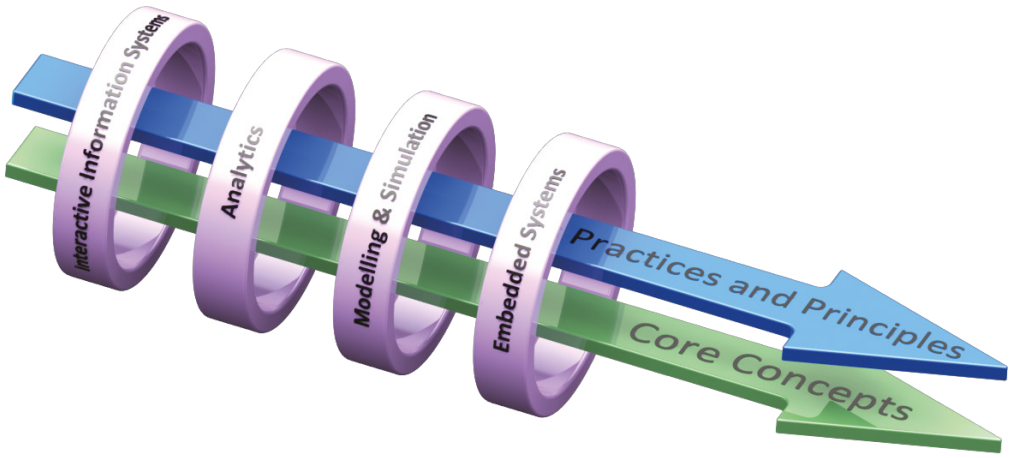

Figure 4: Structure of Leaving Certificate Computer Science

6 Key skills of senior cycle

Recent developments in curriculum and assessment at senior cycle have focused on the embedding of key skills within learning outcomes. This is accompanied by a different approach to assessment in which students can generate responses that reveal the depth of their understanding. The embedding of key skills requires careful consideration of the balance between knowledge and skills in the curriculum and in learning, and of finding appropriate ways of assessing them.

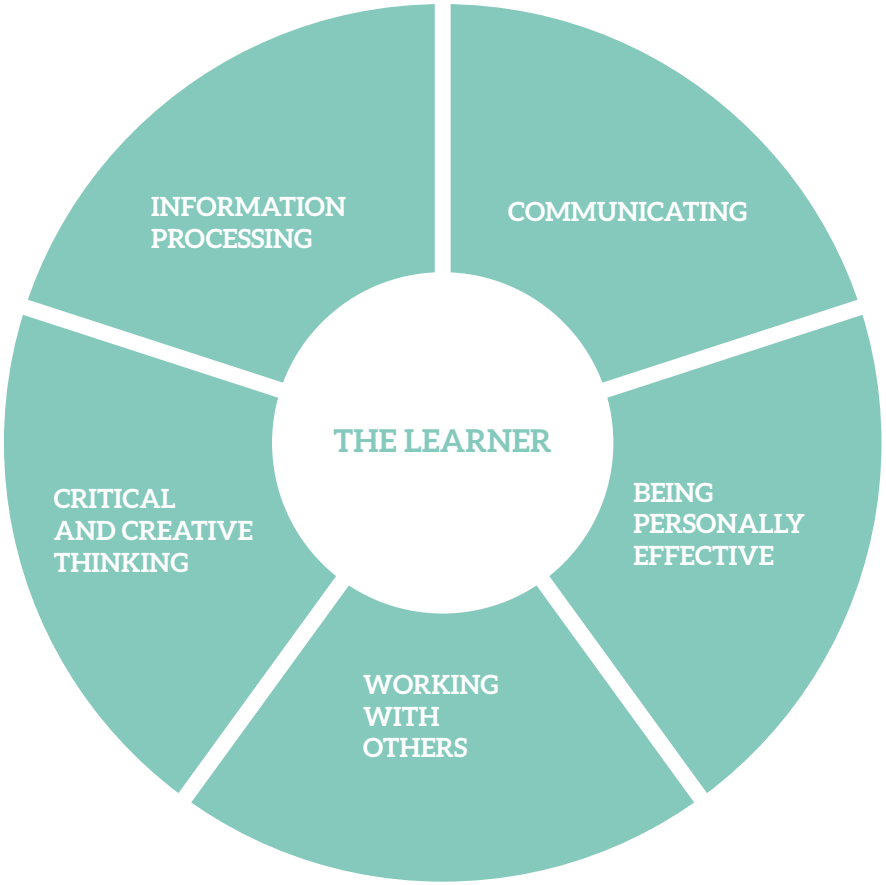

Figure 5: Key skills of senior cycle

The key skills of information processing; being personally effective; communicating; critical and creative thinking and working with others, and the learning outcomes associated with them, comprise the NCCA Key Skills Framework. The Key Skills Framework was developed to provide a common, unified approach for embedding key skills across all future Leaving Certificate specifications. These skills are identified as being important for all students to achieve to the best of their ability, both during their time in school and in the future, and to fully participate in society, in family and community life, the world of work and lifelong learning. Computer science develops these skills in the following ways:

Information processing

Learning in computer science takes place in an information-intensive environment; it promotes independent research activities in which students are required to access a wide variety of external materials to tackle questions. Tasks in computer science address selection, evaluation and recording of information; as students engage in problem solving, they make decisions and judgments based on data and qualitative and quantitative information. In this information-intensive environment, students develop an appreciation of the differences between information and knowledge and the roles that both play in making decisions and judgements. Programming teaches respect for accuracy and attention to detail and provides a platform to manipulate and process abstract forms of information. The consequence of a lack of precision is that the program fails or produces inaccurate or inconsistent results for different inputs. Similarly, the idea of breaking down a problem into sub-problems that can be solved separately takes a very concrete form in computer science, in which information-systems can be accessed through carefully-defined interfaces.

Critical and creative thinking

Design, modelling, and programming require careful analysis of patterns and relationships, which develops skills of higher-order reasoning and problem solving. Part of computational thinking is the ability to identify, analyse and deconstruct problems, explore options and alternatives, and hence solve problems. Hypothesising, making predictions, examining evidence, and reaching conclusions underpins the core of all the activities that students will undertake as part of computer science. As they develop these skills, students reflect critically on the forms of thinking and values that shape their own perceptions, opinions and knowledge. This develops the metacognitive dimension of knowledge which is essential in promoting good habits of mind. In computer science, students are designers and creators of technology rather than merely users of technology. Students must be creative and expressive to design artefacts that solve specific problems.

Communicating

Strong communication skills are developed in collaborative project work. Students use technology to communicate both face-to-face and through digital media. Although literacy skills are not targeted directly, they enable full participation in the learning experience. Internet research and the use of external sources require and build analysis and interpretation skills. Students will read a wide range of information sources. As part of the course students will be required to express and share their opinions through dialogue, discussion and argument. This encourages engaging in dialogue, listening attentively and eliciting opinions, views and emotions. They will also learn to provide technical information in ways that are relevant to and easily understood by people with diverse levels of technical knowledge and understanding. There is an opportunity to develop communication skills further as students compose and present using a variety of media.

Working with others

Leaving Certificate Computer Science is underpinned by collaboration and working with others. In their project work, students gain appreciation of the dynamics of groups and the social skills needed to engage in collaborative work. Computer science contributes to an appreciation that working collectively can help motivation, release energy and capitalise on all the talents in a group. One of the crucial factors in working with others is to identify, evaluate and achieve collective goals. Students learn to negotiate and resolve conflicts as they discuss their different strategies and achieve consensus.

Being personally effective

Self-awareness and persistence in the face of challenges enable students to grow and to develop. In computer science, students work on uncertain problems and learn to persist through ambiguity and face the risk of failure. As they work through challenges and potential failure they build persistence and resilience which serves them in all areas of their lives. There is no right way to answer a problem or set up a problem-solving strategy; as students build confidence in their self-direction they develop tenacity and rigour. Through developing the skill of being personally effective, students develop strategies for problem solving and for learning in general.

7 Teaching and learning

Senior cycle students are encouraged to develop the knowledge, skills, attitudes, and values that will enable them to become independent learners and to develop a lifelong commitment to improving their learning. Leaving Certificate Computer Science supports the use of a wide range of teaching and learning approaches. During the course, students will develop learning strategies that are transferable across different tasks and different subjects, enabling them to make connections between computer science, other subjects, and everyday experiences. Through engaging in self-directed learning and reflection, students will plan, monitor, and evaluate their own learning and develop a positive sense of their own capacity to learn. By engaging in group work students will develop skills in reasoned argument: listening to each other, informing one another of what they are doing, and reflecting on their own work and that of others. They will develop skills in communication by collaborating to generate reports and present them to their peers. The strand 3 tasks will enable students to take an active role in their own learning by setting goals, developing action plans, and receiving and responding to assessment feedback.

Applied learning tasks

Learning in computer science needs, as far as is practical, to be applied to problem solving and design exercises. The strand 3 applied learning tasks that students undertake collaboratively during the two years of the course, provide significant engaging opportunities for students to work within the practices and principles of computer science and to apply the core concepts in authentic situations. The resourcefulness of the student will be paramount to the success of the tasks. Students will be expected to learn new concepts and skills according to the demands of the chosen tasks. The computational artefacts that students design should be personally relevant to them or their peers, to their community or to society in general. Examples of computational artefacts include programs, simulations, visualisations, games, digital animations, robotic systems, and apps. Over the course of the two years of computer science students will:

1. Create an artefact or website that can display information from a database.
2. Create an interdisciplinary artefact using some form of analytics.
3. Develop a computer system that simulates or models a problem that is difficult to solve analytically.
4. Implement an embedded system that uses sensors and controls digital inputs and outputs.

In each of the tasks, students work together to apply learning from strands 1 and 2, in addition to cumulative learning from the tasks, so that they will have the opportunity to achieve all of the learning outcomes to their full extent.

The learning outcomes from all strands are interwoven and to complete their strand 3 applied learning tasks students:

- approach problems in a systematic way and use **abstraction** to identify tasks and select appropriate strategies to generate solutions
- create visual representations or models, and decide which tools to use and which **algorithms** to use, adapt or create as they employ appropriate techniques to develop their solution
- develop **computer systems** as they use programming, analysis and design skills combined with hardware knowledge to create network/Internet/cloud-based applications
- **evaluate and test** their solutions to identify and remove errors from their programs and base their solutions upon integration, analysis and evaluation of qualitative and quantitative information and data.

As they progress through the practical exercises and applied learning tasks, students learn from their successes and their mistakes. They take this learning to the next task to effectively solve new problems in different situations.

Teachers will assess and provide feedback on student learning as part of ongoing teaching and learning in the classroom. The strand 3 tasks will not be assessed by the SEC. The learning achieved through practical exercises and the applied learning tasks will be assessed both by the coursework project assessment and by the end-of-course examination. Both teacher and student will be required to verify completion of the strand 3 applied learning tasks.

Differentiation

The Leaving Certificate Computer Science specification is differentiated to cater for students of differing abilities and levels of achievement.

DIFFERENTIATION THROUGH THE LEARNING OUTCOMES OF THE SPECIFICATION

Ordinary level	Higher level
Only the learning outcomes presented in normal type. Students engage with a broad range of knowledge, mainly concrete in nature, but with some elements of abstraction or theory. They will be expected to demonstrate and use a moderate range of practical and cognitive skills and tools and to plan and develop simple investigative strategies. They will be expected to select from a range of procedures and apply known solutions to a variety of problems in both familiar and unfamiliar contexts. They will design and produce computational artefacts that serve a useful purpose.	All learning outcomes including those in **bold type.** Students engage with a broad range of knowledge, including theoretical concepts and abstract thinking, with significant depth in some areas. They will be expected to demonstrate and use a broad range of specialised skills and tools to evaluate and use information, to plan and develop investigative strategies, and to determine solutions to varied, unfamiliar problems. They will be expected to identify and apply skills and knowledge in a wide variety of both familiar and unfamiliar contexts. They will design and produce computational artefacts that serve a useful purpose.

Table 2: Differentiation in Leaving Certificate Computer Science

DIFFERENTIATION IN TEACHING AND LEARNING

Students vary in the amount and type of support they need to be successful. The use of strategies for differentiated learning, such as adjusting the level of skills required, varying the amount and the nature of teacher intervention, and varying the pace and sequence of learning will allow students to interact at their own level.

DIFFERENTIATION IN ASSESSMENT

Assessment of Leaving Certificate Computer Science will be based on the learning outcomes in the specification. The end-of-course examination will be assessed at two levels, Higher and Ordinary. At Higher level, all the learning outcomes will be assessed including those presented in bold type. At Ordinary level, only those learning outcomes that are presented in normal type will be assessed. Examination questions will require candidates to demonstrate knowledge, understanding, application, analysis, and evaluation appropriate to each level. Differentiation at the point of assessment will also be achieved through the depth and complexity of the questions and tasks, the stimulus material used, and the extent of the structured support provided for examination candidates at different levels.

Time allocation

Computer science is designed for 180 hours of class contact time. Meeting each learning outcome will be achieved through an interweaving of all three strands, and through a balance of theoretical learning, applied learning (through both the four applied learning tasks and tasks set by the teacher), problem-based learning and project management.

8 Strands and learning outcomes

Strand 1: Practices and principles

The practices and principles of computer science describe the behaviours and ways of thinking that computationally-literate students use to fully engage in a data-rich and interconnected world. Computational thinking, at the heart of computer science practices, is a problem-solving process that involves designing solutions that exploit the power of computers. The practices and principles are encountered in a context-based approach related to social, professional, and scientific contexts. Studying the role of computers in society will enhance students' attitudes towards computer science and make it more meaningful and relevant. In learning about designing and developing, students will recognise the creative challenge involved in creating artefacts and in project management.

Students learn about:[2]	Students should be able to:
S1: Computational thinking	1.1 describe a systematic process for solving problems and making decisions
	1.2 explain how the power of computing enables different solutions to difficult problems
Problem solving	1.3 solve problems by deconstructing them into smaller units using a systematic approach in an iterative fashion
	1.4 solve problems using skills of logic
Logical thinking	1.5 evaluate alternative solutions to computational problems
	1.6 explain the operation of a variety of algorithms
Algorithmic thinking	1.7 develop algorithms to implement chosen solutions
	1.8 evaluate the costs and benefits of the use of computing technology in automating processes
	1.9 use modelling and simulation in relevant situations
	1.10 discuss when heuristics should and could be used and explain the limitations of using heuristics

[2] The column 'Students learn about' lists some specific areas that students must learn.

Students learn about:[2]	Students should be able to:
S1: Computers and society Social and ethical considerations of computing technologies **Turing machines** The Internet Machine learning Artificial intelligence User-centred design	1.11 discuss the complex relationship between computing technologies and society including issues of ethics 1.12 compare the positive and negative impacts of computing on culture and society 1.13 identify important computing developments that have taken place in the last 100 years and consider emerging trends that could shape future computing technologies 1.14 explain when and what machine learning and AI algorithms might be used in certain contexts 1.15 consider the quality of the user experience when interacting with computers and list the principles of universal design, including the role of a user interface and the factors that contribute to its usability **1.16 compare two different user interfaces and identify different design decisions that shape the user experience** 1.17 describe the role that adaptive technology can play in the lives of people with special needs 1.18 recognise the diverse roles and careers that use computing technologies
S1: Designing and developing Design process Working in a team, assigning roles and responsibilities Communication and reporting Software development and management	1.19 identify features of both staged and iterative design and development processes 1.20 collaborate and assign roles and responsibilities within a team to tackle a computing task 1.21 identify alternative perspectives, considering different disciplines, stakeholders and end users 1.22 read, write, test, and modify computer programs 1.23 reflect and communicate on the design and development process

Strand 2: Core concepts

This strand introduces five core concepts that represent major content areas in the field of computer science: *Abstraction, Algorithms, Computer systems, Data,* and *Evaluation and testing.* The core concepts are developed theoretically and applied practically. In this way, conceptual classroom-based learning is intertwined with experimental computer lab-based learning throughout the two years of the course.

Students learn about:	Students should be able to:
S2: Abstraction	2.1 use abstraction to describe systems and to explain the relationship between wholes and parts 2.2 use a range of methods for identifying patterns and abstract common features 2.3 implement modular design to develop hardware or software modules that perform a specific function 2.4 illustrate examples of abstract models
S2: Algorithms Programming concepts	2.5 use pseudo code to outline the functionality of an algorithm
Sorting: Simple sort, Insert sort, Bubble sort, **Quicksort**	2.6 construct algorithms using appropriate sequences, selections/conditionals, loops and operators to solve a range of problems, to fulfil a specific requirement 2.7 implement algorithms using a programming language to solve a range of problems
Search: Linear search, Binary search	2.8 apply basic search and sorting algorithms and describe the limitations and advantages of each algorithm 2.9 assemble existing algorithms or create new ones that use functions **(including recursive)**, procedures, and modules
Algorithmic complexity	**2.10 explain the common measures of algorithmic efficiency using any algorithms studied**
S2: Computer systems **CPU: ALU, Registers, Program counter, Memory**	2.11 describe the different components within a computer and the function of those components
Basic electronics: voltage, current, resistors, capacitors, transistors	2.12 describe the different types of logic gates **and explain how they can be arranged into larger units to perform more complex tasks**
Operating system layers: Hardware, OS, Application, User	2.13 describe the rationale for using the binary number system in digital computing and how to convert between binary, hexadecimal and decimal 2.14 describe the difference between digital and analogue input
Web infrastructure - Computer Network Protocols: HTTP, **TCP, IP, VOIP**	2.15 explain what is meant by the World Wide Web (WWW) and the Internet, including the client server model, hardware components **and communication protocols**

Students learn about:	Students should be able to:
S2: Data Boolean, integer, real, char, string, date, array 8-bit ASCII **Non-Roman character sets** Unicode: UTF-8, Emojis Information systems	2.16 use data types that are common to procedural high-level languages 2.17 use ASCII and Unicode character sets to encode/decode a message and consider the importance of having such standards 2.18 collect, store and sort both continuous and discrete data
S2: Evaluation and testing Debugging Testing: Unit test, **Function test, System test**	2.19 test solutions and decisions to determine their short-term and long-term outcomes 2.20 identify and fix/debug warnings and errors in computer code and modify as required 2.21 critically reflect on and identify limitations in completed code **and suggest possible improvements** 2.22 explain the different stages in software testing

Strand 3: Computer science in practice

Computer science in practice provides multiple opportunities for students to use their conceptual understanding in practical applications. Over the two years of the course students engage with four team-based applied learning tasks. Student groups plan, design and develop computational artefacts that are personally relevant or beneficial to their community and society in general. Examples of computational artefacts include programs, games, simulations, visualisations, digital animations, robotic systems, and apps. Students are expected to document, reflect and present on each applied learning task.

APPLIED LEARNING TASK 1: INTERACTIVE INFORMATION SYSTEMS

Design is one of the key practices and principles of computer science. As designers and creators of technology, students can be innovative and expressive through the creation of artefacts. Computer science is also an information-intensive discipline that involves the selection, evaluation, recording and presentation of information. In this applied learning task, students will develop an interactive website that can display information (either local or remote data) from a database to meet a set of user needs. Through planning and designing an infrastructure that can display data, students will develop their knowledge of the role computing systems can play in communicating with and providing information about the world around them.

Students learn about:	Students should be able to:
Information systems User-centred design Web design File systems and **relational** databases Design process	3.1 understand and list user needs/requirements before defining a solution 3.2 create a basic **relational** database to store and retrieve a variety of forms of data types 3.3 use appropriate programming languages to develop an interactive website that can display information from a database that meets a set of users' needs

APPLIED LEARNING TASK 2: ANALYTICS

Hypothesising, making predictions, examining evidence, recognising patterns and reaching conclusions are at the heart of computer science. In this applied learning task, students will identify an interdisciplinary topic, develop a hypothesis and utilise existing resources to highlight the salient information and inform future decisions. By identifying, analysing, and deconstructing a problem, students will deepen their understanding of the practices and principles of computer science.

Students learn about:	Students should be able to:
Analytics Abstraction Data collection and analysis Interpretation of data Algorithms	3.4 develop algorithms that can find the frequency, mean, median and mode of a data set 3.5 structure and transform **raw** data to prepare it for analysis 3.6 represent data to effectively communicate in a graphical form 3.7 use algorithms to analyse and interpret data in a way that informs decision-making

APPLIED LEARNING TASK 3: MODELLING AND SIMULATION

Modelling, programming, and coding require careful analysis of patterns and relationships to solve problems. In this applied learning task, students will engage with a problem that is difficult to solve analytically, but that is amenable to a solution using simulation or modelling. Students will develop a computer system that simulates or models the problem. Engaging with a problem in this way will heighten students' awareness and understanding of how algorithms can be used across a wide range of disciplines and subjects.

Students learn about:	Students should be able to:
Modelling/simulation Abstraction Algorithms	3.8 develop a model that will allow different scenarios to be tested 3.9 analyse and interpret the outcome of simulations **both before and after modifications have been made** **3.10 explain the benefits of using agent-based modelling and how it can be used to demonstrate emergent behaviours**

APPLIED LEARNING TASK 4: EMBEDDED SYSTEMS

The design and application of computer hardware and software are a central part of computer science. In this applied learning task, students will implement a microprocessor system that uses sensors and controls digital inputs and outputs as part of an embedded system. By building the component parts of a computer system, students will deepen their understanding of how computers work and how they can be embedded in our everyday environments.

Students learn about:	Students should be able to:
Embedded systems	3.11 use and control digital inputs and outputs within an embedded system
Computing inputs and outputs	3.12 measure and store data returned from an analogue input
Computer systems	3.13 develop a program that utilises digital and analogue inputs
Design process	3.14 design automated applications using embedded systems

9 Assessment

Assessment in senior cycle involves gathering, interpreting and using information about the processes and outcomes of learning. It takes different forms and is used for a variety of purposes. It is used to determine the appropriate route for students through a differentiated curriculum, to identify specific areas of difficulty or strength for a given student and to test and certify achievement. Assessment supports and improves learning by helping students and teachers to identify next steps in the teaching and learning process.

As well as varied teaching strategies, varied assessment strategies will support learning and provide information that can be used as feedback so that teaching and learning activities can be modified in ways that best suit individual learners. By setting appropriate and engaging tasks, asking higher-order questions and giving feedback that promotes learner autonomy, assessment will support learning and summarise achievement.

There are several important aspects of computer science assessments to consider: the use of authentic tasks, the breadth of concepts being assessed, and the special role computers can play in delivering instruction and measuring performance. Compared to other subjects, computer science provides a unique opportunity to take advantage of online learning and computerised assessment. Students can create programs such as games, apps and simulations within an environment that also collects data, analyses achievement, and communicates progress to both students and teachers.

Project-based/portfolio assessment of coursework can measure many of the computer science learning outcomes associated with performance. Coursework assessment provides students with opportunities to demonstrate their understanding in multiple ways that highlight their creativity, interests, and understanding.

Assessment for certification

Assessment for certification is based on the aim, objectives, and learning outcomes of this specification. Differentiation at the point of assessment is achieved through examinations at two levels –Ordinary level and Higher level.

There are two components to the assessment of Leaving Certificate Computer Science: (i) an end-of-course examination and (ii) coursework. Both components reflect the relationship between the application of skills and the theoretical content of the specification.

The end-of-course assessment may comprise questions of varied format and type. The questions will assess both the core concepts and the practices and principles of computer science. The questions will be based on the learning outcomes in the specification; however, any question may address more than one learning outcome, or require students to combine knowledge and skills from across several areas of the specification.

The coursework assessment will require students to demonstrate proficiency in course content and skills that are not easily assessed by the end-of-course examination. The assessment will require students to create an innovative computational artefact, and to report on the work and process involved. Students must acknowledge, through appropriate citations and references, the source or author of all information or evidence taken from someone else's work. It is anticipated that student work for both components will be submitted electronically and will be marked by the State Examinations Commission (SEC).

The assessment of both components will be aligned with the objectives of the specification, and assess the extent to which students:

- understand how computing technology presents new ways to address problems
- use computational thinking to analyse problems, and to design, develop and evaluate solutions
- can read, write, test and modify computer programs
- understand how computers work and the component parts of computer systems and how they interrelate, including software, data, hardware, communications, and users
- understand the evolution of computing technology and appreciate the ethical and social implications of the use of computing technology in contemporary and future social issues
- work independently, communicate effectively, and understand the factors that influence collaboration and teamwork
- become responsible, competent, confident, reflective and creative users of computing technology.

STRUCTURE OF ASSESSMENT FOR CERTIFICATION

There are two assessment components at each level, an end-of-course examination (70%) and coursework (30%).

Component	Percentage
End-of-course examination ▸ Written and computer-based assessment of learning outcomes	**70**
Coursework assessment ▸ One computational artefact with report	**30**
Total	**100**

Table 3: Overview of assessment

End-of-course examination

The end-of-course examination will be made up of a range of question types. The questions will require students to demonstrate knowledge, understanding, application, analysis, evaluation and creativity appropriate to each level. The key skills are embedded in the learning outcomes and will be assessed in the context of the learning outcomes. The examination will assess:

- knowledge and recall of facts, principles and methods relating to computer science
- application of knowledge and understanding of the principles and concepts of computer science, including abstraction, logic, algorithms and data representation, and how to analyse problems in computational terms
- ability to write code and to compile, test and debug program code
- ability to evaluate computer systems that solve problems, making reasoned judgements about these and presenting conclusions
- problem solving based on integration, analysis and evaluation of qualitative and quantitative information and data, using knowledge gained from all three strands
- understanding of the ethical, historical, environmental and technological aspects of Computer Science, and of how computer science contributes to the social and economic development of society.

The examination will have sections covering questions that address:

- Computer science topics across the entire specification
- Practical questions requiring the use of a programming language
- Questions based on contexts and drawn from across different areas of the specification.

END-OF-COURSE ASSESSMENT CRITERIA

High level of achievement	Moderate level of achievement	Low level of achievement
demonstrates a thorough knowledge and understanding of the principles and concepts of Leaving Certificate Computer Science with few significant omissions.	demonstrates a good knowledge and understanding of the principles and concepts of Leaving Certificate Computer Science.	demonstrates a limited knowledge and understanding of the principles and concepts of Leaving Certificate Computer Science.
consistently applies knowledge and understanding of the principles and concepts of computer science to problem solving in both familiar and new contexts using appropriate computational thinking methods.	applies knowledge and understanding of principles and concepts of computer science to problem solving in both familiar and some new contexts using appropriate computational thinking methods.	selects appropriate facts and principles to solve problems concerning familiar material using a limited range of computational thinking methods.
is able to write, compile, test and debug program code in a manner that eliminates almost all errors.	is able to write, compile, test and debug program code with some errors.	only has a limited ability to write compile, test and debug program code.
consistently designs, programs and evaluates computer systems that solve problems, making reasoned judgements about these and presenting conclusions.	designs programs and evaluates some computer systems that solve problems, making judgements about these and presenting conclusions.	designs programs that do not solve problems that they were designed to solve. Presents limited evaluation of some computer systems without making judgements about these or presenting conclusions.
demonstrates a thorough knowledge and understanding of the ethical, historical, environmental and technological aspects of computer science, and of how computer science contributes to her/his personal life and to the social and economic development of society.	demonstrates a good knowledge and understanding of the ethical, historical, environmental and technological aspects of computer science, and of how computer science contributes to her/his personal life and to the social and economic development of society.	demonstrates a limited knowledge and understanding of the ethical, historical, environmental and technological aspects of computer science, and of how computer science contributes to her/his personal life and to the social and economic development of society.

Table 4: End-of-course examination assessment criteria

Coursework assessment

The coursework assessment will use practical situations to assess how students design data structures and develop algorithms, integrate ideas, test hypotheses, and explore alternative approaches. It will be similar to the structure of the strand 3 applied learning tasks that students complete during the two years of the course. However, the coursework assessment must be carried out individually. Students will not be permitted to work in groups for the coursework assessment.

Towards the end of term 1 of the second year of the course, the State Examinations Commission (SEC) will set a task in which students are required to generate a computational artefact in response to a brief set out by the SEC. The time-period for completion of the coursework will be set out in the brief. A period of 10 weeks is anticipated, after which the completed task is submitted, electronically, to the SEC for marking. The date for submission will be set by the SEC each year.

COURSEWORK ASSESSMENT CRITERIA

High level of achievement	Moderate level of achievement	Low level of achievement
systematically breaks down problems and filters out unnecessary information and can explain the processes involved. Uses innovative thinking in design and development.	identifies problems/things that can be solved. Uses innovative thinking in design and development.	engages with limited aspects of the problem. Avoids problems/ challenges that have more than one step or part to solving them. May unintentionally over-complicate problems.
independently designs, models, tests, debugs and refines solutions (using a test plan and data where appropriate), and chooses an appropriate way to represent solutions.	iteratively develops, tests, and debugs solutions.	expresses ideas at a basic level. Submits the first working solution as the finished product.Testing, debugging and refinement of solutions is done in a linear fashion.
consistently displays curiosity to exhaustively investigate and analyse a broad range of appropriate problems/solutions.	deals with complexity and with open-ended problems.	requires a plan and clear expectations or deliverables. Follows instructions and is limited in their self-direction.
independently identifies and acts on patterns in problems/solutions. Independently seeks out pre-existing solutions, transferring ideas and/or solutions from one problem context to another.	adapts existing knowledge or solutions to solve new problems and weighs outcomes carefully.	application of previous learning to new problems is limited.
celebrates ambiguity and having different interpretations. Compares the performance of different solutions that solve the same problem.	shows an ability to tolerate ambiguity in both problems and solutions.	uses pre-learned solutions to attempt to solve new problems. Has difficulty accepting ambiguity in problems or their solutions.

Table 5: Coursework assessment criteria

ASSESSMENT PROGRAMMING LANGUAGE

Leaving Certificate Computer Science does not require a specific language. However, following the initial years of the subject, Python will be the language assessed in the end-of-course assessment and Python and Javascript in the coursework assessment. This will continue to be reviewed on an ongoing basis. There is no restriction in choice of language used in the strand 3 applied learning tasks.

Reasonable accommodations

This Leaving Certificate Computer Science specification requires that students engage with practical applications of computational thinking on an ongoing basis throughout the course. In addition, the assessment involves a coursework element, which accounts for 30% of the total marks awarded. This emphasis on practical applications may have implications for students with physical/medical/sensory and/or specific learning difficulties. In this context, the scheme of Reasonable Accommodations, operated by the State Examinations Commission, is designed to assist candidates in the Leaving Certificate who have physical/medical/sensory and/or specific learning difficulties. In considering the course, it is recommended that any student with a disability should contact the State Examinations Commission beforehand to find out if any accommodations can be made.

10 Appendices

Appendix A: Glossary of action verbs used

Verb	Description
Analyse	study or examine something in detail, break down in order to bring out the essential elements or structure; identify parts and relationships, and to interpret information to reach conclusions
Annotate	add brief notes of explanation to a diagram or graph
Apply	select and use information and/or knowledge and understanding to explain a given situation or real circumstances
Appraise	evaluate, judge or consider text or a piece of work
Appreciate	recognise the meaning of, have a practical understanding of
Brief description/ explanation	a short statement of only the main points
Argue	challenge or debate an issue or idea with the purpose of persuading or committing someone else to a particular stance or action
Calculate	obtain a numerical answer showing the relevant stages in the working
Classify	group things based on common characteristics
Comment	give an opinion based on a given statement or result of a calculation
Compare	give an account of the similarities between two (or more) items or situations, referring to both (all) of them throughout
Consider	describe patterns in data; use knowledge and understanding to interpret patterns, make predictions and check reliability
Construct	develop information in a diagrammatic or logical form; not by factual recall but by analogy or by using and putting together information
Contrast	detect correspondences between two ideas
Convert	change to another form
Criticise	state, giving reasons the faults/shortcomings of, for example, an experiment or a process
Deduce	reach a conclusion from the information given
Define	give the precise meaning of a word, phrase, concept or physical quantity
Demonstrate	prove or make clear by reasoning or evidence, illustrating with examples or practical application

Verb	Description
Derive	arrive at a statement or formula through a process of logical deduction; manipulate a mathematical relationship to give a new equation or relationship
Describe	develop a detailed picture or image of, for example a structure or a process, using words or diagrams where appropriate; produce a plan, simulation or model
Determine	obtain the only possible answer by calculation, substituting measured or known values of other quantities into a standard formula
Differentiate	Identify what makes something different
Discuss	offer a considered, balanced review that includes a range of arguments, factors or hypotheses; opinions or conclusions should be presented clearly and supported by appropriate evidence
Distinguish	make the differences between two or more concepts or items clear
Estimate	give a reasoned order of magnitude statement or calculation of a quantity
Evaluate (data)	collect and examine data to make judgments and appraisals; describe how evidence supports or does not support a conclusion in an inquiry or investigation; identify the limitations of data in conclusions; make judgments about the ideas, solutions or methods
Evaluate (ethical judgement)	collect and examine evidence to make judgments and appraisals; describe how evidence supports or does not support a judgement; identify the limitations of evidence in conclusions and make judgments about ideas, solutions or methods
Explain	give a detailed account including reasons or causes
Examine	consider an argument or concept in a way that uncovers the assumptions and interrelationships of the issue
Find	general term that may variously be interpreted as calculate, measure, determine etc.
Formulate	Express the relevant concept(s) or argument(s) precisely and systematically
Group	identify objects according to characteristics
Identify	recognise patterns, facts, or details; provide an answer from a number of possibilities; recognise and state briefly a distinguishing fact or feature
Illustrate	use examples to describe something
Infer	use the results of an investigation based on a premise; read beyond what has been literally expressed
Investigate	observe, study, or make a detailed and systematic examination, in order to establish facts and reach new conclusions
Interpret	use knowledge and understanding to recognise trends and draw conclusions from given information
Justify	give valid reasons or evidence to support an answer or conclusion

Verb	Description
List	provide a number of points, with no elaboration
Measure	quantify changes in systems by reading a measuring tool
Model	generate a mathematical representation (e.g., number, graph, equation, geometric figure) for real world or mathematical objects, properties, actions, or relationships
Order	describe items/systems based on complexity and/or order
Outline	give the main points; restrict to essentials
Plot	a graphical technique for representing a data set, usually as a graph showing the relationship between two or more variables.
Predict	give an expected result of an event; explain a new event based on observations or information using logical connections between pieces of information
Prove	use a sequence of logical steps to obtain the required result in a formal way
Provide evidence	provide data and documentation that support inferences or conclusions
Recognise	identify facts, characteristics or concepts that are critical (relevant/ appropriate) to the understanding of a situation, event, process or phenomenon
Recall	remember or recognise from prior learning experiences
Relate	associate, giving reasons
Sketch	represent by means of a diagram or graph (labelled as appropriate); the sketch should give a general idea of the required shape or relationship, and should include relevant features
Solve	find an answer through reasoning
State	provide a concise statement with little or no supporting argument
Suggest	propose a solution, hypothesis or other possible answer
Synthesise	combine different ideas in order to create new understanding
Understand	have and apply a well-organised body of knowledge
Use	apply knowledge or rules to put theory into practice
Verify	give evidence to support the truth of a statement

Appendix B: Glossary of core concepts

Abstraction

Abstractions are formed by identifying patterns and extracting common features from specific examples to create generalisations. Using generalised solutions and parts of solutions designed for broad reuse simplifies the development process and helps to manage complexity.

Data

Students learn how data about themselves and their world is collected and used. Data is collected and stored so that it can be analysed to better understand the world and make more accurate predictions.

Data is collected with both computational and non-computational tools and processes.

Computing systems

A computing system consists of hardware, software, computational processes and networks and users. Students will develop programming, analysis and design skills combined with the hardware knowledge needed to create network/Internet/cloud-based applications. They will learn how computing devices (such as smart devices, desktop computers and tablets) communicate with each other and the world around them and how to plan and design the infrastructure and systems that allow this to happen.

Algorithms

An algorithm is a sequence of steps designed to accomplish a specific task. Algorithms are translated into programs, or code, to provide instructions for computing devices. Students learn how to read, write, modify and test algorithms, as well as how to evaluate competing algorithms. The words programming, coding and programming language are defined as:

- Programming is the craft of analysing problems and designing, writing, testing and maintaining programs to solve them
- Coding is the act of writing computer programs in a programming language
- A programming language is the formal language used to give a computer instruction

Software evaluation

Software evaluation is the process of determining if the program or combination of programs is the best possible solution to a given problem or task. The evaluation process should include factors such as feasibility, efficiency, and ethical use.

Software testing

Software testing is the process of finding and correcting errors (bugs) in a program or system and ensuring that the program produces the intended output. Debugging includes identifying errors, gaps, and missing requirements.

Heuristic

A heuristic is an approach to problem solving which aims to make an approximate solution to the problem. This can be used when time and resources are limited. The solution may not be feasible using classic or standard methods but should aim to approximate the optimal solution. This may involve the loss of precision, accuracy, optimal performance or completeness.

NCCA

An Chomhairle Náisiúnta Curaclaim agus Measúnachta
National Council for Curriculum and Assessment

An Roinn Oideachais
Department of Education