De Lisle College Curriculum

“Rooted and grounded in Love”.

Computer Science

Intent – 'Rooted and grounded in Love'

Intent – Rooted

It is our mission to enable all students to become confident, compassionate, and discerning users and creators of technology and digital media.

Our Computer Science and Digital Media curriculum is designed to equip students with the skills, knowledge, and values they need to flourish in a fast-moving and ever-changing digital world. From Year 7 through to post-16 study, our aim is to nurture problem-solvers and critical thinkers—young people who are reflective, analytical, and adaptable, prepared to take their place in the future with both technical expertise and a strong moral compass.

Our curriculum develops technical skill alongside ethical awareness. We teach programming and digital design through a structured, progressive model that enables students to understand not just how digital systems work, but why they are built the way they are—and how they impact individuals, society, and the planet. Students will learn to design digital content, code responsibly, and analyse media with clarity and purpose, always asking how technology can be used for good.

At Key Stage 5, students can specialise in A-Level Computer Science or A-Level Media Studies, or both. In Computer Science, they are challenged to think abstractly and systematically -- developing secure programming skills, working on extended practical projects, and exploring topics such as CPU architecture, searching and sorting algorithms, cybersecurity, and ethics in technology. In Digital Media, students develop their practical media production skills while studying the media industry, the psychology of media consumption, and the power of narrative in visual storytelling.

Our curriculum is rooted in the Catholic values of compassion, truth, and service. Inspired by the example of Christ, students are encouraged to use technology not just for innovation, but for inclusion and justice. Whether designing a program or analysing a media campaign, students are asked: *Who is this for? Who might it exclude? How can we make it better for all?* We draw on Catholic Social Teaching to reflect on issues such as digital poverty, misinformation, and the responsible use of AI.

We take inspiration from St. Isidore of Seville, the patron saint of the internet and technology, who devoted his life to gathering and sharing knowledge for the good of others. His legacy reminds us that information should be used to build understanding and community—not division. In his spirit, our students are called to be ethical digital citizens and creators—thoughtful, informed, and guided by a commitment to the common good. Through this curriculum, our students grow in understanding of themselves, of others, and of the digital world they will shape.

Computer Science

Year 12

De Lisle College
A Catholic Voluntary Academy

Curriculum Outline

Link to Catholic Social Teaching:

- **Human Dignity:** Consider how technology should respect privacy, accessibility and the needs of all users when designing software and digital services.
- **Rights and Responsibilities:** Explore ethical issues surrounding data protection, cybersecurity, hacking and the responsible use of personal information.
- **The Common Good:** Study how computer systems, networks and databases support communication, healthcare, education and wider society.
- **Justice and Inclusion:** Discuss the digital divide and how technology can both reduce and create inequalities in access and opportunity.
- **Stewardship:** Examine the environmental impact of computing, including energy use, e-waste and sustainable technology practices.
- **Participation:** Through programming projects, students develop solutions that meet genuine user needs and positively contribute to their communities.

Advent 1	12.1 – CPU Architecture, Memory and Storage 12.2 – Software and Operating Systems Programming – Core Techniques	**Advent 2**	12.3 – Application Generation 12.4 - Types of Programming Languages Programming – Intro to Object-Oriented Programming
Description	**CPU Architecture, Memory and Storage** Learn how computers process, store and retrieve data, exploring the components that make modern computing systems work. **Software and Operating Systems** Study how operating systems manage hardware and software and investigate the role of system and application software. **Core Techniques** Develop fundamental programming skills including variables, selection, iteration, procedures and problem-solving.	Description	**Application Generation** Explore the different stages of software development and how applications are designed, created and maintained. **Types of Programming Languages** Compare low-level and high-level languages and learn how programs are translated into machine-readable code. **Introduction to Object-Oriented Programming** Learn how classes, objects, inheritance and encapsulation are used to create efficient software solutions.

Lent 1	12.5 - Encryption and Hashing 12.6 - Networks 12.7 – Databases Programming – Introduction to 2D Game Development	**Lent 2**	12.8 – Web Technologies and Development 12.9 – Project Methodologies Programming – Mini-project
Description	**Encryption and Hashing** Discover how data is protected through encryption and hashing techniques used in cybersecurity and digital communication. **Networks** Study how devices communicate across networks, including network topologies, protocols and internet technologies. **Databases** Learn how data is organised, stored and queried using database structures and management systems. **Introduction to 2D Game Development** Apply programming skills to create simple interactive games while developing problem-solving and computational thinking.	Description	**Web Technologies and Development** Explore how websites are created using web technologies and understand how data is transmitted online. **Project Methodologies** Learn professional approaches to planning, designing and managing software development projects. **Mini-Project** Apply Year 12 knowledge and skills to design, develop and evaluate a small programming project.
Pentecost 1	12.11 Binary Numbers – Positive, Negative and Floating Point 12.12 Data Structures – Stacks and Queues Programming – Introduction to coursework	**Pentecost 2**	12.13 Data Structures – Linked Lists, Graphs and Trees 12.14 – Algorithms – Searching, Sorting and Pathfinding Programming – Coursework project – Analysis and Design
Description	**Positive, Negative and Floating Point** Learn how computers represent and process numbers using binary systems and data representation techniques. **Stacks and Queues** Investigate how data can be organised efficiently using common computer science data structures. **Introduction to Coursework** Begin planning the programming project required for the non-exam assessment component of the course.	Description	**Linked Lists, Graphs and Trees** Study advanced methods of organising data and their real-world applications in computer systems. **Algorithms: Searching, Sorting and Pathfinding** Explore efficient problem-solving techniques used to process, organise and analyse data. **Coursework Project Analysis and Design** Research a user problem and produce detailed plans, designs and requirements for a programming solution.

Computer Science

Year 13

De Lisle College
A Catholic Voluntary Academy

Curriculum Outline

Link to Catholic Social Teaching:

- **Human Dignity:** Consider how technology should respect privacy, accessibility and the needs of all users when designing software and digital services.
- **Rights and Responsibilities:** Explore ethical issues surrounding data protection, cybersecurity, hacking and the responsible use of personal information.
- **The Common Good:** Study how computer systems, networks and databases support communication, healthcare, education and wider society.
- **Justice and Inclusion:** Discuss the digital divide and how technology can both reduce and create inequalities in access and opportunity.
- **Stewardship:** Examine the environmental impact of computing, including energy use, e-waste and sustainable technology practices.
- **Participation:** Through programming projects, students develop solutions that meet genuine user needs and positively contribute to their communities.

Advent 1	13.1 Boolean Algebra 13.2 Legislation Programming – Coursework project – Implementation	**Advent 2**	13.3 Algorithms and Data Structures Programming – Coursework project – Implementation
Description	**Boolean Algebra** Learn the mathematical logic that underpins computer circuits, programming and data processing. **Legislation** Study legal and ethical issues in computing, including data protection, intellectual property and cybercrime. **Coursework Project Implementation** Develop and test the main software solution designed during the planning stage of coursework.	Description	**Algorithms and Data Structures** Build deeper understanding of advanced algorithms and their impact on program efficiency and performance. **Coursework Project Implementation** Continue developing, refining and documenting the coursework solution using professional programming practices.

Lent 1	13.4 – Year 12 Revisited – Part 1 Programming – Coursework project – Testing and Evaluation (Submission)	**Lent 2**	13.5 – Year 12 Revisited – Part 2 Final Mock exams, Immersion and exam prep
Description	**13.4 Year 12 Revisited Part 1** Review and consolidate key Year 12 topics while developing exam technique and confidence. **Programming: Coursework Testing and Evaluation** Complete testing, evaluation and documentation before submitting the final coursework project.	Description	**13.5 Year 12 Revisited Part 2** Reinforce understanding of core computer science concepts through revision, exam practice and application. **Final Mock Exams, Immersion and Exam Preparation** Prepare thoroughly for external examinations through targeted revision and assessment activities.
Pentecost 1	13.5 – Year 13 Revisited	**Pentecost 2**	COURSE COMPLETED
Description	**13.6 Year 13 Revisited** Review advanced topics, practise exam questions and strengthen understanding across the full specification.	Description	N/A