| Digital Technologies Curriculum | | Year Levels: 9-10 |
|---|---|---|
| **Content Descriptors** | **Example Can Do Statements - SOLO Taxonomy**<br>https://www.digitaltechnologieshub.edu.au/teachers/scope-and-sequence/9-10<br>https://aca.edu.au/curriculum/9-10/ | **Possible Activities**<br>(All Links work as of April 2020) |
| **Digital Systems**<br><br>**#1 Investigate the role of hardware and software in managing, controlling and securing the movement of and access to data in networked digital systems (ACTDIK034)** | Connected via a Network (Level 9)<br>Systems Thinking<br>• I can describe LAN and WAN networks.<br>• I can describe protocols such as HTTP, TCP, IP and SMTP.<br>• I can draw a labelled diagram to show how a network operates.<br>• I can explain how a file is sent over the internet or between computers and refer to relevant protocols.<br>• I can compress a file to send it via internet.<br>• I can set up a programming board and edit a Python program to send messages between devices.<br><br>Data: Controlled and Secure (Level 10)<br>Systems Thinking<br>• I can IDENTIFY ways authorisation is used to protect our data and information by restricting use and access to digital systems<br>• I can describe ways authorisation is used to protect our data and information by restricting use and access to digital systems.<br>• I can DESCRIBE different types of malicious code.<br>• I can EXPLAIN how encryption is used in cyber security of digital systems<br>• I can EXPLAIN how malicious code can affect digital systems<br>• I can DESIGN a secure system that considers approaches to cyber security. | • https://www.digitaltechnologieshub.edu.au/teachers/scope-and-sequence/9-10<br>• https://aca.edu.au/resources/#years-9-10<br>• **Working together** - Through role-play and other activities, students develop understandings of the role of hardware and software in managing, controlling and securing the movement of and access to data in networked digital systems.<br>• **The Internet: Wires, Cables and WiFi** - This excellent expanding video series is produced by code.org. Most relevant for this lesson sequence is the episode: The Internet: Packets, Routing and Reliability.<br>• **Talk to me!** - With a focus on input devices, students are encouraged to explore the possibilities and new types of functionality enabled by these technologies over time.<br>• **Keeping Secrets** - Encryption of data is a means of protecting data, one example being the use of secret and public keys. This learning sequence examines cryptography and modern encryption methods for transmitting digital data securely.<br>• **Infrastructure:** a PowerPoint presentation and activity to support students in exploring the elements of technological infrastructure.<br>• **New Tech and Old Tech**: this activity allows students to compare old and new tech to experience how significant the advances in technology have been. Includes detailed lesson plan and student resources. Note that teachers will need to create a free login to access this resource. |

| | | |
|---|---|---|
| | | • **Inventing the Internet:** students work together to invent solutions and protocols to many of the problems that arise in the structure and function of the Internet. (lessons 8 – 11)<br>• **Networks explained**: this PowerPoint presentation explains different types of computer networks (FUSE) |
| **Data and Information**<br><br>**#2 Analyse simple compression of data and how content data are separated from presentation (ACTDIK035)** | Systems Thinking<br>• I can describe how data is compressed<br>• I can reduce the file size of an audio or image file and explain the process as either lossless or lossy.<br>• I can compress a file to send it via internet.<br>• I can represent documents by separating content and presentation (e.g. HTML and CSS) and explain why this separation is important.<br>• I can use an algorithm to identify patterns in data (e.g. repeated pixels in an image) and represent them in a compressed way (e.g. run-length encoding). | • **Text Compression:** explores with students how computers have a limited amount of space to store data and information and how they represent it. (unplugged)<br>• **Digital Information Unit:** what can be represented with a single bit and how do we get a single bit of information from one place to another? This unit explores the technical challenges and questions that arise from the need to represent digital information in computers and transfer it between people and computational devices.<br>• **Seeing the Big Picture** - This practical learning sequence examines lossy and lossless techniques of data compression. Students perform comparisons of various compressions on a variety of images.<br>• **A Matter of Style** - To understand the benefits of stylesheets in separating style and content and to learn to use them |
| **Data and Information**<br><br>**#3 Develop techniques for acquiring, storing and validating quantitative and qualitative data from a range of sources, considering privacy and security requirements (ACTDIP036)**<br><br>**Data and Information**<br><br>**#4 Analyse and visualise data to create information** | Data Driven Innovation (Level 9)<br>Systems Thinking<br>• I can identify the types of data my smartphone usage would reveal.<br>• I can describe ways personal data is used to inform companies.<br>• I can explain the key components of a business digital solution that identifies and solves a problem<br>• I can explain how data is used in a relevant example such as automated vehicles.<br>• I can produce a flow chart that shows the use of data in a relevant example that involves automation.<br><br>Organise, Visualise and Analyse (Level 10) | • https://www.digitaltechnologieshub.edu.au/teachers/scope-and-sequence/9-10<br>• **Everything you always wanted to know** - Students design, build and evaluate their own database and perform queries and build reports based on that database. Students should have prior experience creating a flat file database.<br>• **Spreadsheets come alive** - Using the 'Odds and evens' problem as a springboard, students construct interactive spreadsheets designed to address particular needs. This lesson also demonstrates an approach to programming known as rapid application development (RAD).<br>• **Seeing the wood for the trees** - using sample study habit data, students summarise data using advanced filtering and grouping |

| | | |
|---|---|---|
| **and address complex problems, and model processes, entities and their relationships using structured data (ACTDIP037)** | Computational Thinking<br>• I can identify a question I want to answer using data I find online or collect via survey.<br>• I can acquire relevant data sets to answer my inquiry question<br>• I can organise, manage and validate data.<br>• I can analyse data using relevant techniques and tools such as a spreadsheet<br>• I can use Python programming to analyse data provided in CSV.<br><br>Design Thinking<br>• I can present information considering both the appearance and functionality of information. | techniques, for example pivot tables in spreadsheets and aggregation functions in databases.<br>• **A spreadsheet's secret weapon** - In this lesson sequence, students will learn to use pivot tables which have been described as the most powerful tool within spreadsheets.<br>• **Who wants to be a millionaire?** - Using the well-known wheat/rice and chessboard problem as a resource, students use spreadsheets to simulate iteration and to solve problems.<br>• **Computer science in a box - Unplug your curriculum:** this resource booklet was produced to be used for students ages 9-14 to teach lesson about how computers work, while addressing critical maths and science concepts such as number systems, algorithms and manipulating variables and logic. |
| **Creating Digital Solutions**<br><br>**#6 Define and decompose real-world problems precisely, taking into account functional and non-functional requirements and including interviewing stakeholders to identify needs (ACTDIP038)**<br><br>**Creating Digital Solutions**<br><br>**#7 Design the user experience of a digital system by evaluating alternative designs against criteria including functionality, accessibility, usability, and aesthetics** | Creating a Digital Game (Level 9)<br>Computational Thinking<br>• I can decompose a problem into smaller parts, focusing on the important information<br>• I can define what the digital game is required to do and identify the functional and non-functional requirements<br>• I can view some code and identify where classes have been used following OOP principles.<br>• I can complete a table that describes how classes can be used to associate variables and functions<br>• I can describe how classes are used in object-oriented programming (OOP) language.<br>• I can remix an example code that has a way of structuring classes, and I can modify it with support to create a new program for a game<br>• I can complete an algorithm as a flowchart with support<br>• I can complete an algorithm as using structured English with support.<br><br>Design Thinking | • https://www.digitaltechnologieshub.edu.au/teachers/scope-and-sequence/9-10<br>• **Developing an Aboriginal seasonal database** - Leveraging the Year 10 Geography curriculum, this sequence works with the CSIRO indigenous seasons calendars. Students produce a searchable database that will capture data using the two data sources.<br>• **Design and Deliver** - students create a website that acts as a showcase for a portfolio of their digital work. They learn about flexible design and how to design a site that can be viewed on a browser using any size of screen.<br>• **Fibonacci Served Three Ways** - students learn to code separate modules that perform discrete functions but collectively meet the needs of the solution. They select the most appropriate algorithm based on the type of problem.<br>• **Behaving with real class** - One challenge in teaching object-oriented principles is finding a suitable programming language. Many of these languages are too complex and their |

| | | |
|---|---|---|
| **(ACTDIP039)**<br><br>**Creating Digital Solutions**<br><br>**#8 Design algorithms represented diagrammatically and in structured English and validate algorithms and programs through tracing and test cases (ACTDIP040)**<br><br>**Creating Digital Solutions**<br><br>**#9 Implement modular programs, applying selected algorithms and data structures including using an object-oriented programming language (ACTDIP041)**<br><br>**Creating Digital Solutions**<br><br>**#10 Evaluate critically how student solutions and existing information systems and policies, take account of future risks and sustainability and provide opportunities for innovation and enterprise (ACTDIP042)** | • I can create an algorithm to describe the flow of interactions for the design of a digital game<br>• I can implement OOP principles using a relevant programming language such as Python<br>• I can incorporate graphics to improve user interface using a library such as PyGame when programming in Python<br>• I can evaluate the usefulness of the programming and the interface of the digital solution.<br><br><u>Robotics and Embedded Systems (Level 10)</u><br><br>Systems Thinking<br>• I can define a robot<br>• I can identify situations where automation would be useful<br>• I can identify parts of an electrical circuit such as sensors, actuators and a development board<br><br>Computational Thinking<br>• I can combine components in an electrical circuit including a development board<br>• I can program a development board following a tutorial and using existing code, for example a sketch for Arduino.<br>• I can program a development board that uses data to trigger a certain behaviour<br>• I can use a robotic kit to build a robot to carry out a specific task.<br><br>Design Thinking<br>• I can design and create my own robotic device using electronic components and other materials (recycled or commercially available)<br>• I can evaluate the digital solution I created, based on criteria such as sustainability, innovation and/or enterprise. | environments too confusing. This lesson sequence offers a choice of one of two approaches in an attempt to address this problem.<br><br>• **Breaking up can be good** - this sequence provides a gentle introduction to the skill of decomposition by having students develop discrete modules which together serve a single need: a maths teacher asks for a program that can be used to demonstrate aspects of maths. This sequence can be used in conjunction with 'Comparing and selecting appropriate algorithms'.<br><br>• **Getting Warmer** - This lesson sequence intentionally uses a visual based programming tool to introduce designing and validating algorithms. Those students who complete this task can move to code the result in any text based language with which they are familiar.<br><br>• **Table Tennis Server Challenge:** a set of printable resources and guidance notes give teachers the support to run a challenge to design a digital solution.<br>• **Robotics Peripheral Vision** - It leads students through a study of human vision and computer programming simulation. Students apply their previous knowledge of arrays and looping structures to implement a new concept of linked lists and RGB decomposition in order to solve the unit's Grand Challenge: writing a program to simulate peripheral vision by merging two images.<br><br>• **CSER Videos and Activities**<br><br>**Apps and Software**<br>• **Alice (free, web):** 3D programming environment which enables students to create an animation for telling a story, playing and interactive game, or a video to share on the web. Alice teaches visually and is more entertainment oriented. It can provide students first exposure to object –oriented programming. Find |

**Creating Digital Solutions**

**#11 Create interactive solutions for sharing ideas and information online, taking into account safety, social contexts and legal responsibilities (ACTDIP043)**

**Data and Information**

**#5 Plan and manage projects using an iterative and collaborative approach, identifying risks and considering safety and sustainability (ACTDIP044)**

Managing a Group Project: Augmented Reality (Level 9)

*Collaboration: Planning and Managing*

Systems Thinking

- I can identify our team roles
- I can identify the main stages of the plan to create our digital solution.
- I can describe my role within the team and what I'm responsible for.
- I can describe what each team role is and who is responsible for which task.
- I can describe the stages of our plan.

Computational Thinking

- I can work as a team and demonstrate my contribution to the group's project deliverables.
- I can relate stages of the plan with our team's roles and responsibilities.

Design Thinking

- I can evaluate my performance in the group based on my role and responsibilities and provide peer feedback to others on my team and relate this to the success of the project.

*Augmented Reality*

Systems Thinking

- I can identify an AR feature.
- I can explain the difference between AR and VR.
- I can describe potential benefits of AR for individuals or groups of people.
- I can install and use apps onto a smartphone or tablet.

Computational Thinking

- I can consider user scenarios and design an AR activity that would add benefit for the user.

teaching support materials on the website including step by step of how to get started. Free download of the software. It is also worth checking out other self-study programs here.

- **Greenfoot (free, web):** Object-Oriented programming in Java, with games and simulations. Uses visual tools coupled with programming, making learning to program easier. Runs on PC, Mac & Linux. Real programming, learning and teach Java and share with user communities.
- **CodingBat (free, web):** live coding problems to build coding skill in Java and Python, created by Nick Parlante at Stanford. The coding problems give instant feedback giving students an opportunity to practice and build understanding of the concepts. Intermediate level.
- **Code Combat (free, web):** the game like platform with the premise that they are stuck in a dungeon surrounded by enemies or racing across a battlefield to conquer a foe. This application puts players in control of their character through programming, using text based Java script.
- **Grok Learning (free introduction to programming, paid subscriptions, web):** introductory courses using the programming language Python for people with no programming experience. Anonymous accounts disappear after 24 hours, so students will need to sign in to save their program.
- **Python (free, computer software Windows, Mac and other):** Python is a programming language which can be used for first time or experienced programmers. There are lots of educational books to support you and your class to get started or use the helpful getting started guides on the website. For further information
- **CS Field Guide (free, web):** this open sourced online textbook resource is currently mapped to the New Zealand Computer Science curriculum. There are two versions of the online textbook, the student versions and the teacher version.
- **Ruby (free, Computer software, Windows, Mac, Linux):** Ruby is a dynamic, open source programming language. Explore the

| | | |
|---|---|---|
| | Design Thinking<br>• I can evaluate an AR environment and describe its usefulness.<br>• I can create an AR activity that provides an enhancement to the reality for the end user.<br><br>Collaborative Project (Level 10)<br>*Collaboration: Planning and Managing*<br>Computational and Systems Thinking<br>• I can identify our team roles<br>Computational Thinking<br>• I can describe my role within the team and what I'm responsible for.<br>• I can describe what each team role is and who is responsible for which task.<br>• I can work as a team and demonstrate my contribution to the group's project deliverables.<br>Design Thinking<br>• I can evaluate my performance in the group based on my role and responsibilities and provide peer feedback to others on my team and relate this to the success of the project.<br><br>*Media campaign and associated media content*<br>Computational and Systems Thinking<br>• I can identify an issue that matters to me<br>• I can identify the key steps for our media plan and help define requirements.<br>Computational Thinking<br>• I can describe the stages of our plan.<br>• I can compare and contrast different types of media both | many documents and tutorials designed to support teachers and students to learn Ruby.<br>• **Raspberry Pi:** credit card sized programming board to help students to learn how to program a computer. Plug in a mouse and a monitor and learn how to program it. Cross curriculum links. Students can take advantage of the size, cost and portability, i.e. My Robot.<br>• **GameMaker (basic package free, web):** GameMaker offers a game maker studio for novice and professional game developers, allowing them to create cross-platform games.<br>• **Khan Academy Computer Science**<br>• **Bootstrap (free, web):** computing creatively and connecting mathematics. With the aim of exciting students about gaming while directly applying algebra to create something. Two main programs of learning with clear curriculum links and lesson plans for teachers. Aimed at students aged 12-16.<br>• **Hot or not?** - Focussing on the incremental or Agile approach to development, students following the evolution of a temperature conversion tool for a Food Technology teacher. It uses an Excel spreadsheet application as the prototyping tool.<br>• **The shock of the new** - Using 4 inventions from 1985, this sequence explores the impact of innovation, supporting circumstances, how individuals contribute to change and the importance of addressing benefits as well as risks in the development of new systems.<br>• **Future Proofing Data** - This sequence aims to identify strengths and weaknesses of past, present and future methods of data storage and recognise the risks and benefits for users. Students explore specific "data dilemmas"<br>• **Technology Benefits**: this lesson makes students aware of the positives that modern communication systems give to society, and asks them to make decisions about the use of these systems for social, economic and environmental purposes. |

| | | |
|---|---|---|
| | traditional and new.<br>• I can consider our audience and demonstrate how we are meeting their needs.<br>• I can demonstrate how our choice of media aligns to our project goals.<br><br>Design Thinking<br>• I can create content for which I was given responsibility and consider format and conventions particular to the technology.<br>• I can evaluate the content I created and reflect on the process | |

**Level 9 to Level 10 Achievement Standard**

By the end of Year 10, students explain the control and management of networked digital systems and the security implications of the interaction between hardware, software and users. They explain simple data compression, and why content data are separated from presentation.

Students plan and manage digital projects using an iterative approach. They define and decompose complex problems in terms of functional and non-functional requirements. Students design and evaluate user experiences and algorithms. They design and implement modular programs, including an object-oriented program, using algorithms and data structures involving modular functions that reflect the relationships of real-world data and data entities. They take account of privacy and security requirements when selecting and validating data. Students test and predict results and implement digital solutions. They evaluate information systems and their solutions in terms of risk, sustainability and potential for innovation and enterprise. They share and collaborate online, establishing protocols for the use, transmission and maintenance of data and projects.

**Years 9 and 10 Band Description**

Learning in Digital Technologies focuses on further developing understanding and skills in computational thinking such as precisely and accurately describing problems and the use of modular approaches to solutions. It also focuses on engaging students with specialised learning in preparation for vocational training or learning in the senior secondary years.

By the end of Year 10, students will have had opportunities to analyse problems and design, implement and evaluate a range of digital solutions, such as database-driven websites and artificial intelligence engines and simulations.

In Year 9 and 10, students consider how human interaction with networked systems introduces complexities surrounding access to, and the security and privacy of, data of various types. They interrogate security practices and techniques used to compress data, and learn about the importance of separating content, presentation and behavioural elements for data integrity and maintenance purposes.

Students explore how bias can impact the results and value of data collection methods and they use structured data to analyse, visualise, model and evaluate objects and events.

They learn how to develop multilevel abstractions, identify standard elements such as searching and sorting in algorithms, and explore the trade-offs between the simplicity of a model and the faithfulness of its representation.

When defining problems students consider the functional and non-functional requirements of a solution through interacting with clients and regularly reviewing processes. They consolidate their algorithmic design skills to incorporate testing and review, and further develop their understanding of the user experience to incorporate a wider variety of user needs. Students develop modular solutions to complex problems using an object-oriented programming language where appropriate, and evaluate their solutions and existing information systems based on a broad set of criteria including connections to existing policies and their enterprise potential. They consider the privacy and security implications of how data are used and controlled, and suggest how policies and practices can be improved to ensure the sustainability and safety of information systems.

Students progressively become more skilled at identifying the steps involved in planning solutions and developing detailed plans that are mindful of risks and sustainability requirements. When creating solutions, both individually and collaboratively, students comply with legal obligations, particularly with respect to the ownership of information, and when creating interactive solutions for sharing in online environments.