Short Course
Coding

Specification for Junior Cycle

June 2016

Contents

Introduction to junior cycle

Junior cycle education places students at the centre of the educational experience, enabling them to actively participate in their communities and in society, and to be resourceful and confident learners in all aspects and stages of their lives. Junior cycle is inclusive of all students and contributes to equality of opportunity, participation and outcome for all.

Junior cycle allows students to make a strong connection with learning by focusing on the quality of learning that takes place and by offering experiences that are engaging and enjoyable for them, and relevant to their lives. These experiences are of a high quality, contribute to the physical, mental and social wellbeing of learners, and where possible, provide opportunities for them to develop their abilities and talents in the areas of creativity and enterprise. The student's junior cycle programme builds on their learning in primary school. It supports their further progress in learning. It helps students to develop the learning skills that can assist them in meeting the challenges of life beyond school.

Rationale

Computer science is present in every aspect of modern society. Correctly-functioning software systems allow airplanes to fly from one city to another; give out money at the ATM and diagnose the level of glucose in your blood. Fundamental understanding of how computer hardware and software operate and relate to everyday life is an increasingly important area of learning for students. Problem solving and computational thinking skills are developed in this course as students build and create software projects using their own ideas and imagination. The course looks to build on any coding skills that primary students might have acquired while offering insight into possible future studies in computer science and software engineering.

Aim

The short course aims to develop the student's ability to formulate problems logically; to design, write and test code through the development of programs, apps, games, animations or websites; and, through their chosen learning activities, to learn about computer science.

Overview: Links

Tables 1 and 2 on the following pages show how coding may be linked to central features of learning and teaching in junior cycle.

Coding and statements of learning

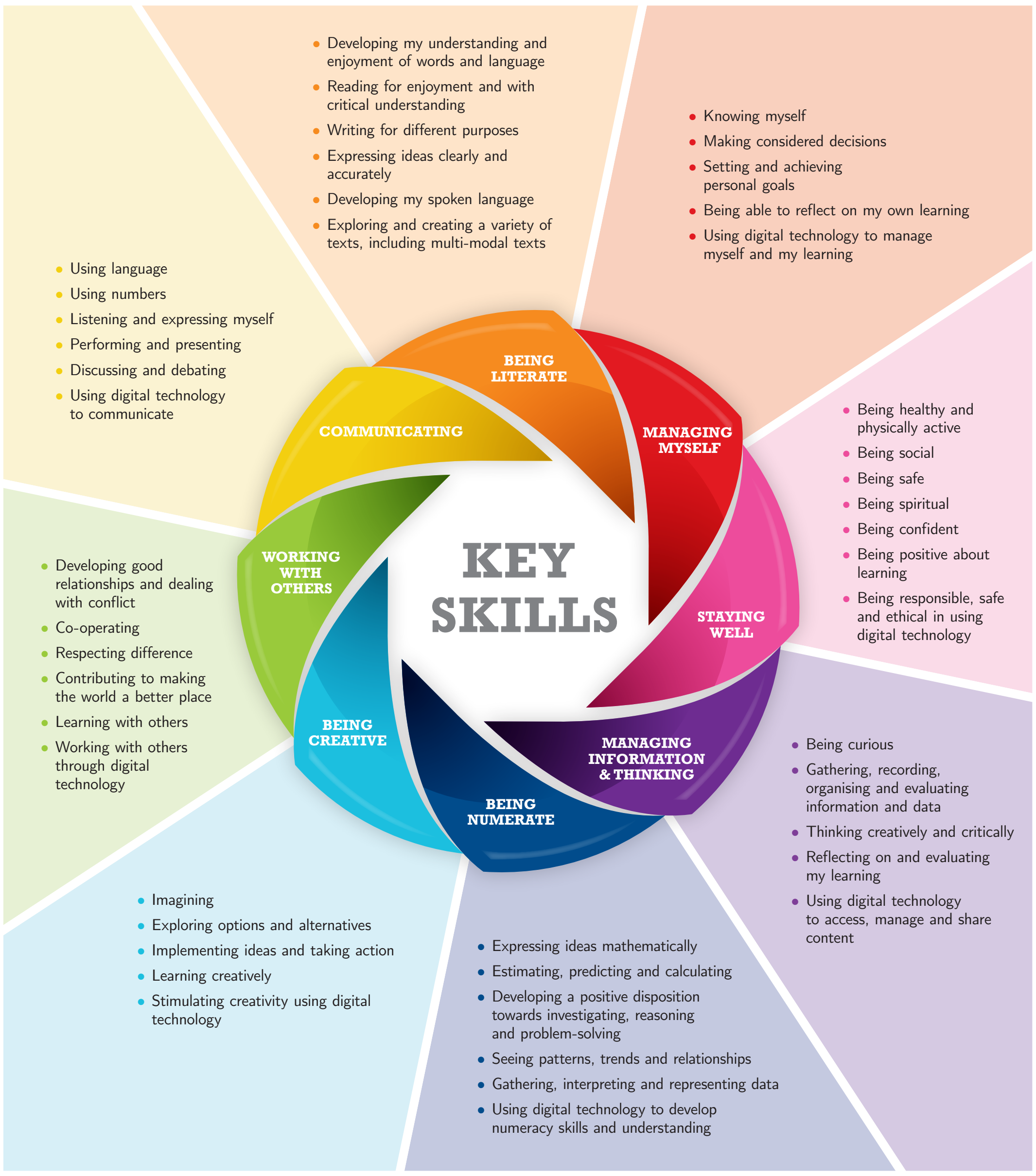
Table 1: Links between junior cycle coding and the statements of learning

Statement	Examples of related learning in the course
SOL 17: The student devises and evaluates strategies for investigating and solving problems using mathematical knowledge, reasoning and skills.	Problem solving and computational thinking are central to this course. Students use their mathematical knowledge, skills and understanding when figuring out, evaluating and implementing solutions to particular problems.
SOL 16: The student describes, illustrates, interprets, predicts and explains patterns and relationships.	Students interpret and describe patterns and relationships as they solve problems and create projects using algorithms and programming languages.
SOL 20: The student uses appropriate technologies in meeting a design challenge.	Students, either individually or as part of a team, research and discuss the most appropriate technologies to use, to solve problems and deliver solutions.
SOL 23: The student brings an idea from conception to realisation.	Students engage in brainstorming and planning activities, move on to the design, development and test phases, culminating in the creation of a project solution to a particular problem.

Coding and key skills

In addition to their specific content and knowledge, the subjects and short courses of junior cycle provide students with opportunities to develop a range of key skills. The junior cycle curriculum focuses on eight key skills.

Figure 1: Key skills of junior cycle


KEY SKILLS
BEING LITERATE
- Developing my understanding and enjoyment of words and language
- Reading for enjoyment and with critical understanding
- Writing for different purposes
- Expressing ideas clearly and accurately
- Developing my spoken language
- Exploring and creating a variety of texts, including multi-modal texts
MANAGING MYSELF
- Knowing myself
- Making considered decisions
- Setting and achieving personal goals
- Being able to reflect on my own learning
- Using digital technology to manage myself and my learning
STAYING WELL
- Being healthy and physically active
- Being social
- Being safe
- Being spiritual
- Being confident
- Being positive about learning
- Being responsible, safe and ethical in using digital technology
MANAGING INFORMATION & THINKING
- Being curious
- Gathering, recording, organising and evaluating information and data
- Thinking creatively and critically
- Reflecting on and evaluating my learning
- Using digital technology to access, manage and share content
BEING NUMERATE
- Expressing ideas mathematically
- Estimating, predicting and calculating
- Developing a positive disposition towards investigating, reasoning and problem-solving
- Seeing patterns, trends and relationships
- Gathering, interpreting and representing data
- Using digital technology to develop numeracy skills and understanding
BEING CREATIVE
- Imagining
- Exploring options and alternatives
- Implementing ideas and taking action
- Learning creatively
- Stimulating creativity using digital technology
WORKING WITH OTHERS
- Developing good relationships and dealing with conflict
- Co-operating
- Respecting difference
- Contributing to making the world a better place
- Learning with others
- Working with others through digital technology
COMMUNICATING
- Using language
- Using numbers
- Listening and expressing myself
- Performing and presenting
- Discussing and debating
- Using digital technology to communicate

This course offers opportunities to support all key skills, but some are particularly significant. The examples below identify some of the elements that are related to learning activities in coding. Teachers can also build many of the other elements of particular key skills into their classroom planning.

Table 2: Links between junior cycle coding and key skills

Key skill	Key skill element	Student learning activity
Being creative	Implementing ideas and taking action	Students brainstorm and generate ideas for design and implementation of solutions and projects.
Being literate	Expressing ideas clearly and accurately	Students create a website to display the results of a topic they have researched.
Being numerate	Developing a positive disposition towards investigating, reasoning and problem-solving	Students create short programs which demonstrate their use and understanding of mathematical and computational ideas.
Communicating	Discussing and debating	Students discuss ideas, evaluate the pros and cons of different approaches, and propose possible solutions. They report on projects and provide feedback to others.
Managing information and thinking	Thinking creatively and critically	Students explore new and different ways of answering questions and solving problems. They use a variety of tools to access, manage and share information such as flow-charts, design documents, code documentation and bug lists.
Managing myself	Setting and achieving personal goals	Students take responsibility for personal learning by setting goals and seeking help when necessary from classmates, the teacher or other appropriate sources, and by reflecting on the feedback they receive.
Staying well	Being safe	Students become aware of the wellness, health and safety issues associated with working with computers, and of practical ergonomic issues regarding the use of computers. Students learn to be responsible, safe and ethical in using digital technology.
Working with others	Cooperating	Students develop good working relationships with others and appreciate the value of respect and cooperation in reaching both collective and personal goals. They learn to appreciate diverse talents and how to engage in collaborative work.

Overview: Course

The specification for this junior cycle short course in coding focuses on developing students' problem-solving skills through three inter-connected strands: **Computer science introduction**, **Let's get connected** and **Coding at the next level**.

Strand 1: Computer science introduction.

In this strand, students explore the range of uses computers have in today's world and learn to understand the hardware and basic software which operates them. This includes learning to write, test and evaluate code.

Strand 2: Let's get connected.

This strand deepens the student's understanding of the computer as a communications tool through the storage and manipulation of data. Students also have the opportunity to identify, research, present and receive feedback on a topic or challenge in computer science that inspires them.

Strand 3: Coding at the next level.

In this strand, students are introduced to more complex levels of coding where they can demonstrate their understanding through documentation, discussion and feedback.

This course is designed to be followed logically through strands 1–3. This enables students to build on the skills they have previously learned. Strand 1 is a basic, introductory strand and should be undertaken first. Schools may wish to approach the other two strands in the order they deem best for students.

Teamwork is encouraged throughout all three strands. Students collaborate, peer-explain, seek feedback, provide feedback and reflect on their work. Practical, hands-on and problem-solving learning activities should be in evidence across all strands of the course. Theoretical concepts can be reinforced through practical work and projects.

Free and open-source software should be used where practical, both to make software tools as widely available to students as possible and so that students have the opportunity to examine the source code of the tools they use.

In the context of their health and wellbeing, student safety in the use of computers is emphasised in the Statements of learning and should be a feature of the experience of students taking the course. There is a further NCCA short course available on digital media literacy which addresses internet safety concerns and responsible use in more detail. Students should also be made aware of the acceptable use policy of their own school.

The Classroom-Based Assessment outlined below reflects the learning students undertake in this NCCA short course. Schools have the flexibility to adapt any NCCA short course to suit their particular needs and school context. If adapting the course, schools may also need to adapt the Classroom-Based Assessment, so that it reflects the learning their students undertook. Schools may also develop their own short course(s) and related Classroom-Based Assessment. Guidelines for schools who wish to develop their own short course(s) are available.

The learning outcomes in this short course are aligned with the level indicators for Level 3 of the National Framework of Qualifications (Appendix 1).

The course has been designed for approximately 100 hours of student engagement.

Expectations for students

Expectations for students is an umbrella term that links learning outcomes with annotated examples of student work. For NCCA-developed short courses, in some cases, examples of work associated with a specific learning outcome or with a group of learning outcomes will be available. Schools who design their own short courses may wish to create a bank of examples of student work for discussion and for future reference.

Learning outcomes

Learning outcomes are statements that describe what knowledge, understanding, skills and values students should be able to demonstrate having completed this junior cycle short course in coding. The learning outcomes set out in the following tables apply to all students and represent outcomes for students at the end of their period of study (approximately 100 hours).

The outcomes are numbered within each strand. The numbering is intended to support teacher planning in the first instance and does not imply any hierarchy of importance across the outcomes themselves.

Strand 1: Computer science introduction

Learning outcomes

Students learn about	Students should be able to
My digital world: The importance of computers in modern society and my life	1.1 present and share examples of what computers are used for and discuss their importance in modern society and in their lives
	1.2 describe the main components of a computer system (CPU, memory, main storage, I/O devices, buses)
	1.3 explain how computers are devices for executing programs via the use of programming languages
Being a coder–step by step: How to start programming and to develop basic algorithms	1.4 develop appropriate algorithms using pseudo-code and/or flow charts
	1.5 write code to implement algorithms
	1.6 discuss and implement core features of structured programming languages, such as variables, operators, loops, decisions, assignment and modules
	1.7 test the code
	1.8 evaluate the results in groups of two or three

Strand 2: Let’s get connected

Learning outcomes

Students learn about	Students should be able to
Making connections: Computers are communication devices	2.1 discuss the basic concepts underlying the internet
	2.2 describe how data is transported on the internet and how computers communicate and cooperate through protocols
	2.3 explain how search engines deliver results
	2.4 build a website using HTML and CSS to showcase their learning
Bits and bytes: How computers store data	2.5 explain how computers represent data using 1’s and 0’s
	2.6 investigate how drawings and photos are represented in computing devices
Real world problems: Computer science inspiring me	2.7 identify a topic or a challenge in computer science that inspires them
	2.8 conduct research on the topic/challenge
	2.9 present a proposal for discussion and reflect on feedback
	2.10 convince their peers that an idea is worthwhile

Strand 3: Coding at the next level

Learning outcomes

Students learn about	Students should be able to
Being a coder: More advanced concepts in programming and computational thinking	3.1 creatively design and write code for short programming tasks to demonstrate the use of operators for assignment, arithmetic, comparison, and Boolean combinations
	3.2 complete short programming tasks using basic linear data structures (e.g. array or list)
	3.3 demonstrate how functions and/or procedures (definition and call) capture abstractions
	3.4 describe program flow control, e.g. parallel or sequential flow of control – language dependent
Documenting the code: Documentation and code analysis	3.5 document programs to explain how they work
	3.6 present the documented code to each other in small groups
	3.7 analyse code to determine its function and identify errors or potential errors

Assessment and reporting

Essentially, the purpose of assessment and reporting at this stage of education is to support learning. This short course supports a wide variety of approaches to assessment. Some learning outcomes lend themselves to once-off assessment, others to assessment on an ongoing basis as students engage in different learning activities such as discussing, explaining, researching, presenting, planning and taking action. In these contexts, students with their teachers and peers reflect upon and make judgements about their own and others' learning by looking at the quality of particular pieces of work. They plan the next steps in their learning, based on feedback they give and receive. Ongoing assessment can support the student in their learning journey and in preparing for the Classroom-Based Assessment related to this short course.

It is envisaged that students will provide evidence of their learning in a variety of ways, including digital media, audio recordings and written pieces.

Assessment is most effective when it moves beyond marks and grades and reporting focuses not only on how the student has done in the past but on the next steps for further learning. Student progress and achievement in short courses, both in ongoing assessments and in the specific Classroom-Based Assessment relating to this short course will be communicated to parents in interim reporting and in the Junior Cycle Profile of Achievement (JCPA). To support teachers and schools, an Assessment Toolkit is available online. Along with the guide to the Subject Learning and Assessment Review (SLAR) process, the Assessment Toolkit will include learning, teaching, assessment and reporting support material.

Classroom-Based Assessment

Classroom-Based Assessments are the occasions when the teacher assesses the students in the specific assessment(s) that are set out in the subject or short course specification. Junior cycle short courses will have one Classroom-Based Assessment. Where feasible, teachers of short courses will participate in learning and assessment review meetings.

Classroom-Based Assessment: Putting the pieces together

This Classroom-Based Assessment is the culmination of the work undertaken in the three strands of the coding short course. The Classroom-Based Assessment should begin after work in the three strands has been completed.

Students will develop a final software project of their choice in teams of two or three. They will research and establish requirements; design, implement and test the software. They will document their work and their code and present the project to their peers for review. They will reflect on feedback and also provide feedback on other students' projects.

Features of quality

The features of quality support student and teacher judgement of the Classroom-Based Assessments and are the criteria that will be used by teachers to assess students' final software projects.

More detailed material on assessment and reporting in this junior cycle coding short course, including features of quality and details of the practical arrangements related to assessment of this Classroom-Based Assessment, will be available in separate assessment guidelines for coding. The guidelines will include, for example, the suggested length and formats for student pieces of work, and support in using 'on balance' judgement in relation to the features of quality.

Inclusive assessment

Inclusive assessment practices, whether as part of ongoing assessment or the Classroom-Based Assessment, are a key feature of teaching and learning in schools. Accommodations, e.g. the support provided by a special needs assistant or the support of assistive technologies, should be in line with the arrangements the school has put in place to support the student's learning throughout the year.

Where a school judges that a student has a specific physical or learning difficulty, reasonable accommodations may be put in place to remove, as far as possible, the impact of the disability on the student's performance in the Classroom-Based Assessment.

Accommodations which enable all students to access curriculum and assessment are based on specific needs. For example, a student who cannot physically type may use free dictation software to complete ongoing assessments and the Classroom-Based Assessment. Equally, a student who cannot speak may sign/draw/write/type/create visuals and subtitles to present and communicate ideas. A student with a specific learning difficulty may benefit from having learning tasks and activities presented in a different way. Comprehensive guidelines on inclusion in post-primary schools are available here and guidelines for teachers of students with general learning disabilities are available here.

Appendix 1:

Level indicators for Level 3 of the National Framework of Qualifications

This short course has been developed in alignment with the level indicators for Level 3 of the National Framework of Qualifications. Usually, for Level 3 certification and awards, the knowledge, skill and competence acquired are relevant to personal development, participation in society and community, employment, and access to additional education and training.

NFQ Level	3
Knowledge *Breadth*	Knowledge moderately broad in range
Knowledge *Kind*	Mainly concrete in reference and with some comprehension of relationship between knowledge elements
Know-how and skill *Range*	Demonstrate a limited range of practical and cognitive skills and tools
Know-how and skill *Selectivity*	Select from a limited range of varied procedures and apply known solutions to a limited range of predictable problems
Competence *Context*	Act within a limited range of contexts
Competence *Role*	Act under direction with limited autonomy; function within familiar, homogeneous groups
Competence *Learning to learn*	Learn to learn within a managed environment
Competence *Insight*	Assume limited responsibility for consistency of self-understanding and behaviour

NCCA

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

AN ROINN
OIDEACHAIS AGUS SCILEANNA

DEPARTMENT OF
EDUCATION AND SKILLS