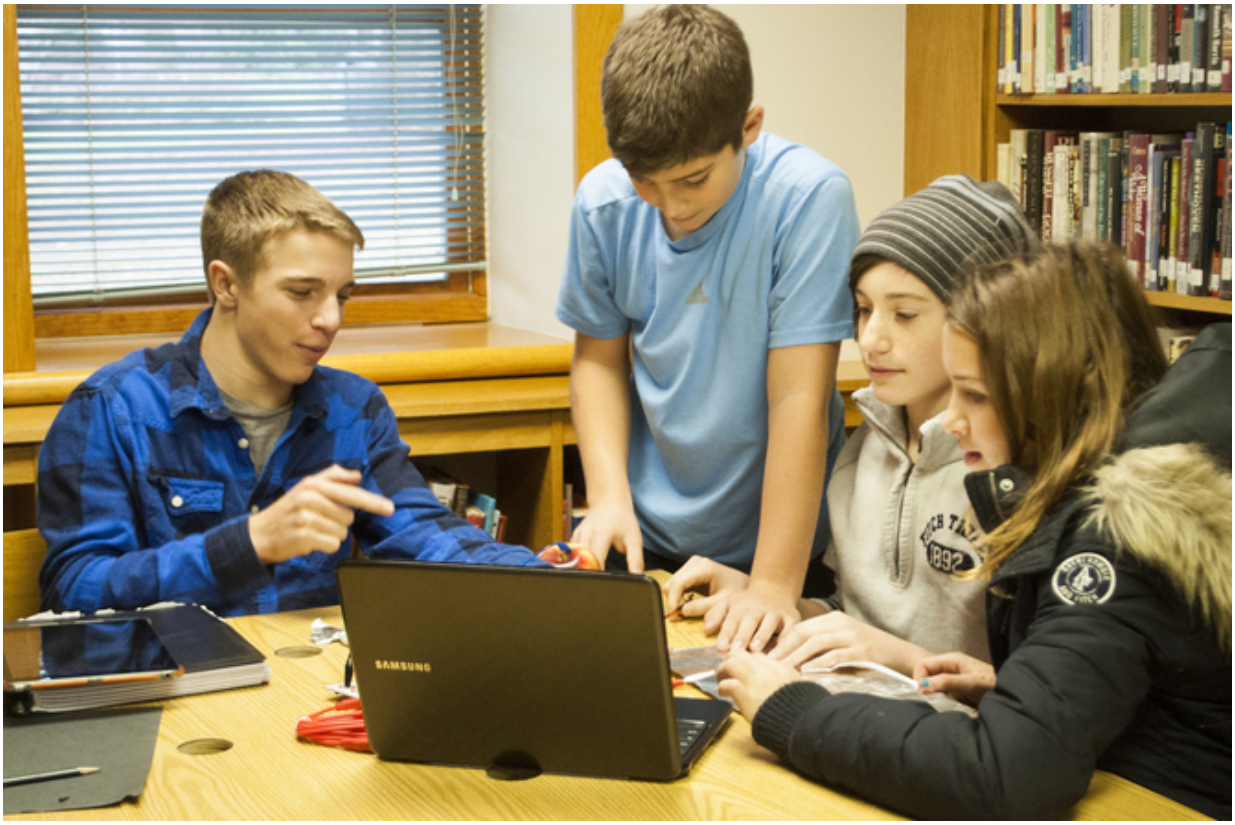
An education for the 21st century means teaching coding in schools

May 21 2015, by Leon Sterling



Programs like Hour of Code introduce computer programming to students in an engaging manner. Credit: Hour of Code 2014/Flickr, CC BY-NC-ND

Bill Shorten's recent announcement that, if elected, a Labor Government would "ensure that computer coding is taught in every primary and

secondary school in Australia" has brought attention to an increasing world trend.

Estonia introduced coding in primary schools in 2012 and the UK followed suit last year. US-led initiatives such as Code.org and the "Hour of Code", supported by organisations such as Google and Microsoft, advocate that every school student should have the opportunity to learn computer coding.

There is merit in school students learning coding. We live in a digital world where computer programs underlie everything from business, marketing, aviation, science and medicine, to name several disciplines. During a recent presentation at a radio station, one of our hosts said that IT would have been better background for his career in radio than journalism.

There is also a strong case to be made that Australia's future prosperity will depend on delivering advanced services and digital technology, and that programming will be essential to this end. Computer programs and software are known to be a strong driver of productivity improvements in many fields.

Being introduced to coding gives students an appreciation of what can be built with technology. We are surrounded by devices controlled by computers. Understanding how they work, and imagining new devices and services, are enhanced by understanding coding.

Of course, not everyone taught coding will become a coder or have a career in information technology. Art is taught in schools with no expectation that the students should become artists.

Drag and drop

A computer program is effectively a means of automating processes. Programs systematically and reliably follow processes and can be used to exhaustively try all the possibilities.

The languages used to program computers have evolved in the 70 years we have been building computers. Interfaces and programming environments have become more natural and intuitive. Language features reflect the applications they're used for.

What is needed to easily express a business process, scientific equation, or data analysis technique is not necessarily the same as what is needed to rapidly develop a video game.

However, throughout the evolution of programming languages, the fundamental principles have remained the same. Computer programming languages express three essential things:

1. The order in which a sequence of instructions is performed
2. A means of repeating a sequence of instructions a prescribed number of times
3. And tests as to whether or not a sequence of instructions is performed.

While personal preference influences which computer language a programmer uses, there is a greater understanding of which languages work well for teaching introductory programming. For example, Scratch is popular for primary school students and is quick to learn. Alice has been used to help students quickly build computer animations. Python is increasingly used for scientific applications. Visual programming languages – where students can drag-and-drop icons rather than type code – allow for rapid development of simple programs.

At Swinburne University of Technology we run workshops to introduce

school students to program NAO robots. Students use the Choregraphe environment to link robot actions from a library.

Students previously unused to programming can develop interesting robot projects in a couple of days. More sophisticated development of the robot requires students to use a more detail-oriented language, such as Python or C++. The simpler options lead to positive student experience.

Computational thinking

Writing and then executing a program gives immediate feedback as to whether you have correctly expressed instructions for the computer. Ultimately, the understanding of how to express concepts so that a computer can perform tasks accurately and efficiently is far more important than the details of the programming language.

Underlying all computer programs are algorithms, which specify in a more abstract way how a task is to be done. Algorithmic thinking – also called computational thinking – underlies computer science, and there has been a growing movement on algorithmic thinking in schools.

The new national curriculum reflects algorithmic processes, and materials are being developed to help teachers with the new curriculum. Victoria has recently developed a new subject for the Victorian Certificate of Education (VCE) entitled Algorithmics.There are even materials for teaching algorithmic thinking without computers. The Computer Science Unplugged movement, led by Tim Bell and colleagues at the University of Canterbury, has developed resources that teach students concepts through movement and fun activities.

Teaching for the this century

Teaching computer coding in schools is very different from initiatives that advocate for computers in the classroom. I was not, and am still not, supportive of compulsory laptop programs in schools.

The idea is not necessarily to expose students to the technology itself, which is almost inevitable these days with the wide penetration of mobile phones. Rather, students are exposed to the skills needed to develop computer applications.

While IT skill shortages is a contentious topic, there is no doubt that not enough of the best and brightest are studying computer science at university. A significant factor is insufficient exposure to the topic at schools. Teaching coding at schools is aimed at addressing the lack.

It might be said that whatever programming language is taught will be obsolete by the time the students enter the workforce. My experience is that, if taught properly, students can rapidly transfer the principles of one language to another.

In the 19th and 20th centuries, the challenge was to understand the physical world, and harness force and energy. This understanding percolated into the school curriculum. In the 21st century, the challenge is to understand and harness data, information and knowledge. Computer programming is a necessary way of introducing students to these concepts.

Source: The Conversation