

# 情報教育/プログラミング教育は コンピューショナルシンキングとメイキングが キーワードみたいかな？



情報教育やプログラミング教育に関して、過去に作ったpptのスライドやFBの投稿画像をかき集めてみました。

内容は私見ですから

寄せ集めスライドなので、私、京町セイカもいろいろなバージョンです。

© Go Ota,  
gohome@v006.vaio.ne.jp



実際にプログラミング教育としてどのようなことがおこなわれているか見ていきましょう。

**メイキング**の解説あります



## 「スモールステップ」と「メイキング」

文部科学省のプログラミング教育実践ガイドで

「特に、教員は目標を細分化し、小さな目標を達成する体験を積み重ねながら最終目標に近づけるようスモールステップで課題を設定することで、児童生徒の『プログラミングは難しい』という思い込みを払拭させ、自分にもできるという自己効力感を高めさせているようです」



もう一つの選択肢 **メイキング**

自由にプログラミングを学習させる  
プログラムで物を作ることを楽しむ

3

## メイキングの考え方

理論背景	構成主義(ピアジェ) -> 構築主義(パパート)
課題:設問	簡潔でゆるく、評価に無縁
課題の アプローチ	ティンカリング:いじくりまわす。自分の好み、方法で行っていく。
活動サイクル	考える->作る->改良する。スパイラルデザイン
解決手順	手順自体を教えるのではなく、それを見つける・活用する場を提供する。
教授と指導	できるだけ少なくする。但し、指導を否定しているわけではない。

メイキングって難しそうですね、でもすでに小学校の図画工作科の学習指導要領の領域「A表現」で“「表したいことを絵や立体，工作に表す」は、およそのテーマや目的をもとに作品をつくろうとすることから始まる。”ってすでに実践しているかも

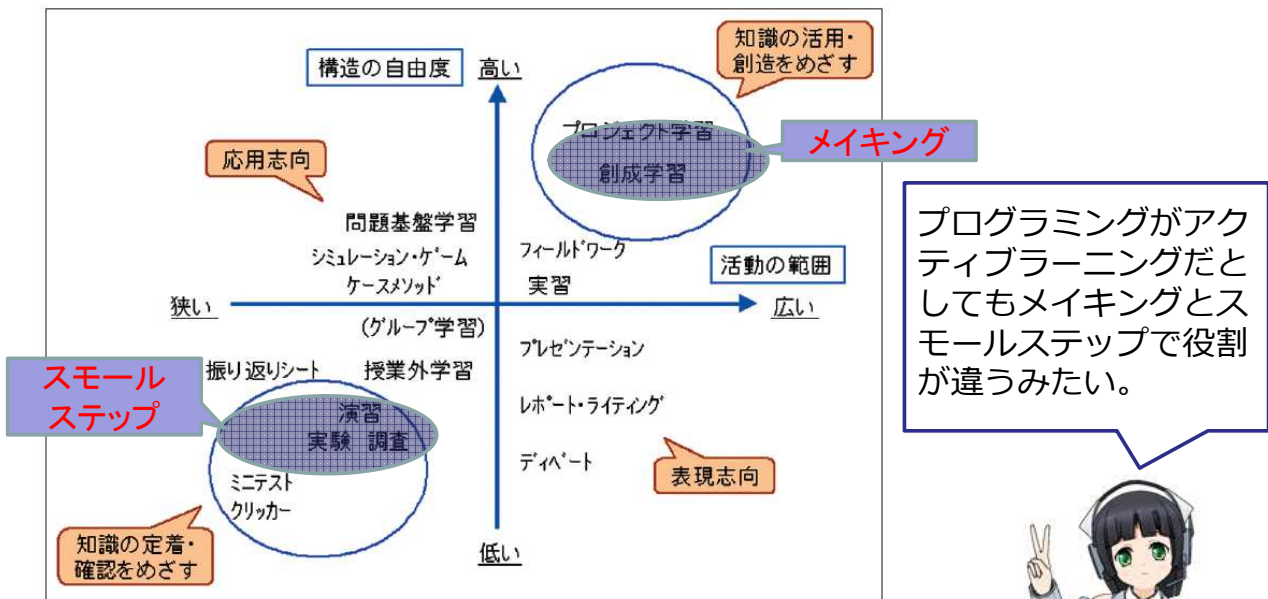


4

# プログラミングにおけるメイキング

- 近年はものづくりやメイキングムーブメントが教育の場でも重要視されてるが、メイキングは、単に3Dプリンターやセンサー・ロボットを授業に導入する以上に大きな教授法の方向展開であり、Papert(1986)で指摘しているように、単なる操作できる教具を与えるだけでなく、学習者が何か意味あるものを作り出す時に教育は最も効果があることを意味する。
- プログラミング教育の場面でも、兼宗、阿部、原田(2009)は「大切なのはその過程で自分で考えたこと、調べたこと、検証したこと、失敗したこと、成功したこと、判らなかったこと、分かったことである。」と述べている。
- 過去の失敗:1980年代や2000年代のSqueakEtoysが学校現場に普及しなかったのは、当時はまだ、このような学習観が十分受け入れられなかったと考えられる。また、兼宗、阿部、原田が「ブロックを適当に組み合わせて不規則に動く作品を子供の創造性と誤認しているものも見受けられる」と指摘しているように、教師の理解が不十分だった。

## プログラミング教育とアクティブラーニング



山地(2014)に太田が追記

**注意:** この図はメイキングとスモールステップの良し悪しを示しているものではありません。それぞれ、アクティブラーニングでも異なる性質であることを示しているだけです



# インフォーマルのプログラミング教育の現状(1)

## (1) Code.org/Hour of code (米国/全世界)

- ・世界中で数億人が参加しているみたい。
- ・スターウォーズやディズニーのキャラを使ったパズル型のチャートリアルが有名(これは**スモールステップ**かな)
- ・ちゃんとした、アンプラグドを含む授業用教材も充実

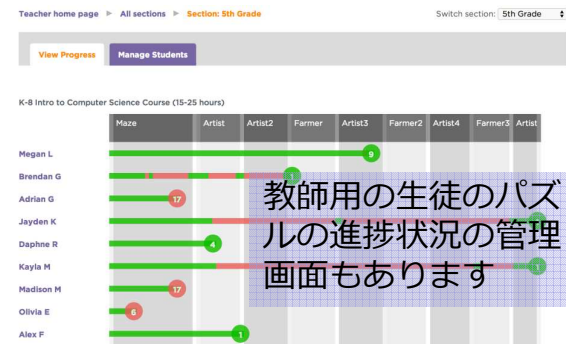


幼稚園／小学校教師向けの資料

コース1 授業計画 | 連続 | フレームワーク | 基準  
 コース2 授業計画 | 連続 | フレームワーク | 基準  
 コース3 授業計画 | 連続 | フレームワーク | 基準  
 コース4(ベータ): 授業計画

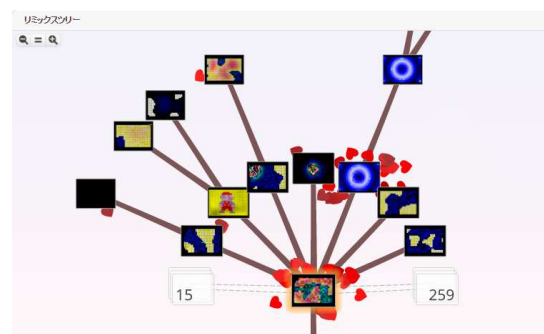
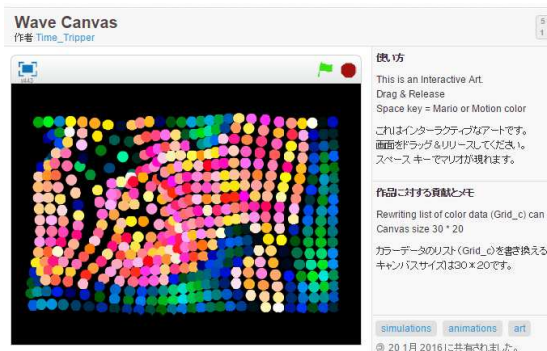
すべてのレッスンプランを表示します。

## CSTA K-12 Computer Science Standards

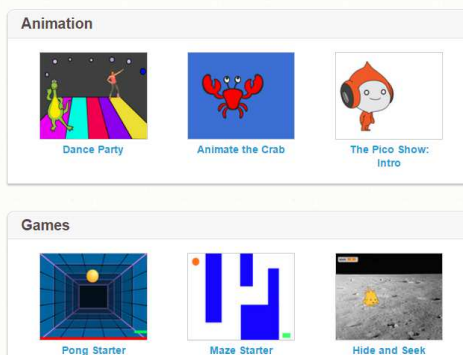


# インフォーマルのプログラミング教育の現状(2)

## (2) Scratchコミュニティ (米国/全世界)



リミックスツリー



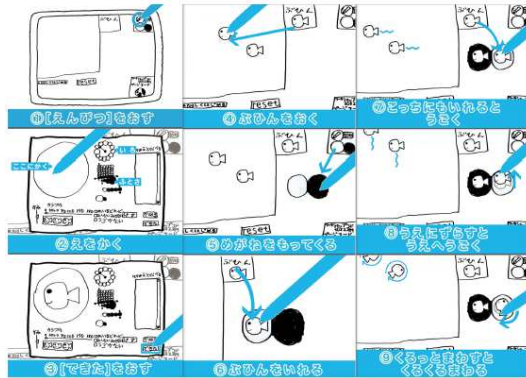
Starter Projects

- ・世界中で子供たちが使っているビジュアル言語
- ・Scratchコミュニティで全世界Connectingして**メイキング**を楽しんでみたい。
- ・ビジュアル言語としては**スモールステップ**でも使えます。

## インフォーマルのプログラミング教育の現状(3)

### (3)みどりっ子クラブ

- ・ 墨田区緑小学校の放課後支援活動の一つの活動(ビスケット使用)
- ・ 非IT専門家のKさんらが、がんばっています(メイキング実現のためのいろいろなノウハウがあります)



紙芝居: 操作説明



設計図

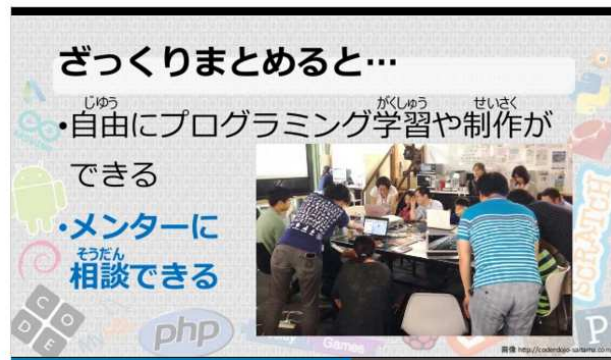
(図引用:原田,勝沼,久野 2014)

9

## インフォーマルのプログラミング教育の現状(4)

### (4)CoderDojo

- ・ 全世界で実施されている子供用プログラミングクラブ。
- ・ 無料であれば、自由に開催できる。日本でも多く地域で開催。
- ・ 自由なプログラミング学習の場: **メイキング**



これから、2014年から始まった英国の教科コンピューティングを見ていきましょう。その中で**コンピューティショナルシンキング**や**体系的な学習内容**を見てみましょう。



11

### 英国の教科コンピューティング:何を学習するか

(教科コンピューティング冒頭の言葉)

高い品質のコンピューティング教育は、生徒が**コンピューティショナルシンキング**と**創造性**を使うことによって、この世界を理解して変えることができるようになります。

(太田訳)

**コンピューティショナルシンキング**って新しい言葉ですね。ただ英国の新しい教科コンピュータはこれが重要みたい。



翻訳原文

National curriculum in England: computing programmes of study

<https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study/national-curriculum-in-england-computing-programmes-of-study>

12

## コンピューテショナルシンキングとは

コンピューテショナルシンキングはPaperが初めて使って、Wingが2006年のエッセイで広めたと言われています。ここでは、このエッセイがいくつかの言葉を拾ってみました。

コンピューテショナルシンキングは、コンピュータ科学者だけではなく、すべての人にとって基本的な技術である。すべての子供の分析的思考能力として、「読み、書き、そろばん（算術）」のほかにコンピューテショナルシンキングを加えるべきである。

コンピューテショナルシンキングとは巨大で複雑なタスクに挑戦したり、巨大で複雑なシステムをデザインしたりするときに、抽象化と分割統治を用いることである。それは問題点の分割である。それは問題の適切な表現法を選ぶことであり、問題を解きやすくするために問題の適切な側面だけをモデル化することである。

コンピュータ科学者のように考えるということは、コンピュータをプログラムできるといふこと以上の意味を持つ。複数のレベルの抽象思考が必要である

人間の思考法のことであり、コンピュータのそれではない。コンピューテショナルシンキングは人間の問題解決法であり、人間がコンピュータのように考えることを目指すものではない。

難しい.....



Wing 2006: <https://www.cs.cmu.edu/~15110-s13/Wing06-ct.pdf>  
 翻訳 <https://www.cs.cmu.edu/afs/cs/usr/wing/www/ct-japanese.pdf>

## 各国のコンピューテショナルシンキングの定義

UK Computer Progress Pathway	Australian Curriculum	CSTA(USA)
抽象化	抽象化	抽象化
デコンポジション		
一般化		モデルとシミュレーション
アルゴリズム思考	特定とアルゴリズム	アルゴリズム
		問題解決
	データの収集、表現と解釈	データ表現
評価	相互作用と影響	

いろいろな国でコンピューテショナルシンキングの内容で定義しています。



Computational Thinking (Wing, J. M., 2006)はいろいろな側面を説明していますが、厳密な定義はしていません、各国とも独自の解釈してますね。

## 補足: コンピューテショナルシンキングの定義

Google for Education	AP Computer Science Principles	UK Computer Progress Pathway	Austrarian Curriculum	CSTA
Abstraction	Abstracting	Algorithmic thinking	Abstraction	Problem solving
Algorithm Design	Connecting computing	Decomposition	Digital systems	Algorithms
Automation	Creating computational artifacts	Generalisation (Patterns)	Interactions and impacts	Data representation
Data Collection	Analyzing problems and artifact	Abstraction		Modeling and Simulation
Data Analysis	Communicating	Evaluation	Specification, algorithms and implementation	Abstraction
Data Representation	Collaborating		Data collection, representation and interpretation	Connection to other field
Decomposition				
Parallelization				
Pattern Generalization				
Simulation				

Googleもコンピューテショナルシンキング推しです。

15

## リソース: Quick Start Computing A CPD tool kit for Primary/ Secondary



↑ 資金



教員のCPD(Continuing Professional Development: 継続教育)用ツール・キット(64ページ + CD-ROM)  
Contents (Secondary teachers)  
Section 1: Leading effective CPD  
Section 2: Getting started with confidence  
Section 3: A road map for managing change  
Section 4: Teaching  
Section 5: Resources  
Section 6: Assessment and progression

CAS(Computing At School)の作成したこのTool Kitは英国のComputingを担当する教員にとってバイブルかもしれません。この一つにすべての情報と手順が入っています。



参考・引用 下記サイトより表示・ダウンロード可能  
「Quick Start Computing」  
<http://quickstartcomputing.org/>

16





# 初等中等教育での教科コンピューティング系統的な学習内容 2つのProgression Pathways

By Topics	By strands
<ul style="list-style-type: none"> <li>・アルゴリズム</li> <li>・プログラミングと開発</li> <li>・データとデータ表現</li> <li>・ハードウェアとソフトウェア</li> <li>・コミュニケーションとネットワーク</li> <li>・情報技術</li> </ul>	<ul style="list-style-type: none"> <li>・コンピュータサイエンス</li> <li>・情報技術</li> <li>・デジタルリテラシー</li> </ul>

CASでは、英国教育省の作成したカリキュラム(たった4ページ)をブレイクダウンして、初等中等教育での教科コンピューティングの学習内容を表形式で表したProgression Pathwaysを公開しています。一覧表になると小中高校で学習する内容の関係がはっきりしますね。なお、同一内容をBy TopicsとBy Strandsの二つの表にしています。



参考・引用 下記サイトより表示・ダウンロード可能  
 「Quick Start Computing」 Section 2: / What are the Progression Pathways?  
<http://quickstartcomputing.org/>



## Progression Pathways by Topics

KS1&2  
Grade 1-2  
Grade 3-6

KS3  
Grade 7-9

KS4  
Grade 10-11

Pupil Progression	Algorithms	Programming & Development	Data & Data Representation	Hardware & Processing	Communication & Networks	Information Technology
↓	<ul style="list-style-type: none"> <li>Understands what an algorithm is and is able to represent them through flowcharts, pseudocode, etc.</li> <li>Understands how computers work and how they are used to solve problems.</li> <li>Understands how algorithms are used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Knows that computers can do things that people can't do.</li> <li>Understands how programs are written and how they are used to solve problems.</li> <li>Understands how programs are used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that digital information is stored in binary code.</li> <li>Understands how data is represented and how it is used to solve problems.</li> <li>Understands how data is used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that computers have hardware and software.</li> <li>Understands how hardware and software are used to solve problems.</li> <li>Understands how hardware and software are used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that computers are connected to other computers.</li> <li>Understands how networks are used to solve problems.</li> <li>Understands how networks are used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that computers are used to solve problems.</li> <li>Understands how computers are used to solve problems.</li> <li>Understands how computers are used to solve problems.</li> </ul>
↓	<ul style="list-style-type: none"> <li>Understands that algorithms can be used to solve problems.</li> <li>Understands how algorithms are used to solve problems.</li> <li>Understands how algorithms are used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that computers can do things that people can't do.</li> <li>Understands how programs are written and how they are used to solve problems.</li> <li>Understands how programs are used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that digital information is stored in binary code.</li> <li>Understands how data is represented and how it is used to solve problems.</li> <li>Understands how data is used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that computers have hardware and software.</li> <li>Understands how hardware and software are used to solve problems.</li> <li>Understands how hardware and software are used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that computers are connected to other computers.</li> <li>Understands how networks are used to solve problems.</li> <li>Understands how networks are used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that computers are used to solve problems.</li> <li>Understands how computers are used to solve problems.</li> <li>Understands how computers are used to solve problems.</li> </ul>
↓	<ul style="list-style-type: none"> <li>Understands that algorithms can be used to solve problems.</li> <li>Understands how algorithms are used to solve problems.</li> <li>Understands how algorithms are used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that computers can do things that people can't do.</li> <li>Understands how programs are written and how they are used to solve problems.</li> <li>Understands how programs are used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that digital information is stored in binary code.</li> <li>Understands how data is represented and how it is used to solve problems.</li> <li>Understands how data is used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that computers have hardware and software.</li> <li>Understands how hardware and software are used to solve problems.</li> <li>Understands how hardware and software are used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that computers are connected to other computers.</li> <li>Understands how networks are used to solve problems.</li> <li>Understands how networks are used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that computers are used to solve problems.</li> <li>Understands how computers are used to solve problems.</li> <li>Understands how computers are used to solve problems.</li> </ul>
↓	<ul style="list-style-type: none"> <li>Understands that algorithms can be used to solve problems.</li> <li>Understands how algorithms are used to solve problems.</li> <li>Understands how algorithms are used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that computers can do things that people can't do.</li> <li>Understands how programs are written and how they are used to solve problems.</li> <li>Understands how programs are used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that digital information is stored in binary code.</li> <li>Understands how data is represented and how it is used to solve problems.</li> <li>Understands how data is used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that computers have hardware and software.</li> <li>Understands how hardware and software are used to solve problems.</li> <li>Understands how hardware and software are used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that computers are connected to other computers.</li> <li>Understands how networks are used to solve problems.</li> <li>Understands how networks are used to solve problems.</li> </ul>	<ul style="list-style-type: none"> <li>Understands that computers are used to solve problems.</li> <li>Understands how computers are used to solve problems.</li> <li>Understands how computers are used to solve problems.</li> </ul>

太田の雑な訳でよければ:  
[http://beyondbb.jp/Materials/UK\\_Pathways\\_TopicsJP\\_Draft0.9.pdf](http://beyondbb.jp/Materials/UK_Pathways_TopicsJP_Draft0.9.pdf)

参考・引用 下記サイトより表示・ダウンロード可能  
 「Quick Start Computing」 Section 2: / What are the Progression Pathways?  
<https://community.computingatschool.org.uk/resources/1692>



# Progression Pathwaysと コンピューショナルシンキング

## Progression Pathwaysの 学習内容(例)

- ・コンピュータは知能を持っていないことと、プログラムが無ければ何もできないことを理解する(AL)
- ・問題を分割し、個々の部分に対しての個別の解決方法を作ることによって、解決方法をデザインする。(DE)(AL)(AB)
- ・いくつかの問題が同様の特徴を共有し、それらを解決する同じアルゴリズムを使用することを認識する。(AL)(GE)
- ・問題解決の質を評価するため基準を使用する、そして将来的な解決と若干の改善を確認することができる。(EV)(GE)

能力	略語	概要
抽象化	AB	問題を単純化するため、重要な部分は残し、不要な詳細は削除する。
デコンポジション	DE	問題をいくつかの部分に理解や解決できるように分解する。
アルゴリズム	AL	問題を解決するための明確な手順で、同様な問題に共通して利用できるものである。
評価	AV	アルゴリズム、システムや手順などの解決方法が正しいか、確認する過程である。
一般化	GE	類似性からパターンを見つけて、それを予測、規則の作成、問題解決に使用する

学習内容が5つのコンピューショナルシンキングの能力のどれに対応するが示されています。例えば(AL)や(DE)等

プログラマ/SEだった人なら良く分かりますとおもうけど、問題解決能力や思考力がプログラマ的思考でうまく定義されているかと思います。

19



## Progression Pathways By strands

KS1&2

KS3

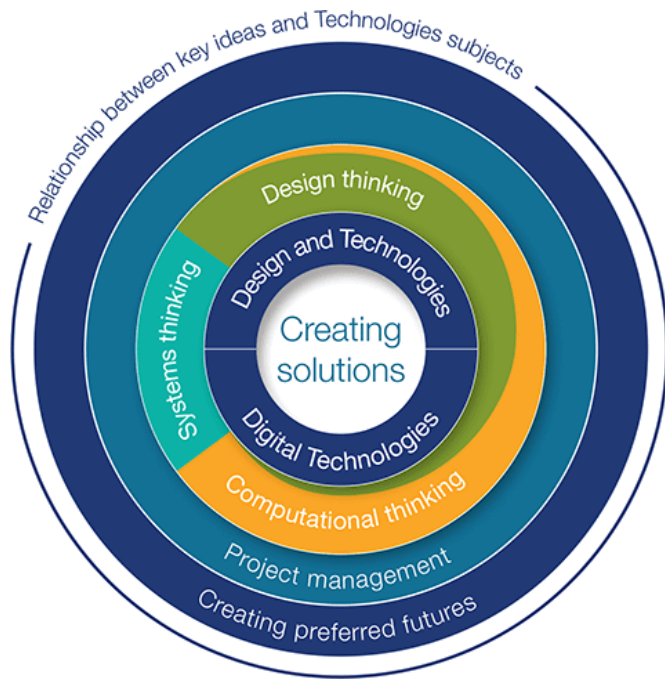
KS4

日本でも小学1年から高校3年までの、こんな2つのPathwaysがほしいですね。



20

# ちょっとだけ、2016年から始まる オーストラリアの新しい情報教育も



あまり国内で注目されていなかったオーストラリアが面白いかもしれない。英国に続き、全国レベルのプログラミング教育の導入というところもありますが教科テクノロジーの中に Design and Technologies科目と Digital Technologyがあるみたい前者が日本の技術・家庭科とほぼ同じみたいで、後者がプログラミング教育を含む情報教育

Digital Technologyの主な考えはやっぱり、**コンピューティショナルシンキング**。

## オーストラリアの初等中等の体系的な学習内容

Strand	Foundation to Year 2	Years 3 and 4	Years 5 and 6	Years 7 and 8	Years 9 and 10 (Elective subject)
Digital technologies: investigation and understanding	Identify, use and explore digital systems (hardware and software components) for a purpose (ACTDP001)	Explore and use a range of digital systems with peripheral devices for different purposes, and transfer different types of data (ACTDP007)	Investigate the main components of common digital systems, their basic functions and interconnections, and how such digital systems may connect together to form networks to transfer data (ACTDP014)	Investigate how data are transferred and secured in wired, wireless and mobile networks, and how the specifications of hardware components impact on network activities (ACTDP023)	Investigate the role of hardware and software in managing, controlling and securing the movement of and access to data in networked digital systems (ACTDP024)
Representation of data	Recognise and explore patterns in data and represent data as pictures, symbols and diagrams (ACTDP008)	Recognise different types of data and explore how the same data can be represented in different ways (ACTDP008)	Investigate how digital systems use whole numbers as a basis for representing all types of data (ACTDP015)	Investigate how digital systems represent text, images and audio data in binary (ACTDP024)	Analyse simple compression of data and how content data are secured from presentation (ACTDP025)
Collecting, managing and analysing data	Collect, explore and sort data, and use digital systems to present the data creatively (ACTDP003)	Collect, access and present different types of data using simple software to create information and solve problems (ACTDP009)	Acquire, view and validate different types of data, and use a range of commonly available software to interpret and visualise data in context to create information (ACTDP016)	Acquire data from a range of sources and evaluate authenticity, accuracy and trustworthiness (ACTDP022)	Develop techniques for acquiring, storing and validating quantitative and qualitative data from a range of sources, considering privacy and security requirements (ACTDP026)
Creating digital solutions by: Defining	Define simple problems, and describe and follow a sequence of steps and decisions (algorithms) needed to solve simple problems (ACTDP004)	Define simple problems, and describe and follow a sequence of steps and decisions (algorithms) needed to solve them (ACTDP010)	Define problems in terms of data and functional requirements, and identify resource needs to previously solved problems (ACTDP017)	Define and decompose real-world problems, taking into account functional requirements and economic, environmental, social, technical and usability constraints (ACTDP027)	Formally define and decompose real-world problems, taking into account functional and non-functional requirements and including interlocking subproblems to clarify needs (ACTDP028)
Designing			Design a user interface for a digital system, generating and considering alternative designs (ACTDP018)	Design the user experience of a digital system, generating, evaluating and communicating alternative designs (ACTDP020)	Design the user experience of a digital system, evaluating alternative designs against criteria including functionality, accessibility, usability, and aesthetics (ACTDP029)
Implementing		Implement simple digital solutions as visual programs with algorithms involving branching (decision), and user input (ACTDP003)	Implement digital solutions as simple visual programs involving branching, iteration (repetition), and user input (ACTDP020)	Design algorithms represented diagrammatically and in English, and trace algorithms to predict output for a given input and to identify errors (ACTDP021)	Design algorithms represented diagrammatically and in English, and trace algorithms to predict output for a given input and to identify errors (ACTDP021)
Evaluating	Explore how people safely use common information systems to meet information, communication and recreation needs (ACTDP005)	Explain how developed solutions and existing information systems meet common personal, school or community needs, and encourage new ways of using them (ACTDP012)	Explain how developed solutions and existing information systems meet sustainable and meet local community needs, considering resource needs and consequences for future applications (ACTDP013)	Evaluate how well developed solutions and existing information systems meet needs, are innovative, and take account of future risks and sustainability (ACTDP022)	Critically evaluate how well developed solutions and existing information systems and jobs, take account of future risks and sustainability and provide opportunities for evaluation and enterprise (ACTDP042)
Collaborating and managing	Work with others to create and organise ideas and information using information systems, and share these with known people in safe online environments (ACTDP006)	Work with others to plan the creation and communication of ideas and information safely, applying agreed ethical and social protocols (ACTDP013)	Manage the creation and communication of ideas and information including online collaborative projects, applying agreed ethical, social and technical protocols (ACTDP022)	Create and communicate interactive ideas and information collaboratively online, taking into account social contexts (ACTDP025)	Create interactive solutions for sharing ideas and information online, taking into account social contexts and legal responsibilities (ACTDP043)
			Plan and manage projects, including tasks, time and other resources required, considering safety and sustainability (ACTDP023)	Plan and manage projects using an iterative and collaborative approach, identifying risks and considering safety and sustainability (ACTDP044)	

オーストラリアの情報教育であるDigital Technologyも小学校から高校までの学習内容の一覧表がありますね。



参考・引用 下記サイトより表示・ダウンロード可能

<http://v7-5.australiancurriculum.edu.au/australian%20curriculum.pdf?Type=0&s=DI&e=ScopeSequence>

ちょっとだけ、各国カリキュラムの中のプログラミング教育の学習内容を見てみました。



## 各国のプログラミング教育の学習内容の体系的比較

	Grade 1	Grade 2	Grade 3	Grade 4	Grade 5	Grade 6	Grade 7	Grade 8	Grade 9	
英国 Computing Progression Pathways	KS1 Level 1 ・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	KS2 Level 2 ・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	KS2 Level 3 ・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	KS2 Level 4 ・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	KS2 Level 5 ・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	KS2 Level 6 ・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	KS2 Level 7 ・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	KS2 Level 8 ・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	KS2 Level 9 ・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	KS2 Level 10 ・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。
米国 CSTA K-12 Computer Science Fundamentals	・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。
オーストラリア 2015 Curriculum Framework for Computing	Grade 1-2 ・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	Grade 3-4 ・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	Grade 5-6 ・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	Grade 7-8 ・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。	Grade 9-10 ・コンピュータの基本的な機能やインストールの仕組みを説明できること。 ・簡単なアルゴリズムを記述し、実行させることができる。					

英国、オーストラリア、米国(CSTA)の情報教育のカリキュラムの中からプログラミング教育に関する学習内容を抽出し表にしてみました。



ちゃんとした大きなドキュメントはこちらから  
<http://beyondbb.jp/Materials/CompCuri.pdf>



## Computing is not equal coding & programming

Computingの主な学習内容がプログラミングであるとマスコミが報道しているのは間違っている

Computingとプログラミングは違うものである。ちょうどこの関係はScienceと実験の関係のようなものである。

Computingで学習することは基本的原理(fundamental principal)とコンピュテーショナルシンキング(computational thinking)である。プログラムはコンピュータとコンピュータを使用するための思考方法の間をつなぐものである。

CASのQuick Start Computingの中からプログラミングに関連した記述を抜き出してみました。日本でもマスコミはプログラミング教育と報道していますね(政府や文科省がプログラミング教育といているので仕方ないか?)



参考・引用 下記サイトより表示・ダウンロード可能  
「Quick Start Computing」 Section 2: / What are the Progression Pathways?  
<http://quickstartcomputing.org/>

## おまけページ: 既存のカリキュラム・教材

諸外国での初等中等学校でのプログラミング教育の状況 \* 計画中

国	初等	中等	補足 教科等(開始年)
英国	必(1-)	必(-11)	Computing(2015)
フィンランド*	必	必	クロスカリキュラム内(2016)
イスラエル		必(10-12)	Computer Science(2000)
ロシア	必(1-)	必(-11)	インフォマティカ(2010)
オーストラリア*	必	必	Design and Technologies Digital Technologies(2016)
シンガポール		必(専攻)	Computer Application
インド	必(3-	必(-12)	Computer Science(2005)
オランダ*	選	選	
デンマーク*		選	

英国とオーストラリアがまとまっていて、詳細の学習内容を分析中。だいたいのカテゴリーの抽出しているところ