Digital Technologies | Hangarau Matihiko

Level 6 – New Zealand Curriculum

# Teaching and learning programme

## Project-based learning in digital technologies: Myths and legends

MINISTRY OF EDUCATION
TE TĀHUHU O TE MĀTAURANGA

Developed by Ruth Davey, Lincoln High School 2017

The full teaching and learning programme resources, associated materials and an assessment task will be supplied in 2018.

**External links to websites**
The Ministry of Education does not accept any liability for the accuracy of information on external websites, nor for the accuracy or content of any third-party website accessed via a hyperlink from this resource. Links to other websites should not be taken as endorsement of those sites or of products offered on those sites. Some websites have dynamic content, and we cannot accept liability for the content that is displayed.

Published 2017 by the Ministry of Education
PO Box 1666, Wellington 6011, New Zealand

www.education.govt.nz

## Summary

In this themed, project-based learning programme, students will gain digital skills and knowledge as they create a game that assists other students to learn about and understand a myth or legend. This resource provides ideas for teachers on how to adapt this approach to different outcomes and themes. It also provides ideas on how to structure the learning so that assessment falls naturally out of the activities that are part of the programme.


The project-based learning approach allows for more holistic learning to take place. It is the teacher's job to ensure that student evidence will be suitable to be assessed against the chosen standards. Students will be working on various aspects of the project throughout the year so that the teaching approach is not simply "to the test" but rather for the development and progression of the project. This allows students to better develop their skills and extend their own knowledge.

Even though all students are working on very similar projects, each is unique. Students are inspired and empowered to meet the challenge of creating a unique and singular outcome, as opposed to a mass crafting of identical outcomes purely for assessment purposes. They are able to interact with each other as they work on their projects, sharing ideas of how various challenges are solved without compromising the authenticity of their own work. This method allows for the teacher to encourage and extend students who are strong learners able to direct their own learning, while also having the opportunity to support and scaffold students who may need more assistance.

## By the end of this teaching and learning programme, students will be able to:

- research to find ideas and user preferences from existing games and programming techniques that will assist them in creating their own game
- create and analyse survey data to enhance their ideas
- iteratively develop and test code for their own game in the chosen computer language
- investigate sorting and searching algorithms
- competently design and evaluate good human computer interfaces.

## Duration (terms, weeks, teaching periods)

This resource outlines a year-long programme. It details what to teach and assess each term and provides the assessments required.

## Key teaching and learning concepts – the big ideas

In New Zealand, some of our students have limited knowledge and access to the myths and legends of the diverse groups of people that make up our population. An interactive game provides the perfect platform to introduce these myths, legends, and traditions in a way that inspires the younger generation to embrace them.

Students will spend the year researching, learning skills, and creating a game that will familiarise other students with their chosen myth or legend. In the process, they will learn how to create a technological outcome, to collaborate with others, to describe and explain their ideas, to develop their outcome in an iterative fashion, and to be aware of the needs of a target audience beyond themselves. They will ultimately develop an outcome that is fit for purpose and takes the needs of others into consideration.

*Note: that not everything taught will be assessed.*

## Alignment to NZC and/or Te Marautanga

**DTHM – Computational Thinking (CT)**

- PO5 – independently decompose problems into algorithms
- PO5 – implement algorithms as a program, with comments
- PO5 – uses organised approach for testing and debugging
- PO6 – apply modular structure to program
- PO6 – store data in collections
- PO6 – can compare costs of two different searching and/or sorting algorithms in relation to number of comparisons
- PO7 – use an iterative process to design, develop, document and test a computer program

**DTHM – Designing and Developing Digital Outcomes (DDDO)**

- PO3 – can select best tools/techniques to solve a problem
- PO3 – work through an iterative process to design, develop, create, store, test and evaluate a program to meet its purpose
- PO4 – independently work through above iterative process

**Generic Technology (brief development)**

- describe the nature of the intended outcome, explaining how it addresses the need or opportunity
- describe key attributes identified in stakeholder feedback
- justify intended outcome based on need and opportunity
- describe specifications that reflect stakeholder feedback.

## Links to other learning areas

The theme of myths and legends could link to Māori, English, or Classics. Other themes could provide links to different subject areas:

- Science – fair testing
- Mathematics – determining best, worse and average values using experimental statistics

## Teaching and learning pedagogy

This is a project-based approach to planning an entire year's work. Students involved in this project learn the required skills and techniques and complete assessments to demonstrate their understanding. They produce one body of work, although this may be broken down into term sections. It is the teachers' job to ensure that students are correctly scaffolded and managed to complete the work and that all the evidence required by the standards is presented by the students.

## Prior knowledge/place in learning journey

Prior knowledge of using a word processor, file and folder management, basic literacy, searching skills when using a web browser, understanding of copyright and creative commons are required.

## Resources required

- Internet connection, web browser and access to YouTube videos
- software – word processor; Python 3.6 or higher; Pygame 2.9; Arcade; text-editing software, such as Notepad++, or IDE, such as Wing 101, Pycharm or Sublime; survey software, such as Microsoft Office 365 Forms or Google Forms; database software, such as Microsoft Access
- hardware – desktop or laptop computer
- resources such as printed cards for sorting and searching data, and other devices used to explore human–computer interface usage (see Appendix)

## How you might adapt this in your classroom

A different programming language or game engine, as well as the resources for editing and running them, can be used. Some options are: App Inventor, Game Maker, GameFroot, Javascript with Phaser or CodeAvengers, Unity, or even Scratch.

One caution here: it is wise to limit the development to a 2D game as there is not enough time or resources to easily develop the 3D graphics for a first-person shooter game. Students should also not attempt to make too many of the required graphics themselves. In coding a game, the important aspect is how the game functions.

- It isn't necessary to do all the assessments, but the entire programme should be taught. Some students need more time to learn some topics than others, so adapt the programme accordingly.
- The searching and sorting assessment could be moved to term 3, doing only some programming of searches and sorts in term 1 as a class programming exercise. Database analysis of user opinions would then move into term 1 and the rest of the programme would move forward accordingly.
- Some students struggle completing the iterative design and coding assessments together. Use your judgement as it is quite acceptable for them to only be assessed on one of these. This allows for differentiation in the programme. They should still attempt some form of both.
- Some schools prefer not to do the external assessment with their students. In this case, leave out the HCI assessment.
- This approach of themed project-based learning can easily be adapted to suit a different theme (for example, transport problems) or a different digital outcome, such as producing an infographic, a website, a brochure, or a printed board or card game. The only additional alterations would be the nature of the research to be conducted and the skills to be taught, enabling the relevant outcome development to be achieved.

  For example, instead of programming a game, a board or card game could be developed using tools like InDesign and Illustrator or equivalent free software such as GIMP and Inkscape. For this alternative, instead of learning to code and use algorithms, students will need to learn to use the chosen drawing software, the principles of design, and how to incorporate these in their work.

Additionally, it is possible to weave in aspects of the Hangarau Matihiko curriculum if desired.

## Assessment

Technology 91044: Undertake brief development to address a need or opportunity (4 credits) **or** AS 91877: Create a proposal for a digital outcome (3 credits)

AS 91885: Demonstrate understanding of searching and sorting algorithms (3 credits)

AS 91879: Develop a digital outcome to manage data (4 credits)

AS 91883: Develop a computer program (4 credits)

AS 91884: Use basic iterative processes to develop a digital outcome (6 credits)

AS 91886: Demonstrate understanding of human computer interaction (3 credits)


# Term 1 outline

## The Learning context:

This term, students will be learning to code in Python, how to think like a coder (algorithms), and how to plan code and test cases for more involved programs. They will be investigating searching and sorting algorithms and do their first assessment. It is important to continually teach good habits like planning and commenting, even for the simplest programs. It is also important that students complete their first assessment this term – credits earned are a good motivator, and students will then be more familiar with the required disciplines in this subject.

It is often better to weave the learning so that although a certain amount of time is devoted to learning a topic, they cover two topics together over the total time allocated. This is done by devoting half the weeks' lessons to each topic and works well when students are learning to code and learning computer science, thinking, analysing, etc. Half the week they will be coding and for the rest of the lessons they will be doing the computer science, thinking, etc. This also has the effect of making them perceive that they continually do some coding, even if it's not every lesson.

| What is being covered | Approximate duration | Specific Learning Outcomes<br>Students will be able to: | Learning Activities | Resources |
|---|---|---|---|---|
| Introduction of teacher and students and of students to the awesome possibilities in IT<br><br>Introduction to the program and basic good practice<br><br>Introduction to Python (or your choice of programming language) | 1 week – or part thereof (first week back at school) | • Understand that there is more to DT than just coding and assessment<br>• Manage their files and folders effectively<br>• Understand some ethics<br>• Write small, basic programs | • Introductions, mihimihi and class rules<br>• Explain overview of the course<br>• Best to warn students that they will be creating a good 2D game – not enough time or manpower to create amazing 3D graphics. This course is about creating a usable program fit for purpose.<br>• "Find someone who …" ice-breaker exercise<br>• Create file and folder structure for the year<br>• Write "hello world" program as an introduction to IDE and Python language<br>• comments and program structure<br>• Introduction to input and output<br>• Exercise 1 | • YouTube videos from "Made in Code" series<br>• "Find someone who …" worksheet<br>• Notes on teaching programming<br>• Python v3.6<br>• Pygame<br>• Arcade<br>• Wing 101<br>• Example code |

## The Learning context *(continued)*:

| What is being covered | Approximate duration | Specific Learning Outcomes<br>Students will be able to: | Learning Activities | Resources |
|---|---|---|---|---|
| Basic Python | 3 weeks | • Work in the IDE to develop small programs<br>• Understand how to use:<br>– variables<br>– if statements<br>– loops<br>– functions | • some print formatting<br>• short practice exercise<br>• variables and CONSTANTS – more conventions<br>• maths expressions<br>• Exercise 2<br>• If statement<br>• thinking – how to decompose a problem example<br>• Exercise 3<br>• for loops – with worked example<br>• for loops exercise 4<br>• while loops – with worked example<br>• while loops exercise 5<br>• simple try except error handling<br>• introducing functions<br>• mixed exercise 8<br>• challenge exercise 9<br>• (strings and lists can also be done if there is time. Exercises included) | • Python exercises<br>• Conditional exercise<br>• For loops exercise<br>• While loops exercise<br>• Mixed loops exercise<br>• Try except example<br>• Try except as a function |

## The Learning context *(continued)*:

| What is being covered | Approximate duration | Specific Learning Outcomes<br>Students will be able to: | Learning Activities | Resources |
|---|---|---|---|---|
| Introduction to algorithms<br><br>Searching and sorting algorithms<br><br>Iterative program development<br><br>Python programming | 4 weeks | • Describe an algorithm<br>• Program in Python using:<br>– basic lists<br>– functions<br>• Create a program using iterative development<br>• Write code to perform:<br>– a search algorithm<br>– a sort algorithm<br>• Understand how to calculate the cost of an algorithm | • What is an algorithm?<br>Some helpful algorithm videos to use in this topic:<br>https://www.youtube.com/watch?v=CvSOaYi89B4<br>https://www.youtube.com/watch?v=6hfOvs8pY1k<br>https://www.youtube.com/watch?v=ENWVRcMGDoU<br>https://www.youtube.com/watch?v=gOKVwRIyWdg<br>https://www.youtube.com/watch?v=vSi6YoTPWLw<br>https://www.youtube.com/watch?v=wTBJeo2rBkk<br>*(It is not necessary to watch all of these at once. Some can be watched as starter activities for lessons)*<br>– Introduction to Algorithms<br>– Have a class discussion about algorithms<br>• introduce lists – the lyrics program<br>• using functions – the lyrics program<br>• programming exercises<br>• magic 8 ball exercise<br>• text adventure exercise<br>• introduce planning & testing with an example<br>• more complex programs exercise<br>• write code to generate a list of random numbers from a minimum to a maximum<br>• Searching algorithms<br>– Intro – A* video https://www.youtube.com/watch?v=ySN5Wnu88nE<br>– Where do we use searching? | • Old phone book<br>• Scale and different weights<br>• CSFieldGuide.org.nz algorithms chapter<br>• Resources:<br>– unsorted list<br>– sorted list<br>• Search/sort cards<br>– word cards<br>– number cards |

## The Learning context *(continued)*:

| What is being covered | Approximate duration | Specific Learning Outcomes<br>Students will be able to: | Learning Activities | Resources |
|---|---|---|---|---|
| | | | – Class discussion on why searching is important https://www.statisticbrain.com/google-searches/<br>• other ideas and Google searching will be useful<br>– Introduce searching algorithms<br>– allow students to experiment with searching:<br>• through phone book for:<br>– a name<br>– a number<br>• through a list of numbers for a specific number (see resource – sorted and unsorted list)<br>• class discussion on using methodical methods when searching<br>• Introduce linear search<br>– practice linear search with various resources<br>– write a program to do a linear search (model good program planning, creating, testing while doing this together)<br>*Note: a good reference book for ideas on how to write such a program: http://arcade-book.readthedocs.io/en/latest/chapters/15_searching/searching.html*<br>• Introduce binary search<br>– practice binary search with various resources<br>– write program to do binary search | |

## The Learning context *(continued)*:

| What is being covered | Approximate duration | Specific Learning Outcomes<br>Students will be able to: | Learning Activities | Resources |
|---|---|---|---|---|
| | | | • Discuss how best we could compare algorithms. Such a discussion should lead to counting of comparisons as a reliable way to determine the cost of the algorithms. Why is measuring how long it takes to perform the algorithm not so effective? (different machines will yield different results. Easier to limit variable change)<br>• Introducing sorting algorithms http://www.youtube.com/watch?v=EdUWyka7kpI<br>– practice following sorting algorithms by hand using various resources<br>– write a program for selection sort https://www.w3resource.com/python-exercises/data-structures-and-algorithms/<br>– Other types of sorts and sort comparisons http://www.youtube.com/watch?v=WcqaSxhIVpc http://youtu.be/aXXWXz5rF64 http://youtu.be/es2T6KY45cA http://www.zutopedia.com/ms_vs_qs.html or search the web for your own searching/sorting videos<br>– find programs for quick sort and/or merge sort<br>– desk check the program to see how it works<br>– use this understanding to add counting of comparison to all sorting programs<br>– Class discussion of real world use of searching and sorting algorithms<br>• what problems would arise if search/sort was slow? | |

## The Learning context *(continued)*:

| What is being covered | Approximate duration | Specific Learning Outcomes<br>Students will be able to: | Learning Activities | Resources |
|---|---|---|---|---|
| Investigation of searching and sorting algorithms assessment | 2 weeks | • Understand how to experiment to get a better idea of implications of searching/sorting big datasets<br>• Complete a report on their findings | https://www.w3resource.com/python-exercises/data-structures-and-algorithms/ Good site for programs, but you need to add comparison count<br>• Use programs developed/found to conduct some experiments on different-sized datasets. The idea here is to do a fair test (as done in science), so the same set of data should be used for both searching algorithms and the test, which should be carried out 10 times to give a reasonable calculated value for the best, worst, and average values (statistics in maths). For even better results, combine results as a class so that you have 30 sets of data for one dataset size.<br>– choose a range of manageable dataset sizes with regular intervals between them – for example, 100, 200, 300, 400, 500<br>– for each dataset size, carry out the following experiment 10 times, noting the results in an Excel spreadsheet each time:<br>• generate a random list of numbers<br>• run two different searches on the same list of items<br>– use Excel functions min, max, and average to calculate the best, worst, and average cost for searching that dataset size from the 10 sets of collected data for each dataset size (see example in resources)<br>– graph the average results using a line graph<br>– repeat this experiment for two sorting algorithms also<br>• Complete the assessment. | |

# Term 2 outline

## The Learning context:

This term, students will begin to develop a brief for their game, learn about game development, learn how to capture and analyse user opinions, do their second assessment and learn how to use a game engine to assist them in creating their game.

Remember to weave the learning through the week. This also means discussing issues when they arise. For example, it will be helpful to discuss some aspects of HCI and design at this early stage of the development.

Students should also be continually practising the planning for code and test cases wherever appropriate.

| What is being covered | Approximate duration | Specific Learning Outcomes<br>Students will be able to: | Learning Activities | Resources |
|---|---|---|---|---|
| Game development theory | ½ week | • Understand some game development theory<br>• See that there is more to game creation than implementing their own idea<br>• Be more aware of the importance of stakeholder preferences | • If possible, organise a visit from a game development professional sometime during this section. FutureInTech can help with this.<br>• A professional will inform students of some of the realities around game creation and reinforce your restrictions because of time constraints – they will only have a few weeks to craft their game program.<br>• Watch “So you want to be a game designer” https://youtu.be/zQvWMdWhFCc<br>• Watch “What is a game?” https://youtu.be/H0ReU2tvLFo<br>• Research game flow theory (not the game flow) http://www.jenovachen.com/flowingames/flowtheory.htm and https://gamedesignconcepts.wordpress.com/2009/07/20/level-7-decision-making-and-flow-theory/<br>• Do Analysing Games worksheet<br>• Class discussion on what is good to include, what is to be avoided | • Analysing Games worksheet |

## The Learning context *(continued)*:

| What is being covered | Approximate duration | Specific Learning Outcomes<br>Students will be able to: | Learning Activities | Resources |
|---|---|---|---|---|
| Using databases to analyse user opinion | 3 weeks | • Create a survey and capture results<br>• Create a database that includes a table, a report, and some queries.<br>• Interpret results of database queries | • Research into opinion surveys – what makes a good survey<br>• Class discussion of results<br>• Research – how to create good questions<br>• create a simple class survey – using Forms (either Google or Office 365). Choose a topic they are all familiar with so no other research is necessary.<br>• Send out survey<br>• Investigate survey tools – produce a PMI chart<br>• Creating a database basics:<br>– what is a database?<br>– what databases do you already use?<br>– why are databases useful?<br>– naming conventions for tables, fields etc.<br>– creating tables<br>– data entry<br>– creating forms<br>– creating queries<br>– creating reports<br>– importing data to database table<br>– using database to analyse results – smart queries<br>– interpreting results<br>• Opinion survey and analysis exercise | • Survey Tools list<br>• Opinion survey exercise<br>• Access or another database<br>• Choice of survey tool |

## The Learning context *(continued)*:

| What is being covered | Approximate duration | Specific Learning Outcomes Students will be able to: | Learning Activities | Resources |
|---|---|---|---|---|
| Database assessment and begin research for initial brief | 2 weeks | • Conduct research on existing outcomes and topic<br>• Conduct research on user preferences<br>• Create a survey<br>• Analyse results<br>• Write a short report explaining the results | • Research existing games – create PMI<br>• Research myths and legends<br>• Research user characteristics to determine possible game preferences<br>• Use research to create thoughtful survey on user game preferences – to inform required attributes and specifications<br>• Create a database to analyse survey<br>• Create queries to analyse user preferences from survey<br>• Send out survey and collect results (at least 10)<br>• Create database to analyse results<br>• Create a range of initial game ideas (at least 3)<br>• Report on user preferences and choice of stakeholders for project (no more than 3 stakeholders) | • About that Game assessment |
| Develop initial brief | 2 weeks | • Create an initial brief detailing important user requirements, constraints, and possibilities | *Note: this section and the following one work well together sharing the week's lessons. As students learn to use the game engine, they firm up their ideas for their own game creation.*<br>• Use research info from Investigating Game Preference assessment to inform brief (no need to do it again)<br>• Share game ideas with chosen stakeholders and ask for feedback<br>• Choose the idea that will be developed and justify choice | • Continue About that Game assessment – Note, 91044 will not be complete until Game On assessment is complete |

## The Learning context *(continued)*:

| What is being covered | Approximate duration | Specific Learning Outcomes Students will be able to: | Learning Activities | Resources |
|---|---|---|---|---|
| | | | • Determine purpose of outcome<br>• Determine user and stakeholder profiles<br>• Determine potential hardware and software for prototype development<br>• Determine key attributes<br>• Determine initial specifications | |
| Learning to create games using Arcade search engine | 2 ½ weeks | • Draw on computer<br>• Understand the game loop used in Pygame<br>• Understand the program structure required for a game<br>• Create a simple platform game | • Teacher: watch "Teaching Python 3.6 using games" for background info on using games to teach Python https://www.youtube.com/watch?v=MbJUMMvNMqk or read http://2017-craven-webinar.readthedocs.io/en/latest/ the notes for this webinar<br>• Teacher reference book: http://arcade-book.readthedocs.io/en/latest/ and http://programarcadegames.com/ video tutorials are also in this online book resource (all of these may be shared with students at teacher discretion – author has granted permission)<br>Students need to explore and experiment with techniques that they will use when developing their game. They should make some basic notes on what they have learned while doing this for use in their brief development in next assessment.<br>• The following techniques will be useful code from the above book for students to explore. The examples and explanations will help develop their understanding so that they will be able to create exciting Arcade-type games of their own. They will not be graded on any of this code, which can be supplied to the students, but will be graded on the parts of the program that they create on their own. | • base_game_code.py (from Arcade resources) |

## The Learning context *(continued)*:

| What is being covered | Approximate duration | Specific Learning Outcomes<br>Students will be able to: | Learning Activities | Resources |
|---|---|---|---|---|
| | | | • Students should make notes on what they have learned, to add to their refined brief. This is investigating techniques and their potential to be included in the outcome<br>• Techniques include but are not limited to:<br>– how to draw on your computer (Ref: chapter 2)<br>– simple shape animation<br>– array backed grids<br>– platformers and variations<br>– mazes<br>– sprites<br>• Brief class discussion about the interface of the game and what could make it more usable. | |

# Term 3 outline

## The Learning context:

This term, students will be creating their game. They will also craft their external report towards the end of the term. They will have 2 weeks in the last term to fix up and complete their external submission.

| What is being covered | Approximate duration | Specific Learning Outcomes Students will be able to: | Learning Activities | Resources |
|---|---|---|---|---|
| Using Arcade game engine to create a game | 1 week | | • More experimenting with Arcade code | |
| Iteratively creating a game and a computer program for the game, refining the brief as progress is made | 6 weeks | • Create a game from their brief<br>• Create a refined brief<br>• Use iterative process to design, create and test game<br>• Create a final brief | *Note: students who struggle with using Arcade and Pygame can create a simple quiz game or guessing game using only Python*<br>• Do the assessment<br>• Refine the brief<br>• The teacher needs to manage student work with regular checkpoints to establish progress<br>• Students may talk to each other and discuss potential solutions or share techniques, but they may NOT copy from, or write code for, each other<br>• Complete the final brief, planning, and prototype | • Game On Assessment |
| HCI and crafting the external report | 1½ weeks plus (1½ weeks of school exams) | • Understand why HCI is important<br>• Create a report for external submission | • Class discussion on usability of interfaces and simple HCI situations and examples<br>• Class discussion on how to evaluate interfaces<br>• Class discussion about heuristics<br>• Do various HCI exercise from CS Unplugged, CS Field Guide or similar resource | • Assortment of devices<br>• HCI External Assessment |

**The Learning context** *(continued)*:

| What is being covered | Approximate duration | Specific Learning Outcomes Students will be able to: | Learning Activities | Resources |
|---|---|---|---|---|
| | | | • Class exercises evaluating Interfaces together and suggesting some improvements<br>• Reflecting on the game they created and the interface it has<br>• Create report. | |

# Term 4 outline

| What is being covered | Approximate duration | Specific Learning Outcomes Students will be able to: | Learning Activities | Resources |
|---|---|---|---|---|
| Term 4: Fix and enhance HCI report | 2 weeks | • Improve external report | • Work on responding to feedback in order to improve external report<br>• Submit external report | |


## Acknowledgements

I would like to acknowledge the following for their ideas and contributions to making this resource possible:

Prof Paul Craven – for Arcade, and his amazing resources that make it possible for year 11 students to create great 2D games in limited timeframes, and for giving us permission to use his resources. He is happy to encourage new programmers.

Julie McMahon – who inspired me with the idea for approaching teaching and learning in this way.