Assessment Task #1: Software Design Report

Due: 1st March 2019

By: Michael Kneale

**Problem Identified: Poor Spelling Education**

Spelling is not being taught to a sufficient degree in many primary schools in Australia. There is a large gap in the education of children after the teaching of sight words in kindergarten. There are also inconsistencies in the teaching of spelling between schools and even classes within the same school, resulting in varied levels of literacy between students.

It is imperative that students are given a good foundation in spelling in their early years of education in order to successfully communicate, both verbally and in writing, as adults, which is emphasised by literacy expert J. Richard Gentry.

**Solution**

SpellZone is a unique piece of education software designed to aid primary school teachers and, in particular, the students, by giving them a platform on which to develop their spelling skills. The SpellZone package includes:

* different levels of difficulty for different ages
* themed spelling lists to interest the children
* help menu so that students do not get lost
* positive reinforcement to encourage continued use of the program

The main objectives of SpellZone are as follows:

* to improve spelling ability in primary school students
* to increase their confidence in writing
* to expand their vocabulary
* to increase the awareness of this gap in the syllabus

**Emerging Technologies in the Educational Environment**

Research on emerging technologies in education revealed the following:

* mobile learning is becoming more widespread due to its potential for accessibility, speed, and connectedness
* the use of artificial intelligence, especially in the field of robotics, has a huge amount of potential because of its interactivity with the user and adaptability
* natural user interfaces (NUIs) allow for the quicker and more intuitive traversal of user interfaces, such as through voice recognition
* adaptive learning technologies adjust accordingly to the user’s skill level and progress
* Internet of Things uses embedded sensors which enable schools or universities to transfer information between each other
* next-generation learning management systems (LMS) are digital learning-centred models which focus on the individual
* cloud computing is growing quickly in popularity as it is helpful for educational facilities that need solutions which are budget friendly
* children, especially primary age children, often learn better when they are having fun, and gaming has become a big part of the emerging technologies in the educational environment, especially over the last decade or so. For instance, websites such as Mathletics and software such as Tux Typing are widely used and have been shown to get results for students

In light of these emerging technologies, SpellZone has been designed with a focus on the individual’s needs and wants, and it is in a colourful format so that it would be appealing for the younger generation.

**Justification of Programming Language**

On the basis of the skills of the programmer and the current educational climate, with regard to technology, Python was chosen as the programming language to be used. It is also the best choice in terms of the future of the relevant education setting, that is, primary school teaching. This is because of the ongoing support Python receives from both its developers and the community, which will mean that if SpellZone requires updates in the near or distant future, it will be able to accommodate these changes.

Python is the ideal choice for the following additional reasons:

* it is one of the most convenient combinations of simplicity and flexibility, being both easy to learn and suitable for both simple and complex applications
* it is compatible with many operating systems (Windows, Mac, Linux, etc.)
* due to its popularity, there is an extremely large number of guides and amount of information to assist, much of which is available for free
* there are a number of third party modules suited to many different programming needs
  + QT Designer, in particular, allows for the implementation of the GUI functionality required to design SpellZone
* the code is readable and maintainable
* Python was TIOBE’s programming language of the year in 2018 – an indication of its quality

**Software and Skills Required to Complete Solution**

The pieces of software required to complete the solution and its documentation are Python, Qt Designer, PyQt5, Command Prompt, and Microsoft Excel, Word, and Publisher.

A large portion of the GUI, including most of the design, will be done with Qt Designer, as it is the easiest and quickest method of designing and rearranging user interfaces. Python, in conjunction with the PyQt5 module, will be used to edit the code itself, doing things such as making buttons perform actions upon being clicked and linking different windows so they can be moved between by the user.

The piece of software used to document the design of the program, as well as most other things, was primarily Microsoft Word. The main reasons for this choice were my pre-existing knowledge of how to use the program and the ease with which anyone can access the file, due to the fact that the majority of homes and schools have access to Word. Additionally, Word has a very simple and intuitive layout and allows for the creation of many pieces of documentation relevant to the solution, including tables and flowcharts. It is also fast to use, especially thanks to the ribbon across the top and its many options regarding font, design, layout, and other aspects of the document. Microsoft Word is necessary for documentation such as the IPO chart and the flowchart. Additionally, Microsoft Excel is necessary for the creation of the Gantt chart, which will help tremendously with the organisational aspect of the project. Microsoft Publisher will be used for the software’s instructions, which will be distributed to the teachers and students using the program.

Even though SpellZone will primarily be designed through Qt Designer and PyQt5, sufficient knowledge of the basics of Python will be required in order to efficiently code the solution as PyQt5 is based on its logic. Skill with Qt Designer and PyQt5 will be required, and especially the ability to use them in combination with one another. Also necessary is sufficient understanding of the industry standards of graphical user interfaces, in order to make the interface functional, aesthetically pleasing (through the fonts, colours, backgrounds, and so on), and familiar (so that it is easy to understand which buttons do what without putting too much time into it). Command Prompt, which comes preinstalled on Windows, is needed for the installation of the PyQt5 module, as well as knowledge on how to use it.

**Pseudocode**

BEGIN

WHILE number of words quizzed < 10

play audio of spelling word

prompt user for spelling

get spelling

IF answer correct

add 1 to score

display “You are correct. Good job!”

ELSEIF answer incorrect

display “That is incorrect. Better luck next time!”

ENDIF

ENDWHILE

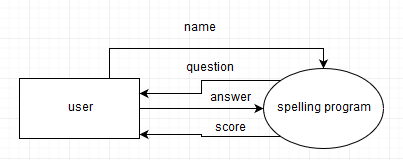
display score screen

END

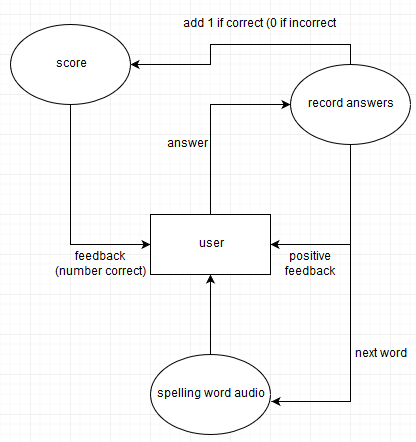
**IPO Chart**

|  |  |  |
| --- | --- | --- |
| **Input** | **Process** | **Output** |
| Name  Letters (for spelling)  Restart quiz with same or higher difficulty level  Exit | Randomly chooses spelling word from selected level  Checks whether word is correct  Add 1 to score for correct answers  Calculate total score as a percentage  Restart test | Student’s name  “Good luck!” at beginning of game  “Better luck next time!” when incorrect  “Good job!” when correct  Sounds of spelling words  Number of stars from overall score |

**Context Diagram**



**Data Flow Diagram**

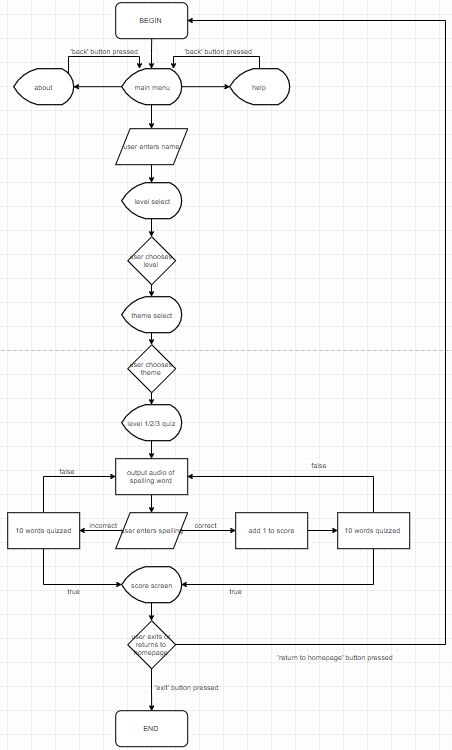


**Structure Chart**

A close up of text on a white background

Description automatically generated

**Systems Flowchart (following page)**



**Desk Check**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Line** | **Output** | **Input** | **userAnswer correct?** | **userScore** |
| 1 |  | “Go” button pressed |  |  |
| 2 | audio of the word “cake” |  |  |  |
| 3 |  | cake |  |  |
| 4 |  |  | True |  |
| 5 |  |  |  | add 1 |
| 6 | audio of the word “play” |  |  |  |
| 7 |  | plai |  |  |
| 8 |  |  | False |  |
| 9 |  |  |  | add 0 |
| 10 | audio of the word “fun” |  |  |  |
| 11 |  | fun |  |  |
| 12 |  |  | True |  |
| 13 |  |  |  | add 1 |
| 14 | audio of the word “clown” |  |  |  |
| 15 |  | clown |  |  |
| 16 |  |  | True |  |
| 17 |  |  |  | add 1 |
| 18 | audio of the word “games” |  |  |  |
| 19 |  | gams |  |  |
| 20 |  |  | False |  |
| 21 |  |  |  | add 0 |
| 22 | audio of the word “card” |  |  |  |
| 23 |  | card |  |  |
| 24 |  |  | True |  |
| 25 |  |  |  | add 1 |
| 26 | audio of the word “party” |  |  |  |
| 27 |  | partee |  |  |
| 28 |  |  | False |  |
| 29 |  |  |  | add 0 |
| 30 | audio of the word “food” |  |  |  |
| 31 |  | food |  |  |
| 32 |  |  | True |  |
| 33 |  |  |  | add 1 |
| 34 | audio of the word “toy” |  |  |  |
| 35 |  | toy |  |  |
| 36 |  |  | True |  |
| 37 |  |  |  | add 1 |
| 38 | audio of the word “child” |  |  |  |
| 39 |  | chield |  |  |
| 40 |  |  | False |  |
| 41 |  |  |  | add 0 |
| 42 | userScore |  |  |  |

**Data Dictionary**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data item** | **Type** | **Length** | **Valid values** | **Description/purpose** |
| userScore | integer | 10 | |0-9| | Tracks how many words the user has spelled correctly |
| userAnswer | Boolean | 10 | |True|, |False| | Allows the program to recognise whether the user answered correctly |
| userWord | string | 20 | |A-Z|, |a-z|, |space|, |0-9|, |#, &, \*, $, etc.| | Spelling of the word entered by the user |

**Identifying Appropriate Test Data and Live Tests**

As the program is not particularly complex, statement coverage, that is, execution of every statement possible in the program at least once, will be easy. Statement coverage in testing will be made simpler through looking at the already constructed flowchart and data flow diagram.

Decision coverage is the coverage of every true or false outcome of every Boolean expression. Given the overall simplicity of the program, especially how few choices the user has the possibility of making, this will be easy to achieve in testing.

Branch coverage is, as the name suggests, testing every possible branch in the programming, and will be similarly easy.

Testing was done by that covered every branch, decision, and statement in SpellZone. This uncovered a few typos, but no major bugs or mistakes were discovered. Testing was also done by potential users of the program, as well as some of my colleagues, and no bugs were discovered.

The ability for the student to type anything into the program when being tested on the spelling of the word was one of the main things tested. This was tested to ensure that the parameters in place, which restrict the character limit to 20 and what could be typed to letters only, were working, and that the user could not break the program by typing an incredibly long or otherwise potentially program-breaking word. They would just be told that what they typed was incorrect. No errors were found regarding this part of SpellZone.

The following inputs were tested to ensure that the program could handle them.

|  |  |  |
| --- | --- | --- |
| **Input (test data)** | **Expected output** | **Actual output** |
| (pressing the spacebar a number of times) | That is incorrect. Better luck next time! | That is incorrect. Better luck next time! |
| ~!@#$%^&\*()-\_=+123 (testing different symbols and numbers) | That is incorrect. Better luck next time! | That is incorrect. Better luck next time! |
| iejenresnaoiwejntlksmdopxkopwaeanlekwajwioje (testing the character limit) | That is incorrect. Better luck next time! | That is incorrect. Better luck next time! |
| 5kd55[js4dpa`5#& (testing having numbers, symbols, and letters simultaneously) | That is incorrect. Better luck next time! | That is incorrect. Better luck next time! |
| kangaroo (testing a regular word) | You are correct. Good job! | You are correct. Good job! |
| CRIMSON (making sure the program accepts correct spellings regardless of capitalisation) | You are correct. Good job! | You are correct. Good job! |
| recEiVe (making sure the program accepts correct spellings regardless of capitalisation) | You are correct. Good job! | You are correct. Good job! |

**Screenshots (following pages)**

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

**Critical Evaluation of Project Against Objectives**

The project failed to meet any its objectives as the program did not end up working on any usable level due to complications with the graphical user interface. Specifically, the switching between windows proved too great an obstacle to overcome, despite many hours of attempting it. The program cannot improve the spelling ability of primary school, increase their confidence in writing, expand their vocabulary, or increase the awareness of this gap in the syllabus, as it does not function as intended.

**Potential Modifications and their Implementation**

The primary modification that is required is to connect the windows of SpellZone so that the user can navigate the program. Audio should also be included so that the user can hear the words they need to spell. These modifications should be achievable but would require a great deal of expertise and time to implement.

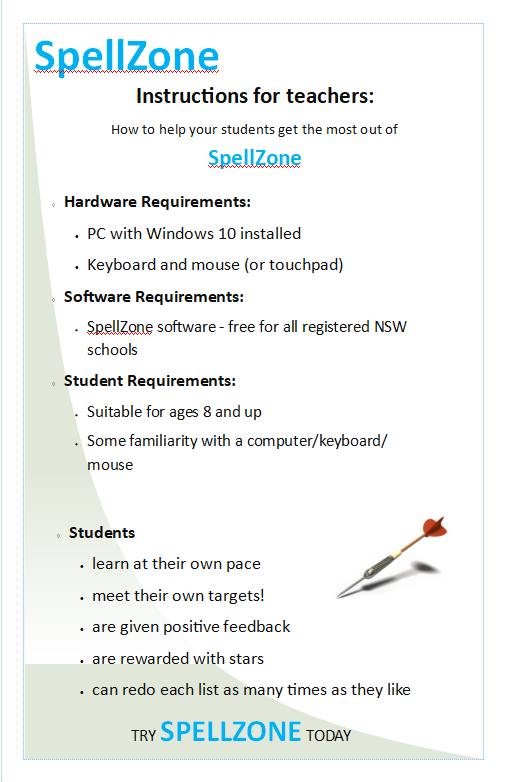
**SOFTWARE DESIGN REPORT – GANTT CHART**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 13/1 | 14/1 | 15/1 | 16/1 | 17/1 | 18/1 | 19/1 | 20/1 | 21/1 | 22/1 | 23/1 | 24/1 | 25/1 | 26/1 | 27/1 | 28/1 | 29/1 | 30/1 | 31/1 | 1/2 | 2/2 | 3/2 |
| Defining and Understanding the problem  Identify a problem/valid need |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Generate ideas to solve need |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Research - programming tools, education |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Applying project management skills |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Planning and Designing  IPO chart |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Pseudocode |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Context diagram |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Data dictionary |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Data flow diagram |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Systems flowchart |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Desk check |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Define test data |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Implementing  Design GUI |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Include common modules |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Begin and continue coding solution |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Systematic removal of errors |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Testing and Evaluating  Test/retest program on different systems |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Critically evaluate project |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Find and remove bugs |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Maintaining  Check all needs have been met |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Identify potential modifications |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Include documentation for the user |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

**Logbook**

|  |  |  |
| --- | --- | --- |
| **Date** | **Tasks done** | **Resources used/needed (apart from Microsoft Word)** |
| **11/1/19** | Identified need of spelling education after consideration of other areas of need  Researched spelling strategies online  Started and completed title page for the documentation, including the assessment name, due date, and author | Anecdotal evidence (many high school children have difficulty spelling)  <https://www.psychologytoday.com/us/blog/raising-readers-writers-and-spellers/201008/no-spelling-book-in-your-child-s-book-bag-spells> |
| **13/1/19** | Reviewed software options for making a Gantt chart (in following column) | <https://www.canva.com/graphs/gantt-charts/>  <https://app.ganttpro.com/>  <https://creately.com/Gantt-Chart-Software>  <https://planhammer.io/online-gantt-chart.html>  Microsoft Excel |
| **14/1/19** | Documented the solution to the identified problem, and identified the objectives of the SpellZone program | N/A |
| **16/1/19** | Started and completed Gantt chart, detailing the projected completion times for every task in the project | Microsoft Excel |
| **17/1/19** | Looked at tutorials on Qt Designer regarding how to design a GUI  Started and finished pseudocode, outlining the process of the quiz itself | <https://www.youtube.com/watch?v=Dmo8eZG5I2w>  <https://www.pythonforengineers.com/your-first-gui-app-with-python-and-pyqt/> |
| **18/1/19** | Researched and documented information regarding the suitability of Python as the programming language to be used  Researched and documented a number of emerging technologies in the educational environment  Started and completed desk check | <https://medium.com/@mindfiresolutions.usa/python-7-important-reasons-why-you-should-use-python-5801a98a0d0b>  <https://www.nmc.org/>  <http://www.opencolleges.edu.au/informed/features/6-emerging-educational-technologies-used-across-globe/>  Effective use of Cloud Computing in Educational Institutions – Tuncay Ercan |
| **20/1/19** | Began coding the GUI of the program and designing the different windows required for the running of the program | Godot |
| **21/1/19** | Started and finished both the IPO chart and the context diagram, both of which portray the program on a very basic and easy to understand level  Switched to Qt Designer for designing the GUI for ease of use  Worked on the main window of SpellZone, including formatting, colour, and font | <https://www.draw.io/>  Qt Designer |
| **22/1/19** | Continued work on main window of SpellZone (font, background, buttons)  Started and completed the data dictionary, detailing the data items used in the program and explaining their uses | Qt Designer |
| **23/1/19** | Elaborated on the software used to document the design of the program, that is, Microsoft Word, Excel, and Publisher | N/A |
| **24/1/19** | Worked on alignment of buttons and other widgets in the program’s windows  Documented the software and skills required to complete the software solution | Qt Designer, Stack Overflow |
| **25/1/19** | Refined design of window and widgets  Reviewed software textbook for help with data flow diagram  Started and completed the data flow diagram | Qt Designer, Stack Overflow, <https://www.draw.io/> |
| **26/1/19** | Reviewed a number of sites regarding spelling and strategies for spelling, as well as spelling lists from my primary school  Developed word lists based on themes as well as different levels of spelling ability | <https://education.nsw.gov.au/teaching-and-learning/student-assessment/smart-teaching-strategies/literacy/writing/stage-1/spelling>  <https://www.enchantedlearning.com/wordlist/>  Crestwood Public School spelling lists |
| **28/1/19** | Started and completed the structure chart and the systems flowchart, both of which give a comprehensive view of how the bulk of the program (the quiz) functions.  Changed size policy of all the program’s windows so that layout does not lose its proper formatting | <https://www.draw.io/> |
| **30/1/19** | Identified and documented appropriate test data, and detailed the live tests that took place in order to ensure the user could not enter data that would disrupt the running of SpellZone  Refined consistency of object names throughout the GUI so that further coding would be simpler | Qt Designer |
| **31/1/19** | Completed instructions for students and teachers and included in documentation, showing them, in a simple and readable way, how to use SpellZone | Microsoft Publisher |
| **2/2/19** | Worked on the functionality of buttons so that the program can be used properly | Qt Designer |
| **3/2/19** | Continued work on the functionality of the buttons | Qt Designer  <https://www.youtube.com/watch?v=dRRpbDFnMHI> |
| **6/2/19** | Experienced great difficulty with the functionality of the buttons, especially regarding circular imports | Qt Designer |
| **10/2/19** | Changed the layout of the windows to a stacked widget to circumvent the issue of circular imports  Changed background from colourful to plain so that the program would run correctly and to improve legibility | Qt Designer |
| **16/2/19** | Added additional information in the documentation for the user regarding the software and hardware required to use SpellZone | N/A |
| **25/2/19** | Failed at properly implementing the functionality of a stacked widget, so the program does not work properly  Took screenshots of all the program’s windows and included them in the documentation, putting them in the order that they appear when the user uses the program | Gyazo (used to take screenshots) |
| **26/2/19** | Critically evaluated the project against the initially identified objectives, showing how SpellZone succeeded and failed in different areas  Identified potential modifications and discussed how they could be implemented, so as to assist the longevity of the program and its continued use in schools | N/A |
| **28/2/19** | Submitted assignment | N/A |

**Documentation for System Users**



The following tables show the words included in each level and theme available in SpellZone. These lists would be updated in the future when the program is refined.

**Themed Word Lists – Level 1**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ANIMALS | SCHOOL | COLOURS | BIRTHDAY | FOOD | BEACH | PARK | HOME |
| dog | school | red | cake | jam | beach | play | door |
| frog | year | green | food | apple | sun | walk | bed |
| bird | week | blue | party | egg | sea | run | floor |
| pig  snake | table  chair | yellow  black | fun  toy | toast  lunch | water  sand | grass  park | pet  food |
| goat | book | brown | play | cook | ball | path | toy |
| cow | paper | gold | games | eat | crab | skip | book |
| cat | neat | white | card | bread | hat | fun | bath |
| horse | class | pink | clown | jelly | ocean | sport | tap |
| lion | study | cream | child | nut | swim | tree | rug |

**Themed Word Lists – Level 2**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ANIMALS | SCHOOL | COLOURS | BIRTHDAY | FOOD | BEACH | PARK | HOME |
| owl | office | orange | friends | cheese | sandals | throw | window |
| deer | homework | crimson | family | banana | surfboard | catch | hallway |
| jaguar | recess | purple | parcel | sandwich | vacation | chase | computer |
| zebra | writing | olive | donkey | lettuce | relaxing | swing | shower |
| crocodile | library | indigo | balloon | dough | sunscreen | outdoor | doorbell |
| flamingo | assembly | grey | birthday | breakfast | rescue | garden | lounge |
| platypus | canteen | teal | candle | butter | goggles | reserve | backyard |
| kangaroo | playground | silver | cupcake | custard | bikini | fence | laundry |
| walrus  elephant | award  science | emerald  beige | festive  present | dinner  salad | dune  towel | soccer  cricket | chores  lawn |

**Themed Word Lists – Level 3**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| ANIMALS | SCHOOL | COLOURS | BIRTHDAY | FOOD | BEACH | PARK | HOME |
| narwhal | dictionary | mauve | gathering | casserole | snorkel | frisbee | wardrobe |
| porpoise | mathematics | chartreuse | invitation | ingredients | lifeguard | gymnastics | telephone |
| quokka | principal | cerulean | lollipop | recipe | underwater | adventure | loungeroom |
| caterpillar  giraffe | information  introduction | burgundy  khaki | chocolate  celebration | muesli  baclava | chlorine  scuba | activity  playground | awning  pergola |
| tortoise | conclusion | lavender | confetti | mayonnaise | current | wildlife | renovation |
| penguin | knowledge | maroon | guest | pumpkin | beachball | eucalyptus | balcony |
| newt | narrative | sapphire | pinata | mousse | sandbar | community | banister |
| vertebrates | stationery | turquoise | receive | dessert | sunglasses | facility | furniture |
| tapir | literature | puce | wrapping | rotisserie | sandcastle | mosquito | address |

**Additional Information for the User**

**Software**: the software is programmed to run on Windows 10 computers, as this is what the vast majority of school computers use in Australia.

**Hardware**: no special hardware is required to use the program. Apart from the computer itself, only the absolute basics are required, that is, a keyboard, a mouse, and a screen. All school computers have these (apart from laptops, where a touchpad will suffice instead of a mouse).