|  |
| --- |
| **4.14.5.1 Analysis**  Students are expected to:   * produce a clear statement that describes the problem area and specific problem that is being solved/investigated * outline how they researched the problem * state for whom the problem is being solved/investigated * provide background in sufficient detail for a third party to understand the problem being solved/investigated * produce a numbered list of measurable, "appropriate" specific objectives, covering all required functionality of the solution or areas of investigation (Appropriate means that the specific objectives are single purpose and at a level of detail that is without ambiguity) * report any modelling of the problem that will inform the Design stage, for example a graph/network model of Facebook connections or an E-R model.   A fully scoped analysis is one that has:   * researched the problem thoroughly * has clearly defined the problem being solved/investigated * omitted nothing that is relevant to subsequent stages * statements of objectives which clearly and unambiguously identify the scope of the project * modelled the problem for the Design stage where this is possible and necessary |

**Title**

Tipsy Hudson (kidz edition)

**Brief Description**

This is a 2-dimensional game where the player races across the road. The player is represented by Hudson, an avatar of your choice (but suitable for primary school students)

**User requirements outline**

A game where the character ‘Hudson’ has to cross a busy road with lots of traffic in both directions without being hit. The game works by giving the students an incentive: the quicker they pick up litter the longer the higher their score. Each level must get more difficult to complete in a time frame.

In between each level, questions should be asked. Level 1 has 3 simple road safety questions (these should be in a random order and there should be enough that they are not repeated unnecessarily. The answers should be displayed to the student on submission indicating correct or not, with some kind of output to indicate success or failure. Each correct answer should add one to the user’s score.

Level 2 has 5 simple legal drinking age questions.

Level 3 has 8 simple statics questions.

The total score should be displayed at the end.

If a student quits the game before the end, the score is not stored.

A list of all top scores should be available for the student to check.

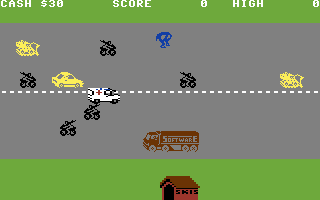
**Target Audience**

Primary school education

**Key features**

* 3 difficulty levels
* 3 lives to last the player through each of the levels (if they get hit by a lorry/train/car etc…)
* Keyboard control
* Multiple moving vehicles

**Background research (examples of existing ‘road-crossing’ games)**

In Horace goes skiing you play as Horace as you cross a road to buy skis and then go skiing.

I will keep the road crossing but not keep the skiing.

In crossy road you have to cross a procedurally generated highway. The game ends when you die

I will keep the idea of the road but will have a limited level layout.



In Kahoot you pick from 4 choices and get points depending on how fast you answer. I will keep the 4 options but will offer no bonus points for answering faster.

Graphical user interface, website

Description automatically generatedIn Quizizz you get to pick from 4 options but also have a variety of powerups to help you get more points. I will keep the multiple choice aspect but will not include powerups.

**Game Background (gameplay)**

There will be multiple cars that will drive in different directions along the lanes of the dual carriageway road. There will be one player character ‘Hudson’ who will be moved across the road by use of the space bar; similar to gameplay in flappy bird. If Hudson makes contact with any of the vehicles it will lose a life and be moved back to the start. The player has 3 lives; when all are lost the game is over. There should be 3 levels to progress through. So the quicker the level is completed the higher the score. The score for all 3 levels added together gives a final score. Every time the game is played, the user details should be stored, along with their score. A list of the top 5 scores (of all players) should be displayed at the end of each game. The date and time of the games played should be stored. How long a person has been at the top of the list should also be stored. The game is intended to be used over a period of time, and therefore there may be hundreds of students to be sorted into order of ‘best’ over time.

**Objectives (needs completing)**

1 Player

1.1 Player can move up

1.1.1 Player’s movement controlled by space.

1.1.3 Player cannot go off the edge of the screen.

1.2 Player can collide with ‘enemies’

1.3.1 Collision with a vehicle causes the player to lose a life.

1.3.1.1 The player respawns them at the spawn point

1.3.1.1.1 Player spawn point is not on a road, so the player doesn’t spawn and collide with vehicle.

1.3.1.1.2 Respawning freezes the game for a split second for impact.

1.3.1.1.3 Player colliding with vehicle with no lives remaining shows a ‘player loses’ screen

2 Vehicles

2.1 Cars and lorries are a variant of Vehicle.

2.1.1 Lorries have the slowest speed, are orange with fixed height and width.

2.1.2 Cars move faster than Lorries

2.1.2.1 Cars can be red, black or blue.

2.1.2.1.1 colour of car is randomly generated.

2.1.2.2 Cars have the same height but can have different lengths.

2.1.2.2.1 cars can be one of two lengths (short and long).

2.1.2.2.2 the car length is randomly generated.

2.2 Vehicles can move across the screen

2.2.1 One side of the road has vehicles moving from left to right.

2.2.2 Another side of the road has vehicles moving from right to left.

2.2.2 Vehicles spawn on screen and despawn off screen

2.2.2.1 They can spawn on set roads so that the player has rest spaces

3 Finish

3.1 The finish does not move

3.2 It always spawn in the same place.

4 Quiz

4.1 Quiz has a question box

4.1.1 The question is chosen from a dictionary of questions

4.2 Quiz has multiple answers to choose from

4.2.1 The player can pick the answer they think is correct

5 Menu

5.1 Menu has a play button

5.1.1 Pressing play starts the game

5.2 Menu has a leaderboard button

5.2.1 Pressing the leaderboard shows the leaderboad

5.3 Menu has a help button

5.3.1 Pressing help will show the player a quick runthrough on how to play the game

|  |
| --- |
| **4.14.5.2 Design**  Students are expected to articulate their design in a manner appropriate to the task and with sufficient clarity for a third party to understand how the key aspects of the solution/investigation are structured and on what the design will rely, eg use of numerical and scientific package libraries, data visualisation package library, particular relational database and/or web design framework.  The emphasis is on communicating the design; therefore it is acceptable to provide a description of the design in a combination of diagrams and prose as appropriate, as well as a description of algorithms, SQL, data structures, database relations as appropriate, and using relevant technical description languages, such as pseudo-code.  Where design of a user interface is relevant, screen shots of actual screens are acceptable. |

For each part of the project (game, quiz, login, etc) include the following:

UI/Forms/Display

Data dictionary

|  |  |  |  |
| --- | --- | --- | --- |
| **Variable Identifier** | **Datatype** | **Validation** | **Description** |
| LivesRemaining | Byte | Between 0 and 3 | Used in the game play to indicate how many lives are remaining before the game is over. The player starts with 3 and the game ends when it becomes 0. |
|  |  |  |  |

Data structures used (include reasons why these are appropriate)

Flowchart(s) and/or pseudocode and/or real code (to represent algorithms and other structures)

Data Flow Diagrams (to show how data is processed, including input and outputs)

Database design (ERD) showing relationships between tables with primary and foreign keys

|  |
| --- |
| **4.14.5.3 Technical solution**  Students should provide program listing(s) that demonstrate their technical skill.  The program listing(s) should be appropriately annotated and self-documenting (an approach that uses meaningful identifiers, with well structured code that minimises instances where program comments are necessary).  Students should present their work in a way that will enable a third party to discern the quality and purpose of the coding. This could take the form of:   * an overview guide which amongst other things includes the names of entities such as executables, data filenames/urls, database names, pathnames so that a third party can, if they so desire, run the solution/investigation * explanations of particularly difficult-to-understand code sections; a careful division of the presentation of the code listing into appropriately labelled sections to make navigation as easy as possible for a third party reading the code listing.   Achievement of the latter, to an extent, is linked to the skill in applying a structured approach during the course of developing the solution or carrying out the investigation. |

|  |
| --- |
| **4.14.5.4 Testing**  Students must provide and present in a structured way for example in tabular form, clear evidence of testing.  This should take the form of carefully selected and representative samples, which demonstrate the robustness of the complete, or nearly complete, solution/thoroughness of investigation and which demonstrate that the requirements of the solution/investigation have been achieved.  The emphasis should be on producing a representative sample in a balanced way and not on recording every possible test and test outcome.  Students should explain the tests carried out alongside the evidence for them.  This could take the form of:   * an introduction and overview * the test performed * its purpose if not self-evident * the test data * the expected test outcome * the actual outcome with a sample of the evidence, for example screen shots of before and after the test, etc, sampled in order to limit volume. |

|  |
| --- |
| **4.14.5.5 Evaluation**  Students should consider and assess how well the outcome meets its requirements.  Students should obtain independent feedback on how well the outcome meets its requirements and discuss this feedback.  Some of this feedback could be generated during prototyping. If so, this feedback, and how/why it was taken account must be presented and referenced so it can be found easily.  Students should also consider and discuss how the outcome could be improved more realistically if the problem/investigation were to be revisited. |