Project Title:

BillyMaths

Centre Number:

64910

Candidate Name:

Billy George Andrew Elderfield

Candidate Number:

6419

Index

**1. ANALYSIS**

1.1 Background/Identification of Problem and Identification of Prospective Users

1.2 Research

1.3 Modelling

1.4 Objectives

**2. DOCUMENTED DESIGN**

2.1 Overall System Design

2.2 Data Dictionary

2.3 Validation

2.4 Files

2.5 Algorithms

2.6 Data Structures

2.7 User Interface

**3. TECHNICAL SOLUTION**

3.1 Program code

**4. TESTING**

4.1 Test Plan

4.2 Test Evidence

**5. APPRAISAL**

5.1 Evaluation of objectives

5.2 Evaluation of whole solution compared with the start

5.3 User Feedback

5.4 Analysis of user feedback

5.5 Extensions and Improvements

**6 Appendix**

6.1 Interview transcripts

6.2 Internet search

1.Analysis

1.1 Background/Identification of Problem and Identification of Prospective Users

Sarah Elderfield is my first user who is a teaching assistant at a middle school. She is my mother and I will have enough time to review my project with her, as we live together. Her area that would benefit from being computerised would be a system that aids students' learning of maths and an alternative to teachers setting work via paper. Feedback obtained from Sarah would be useful because she can give an insight into the way teachers set work and view students' progress and what tools they would find helpful to make their job easier.

Kirsty Elderfield is my second user who is a student at a middle school. She is my sister and I will have enough time to review my project with her, as we live together. She struggles with maths often and feels that the written content that they must work on for homework is not entirely helpful in achieving a full understanding in a certain field of maths. She also slightly lacks the incentive to learn to the maximum of her capacity. Feedback obtained from Kirsty would be useful because she can give an insight into the way students complete the work and what tools they would find helpful to further their learning.

My users have problems related to getting students to do their work and having the resources needed to further students' understanding in certain topics of maths. This is because the current system has little in the way of monitorisation and help for students outside the classroom. My program will allow students to do practice questions and compete in multiplayer 1v1 duels where users try to complete a set of identical questions on a certain topic in the fastest time possible. Information about the worksheets and duels they complete will be viewable on their individual profiles and on the global leader board. This will create a competitive atmosphere where students will hopefully feel incentivized to improve their mathematical skills through repetition of maths questions so that they can have a higher rank than their friends/class.

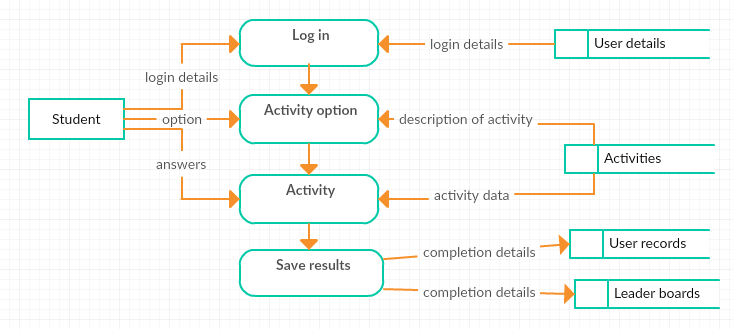
1.2 Research

For my research I interviewed my clients extensively and looked up multiple sites that already provide a similar service to that which I hope my program will do. I obtained a wealth of information directly from my clients and from the sites, which I found useful in deciding the direction I wish to take my program. The main points which I took from my research are that I need to: Have a simple but mature user interface and aesthetics, focus solely on key stage 3 topics and be as in-depth about them as possible, create revision practice tools where students can complete worksheets on certain topics, have a 1v1 duel mode that creates a competitive atmosphere and develop students’ mental maths speed, have a leader board system that would engage students to challenge themselves.

1.3 Modelling

DFDs

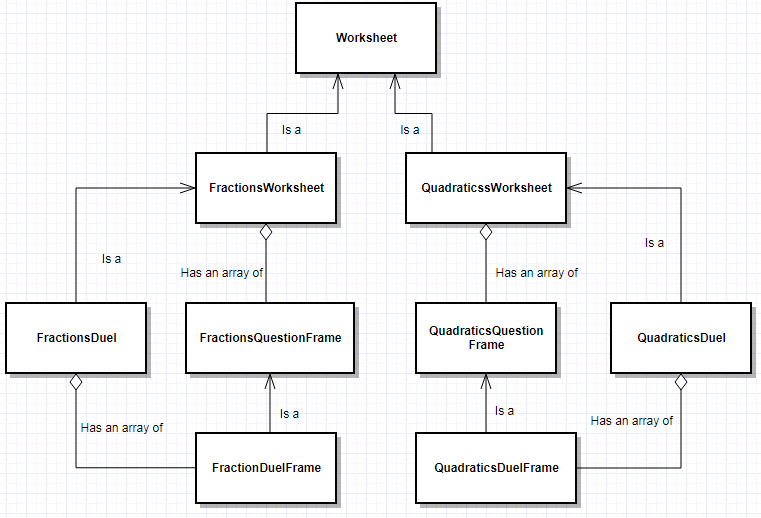
Student completing an activity



This DFD represents the data flow when a user logs in, chooses an activity and completes it. The login process requires input from the user for their username and password, it also needs their encrypted password and username from the user details data store. The login process then compares the usernames and passwords and if they fail to match, informs the user of the incorrect field and if they do match, then the user will have access to the program. The user will then navigate through the different activity options available and choose one. The descriptions and names of the activities will be fetched from the Activities data store and displayed to the user. The user must input their choice of activity to then proceed in completing that activity. The activity itself will need attributes from the Activities data store to generate a set of questions for the practice question sheet or game, also the user must input answers to the questions. The Save results process simply updates the User records data store with information on the time spent and marks achieved. If the activity was a game, then the Leader boards data store would also be updated.

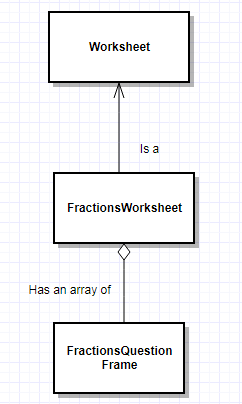
**Class Diagrams**

Overall use of worksheets and duels



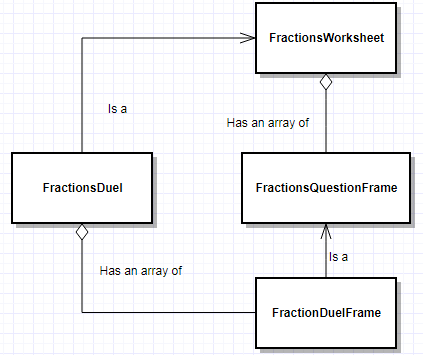
An overview of how the duels and worksheets are constructed.

Use of worksheets



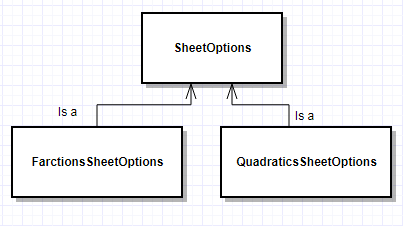
The worksheet base class includes all the general UI and methods including: a method for marking a worksheet and displaying results, also a method for changing to the next/ previous pages. The FractionsWorksheet then has to grid all its questions in a different way to other worksheets, so that is why it needs its own class. The other reason is because it requires different question frames, these question frames are put into an array to be gridded in order and after the worksheet is complete the runs through each element in the array to mark each question.

Use of duels



A duel is structured with all the UI of a worksheet without the menu bar at the top to change page and mark the questions. Instead there is only one page because the maximum number of questions in a fractions duel is 20 and that fits on one page. The marking is also carried out every 0.5 seconds automatically so there is no need for the user to mark the work. The question frame is also different because instead of having to generate the values for the question, the server supplies the values so these need to be handled in the inherited class. Also, when the questions are marked, they would not be corrected until the duel is over.

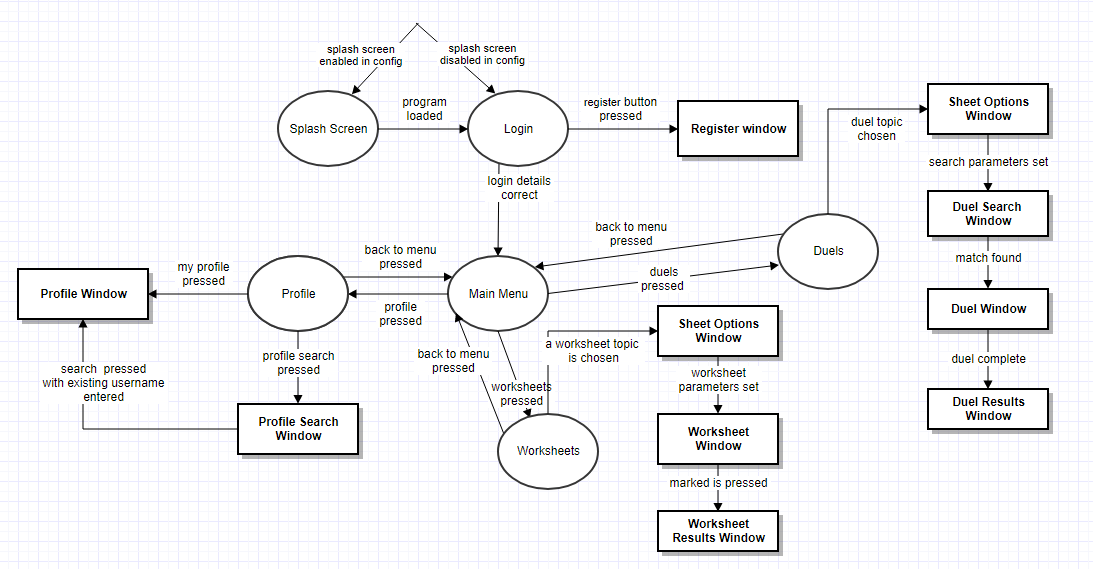
User of sheet option windows

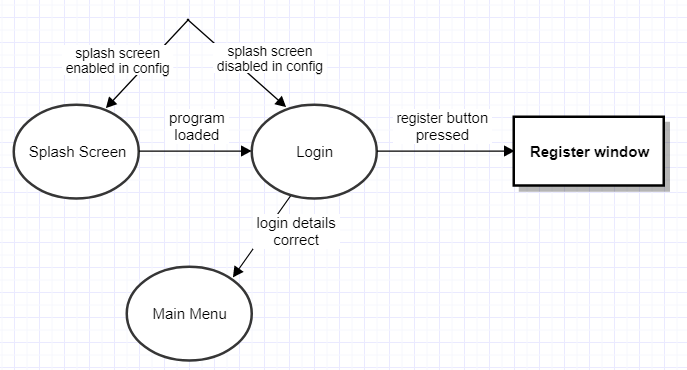


The base SheetOptions class includes the necessary attributes for each child class (question number and selected functions) and the generate method used by all child classes to create the necessary worksheet or duel. The fractions and quadratics sheet options both have different types of questions for example: fractions questions can be (addition, subtraction multiplication or division), the quadratics questions can either be expanding or factorising.

**State transition diagrams**

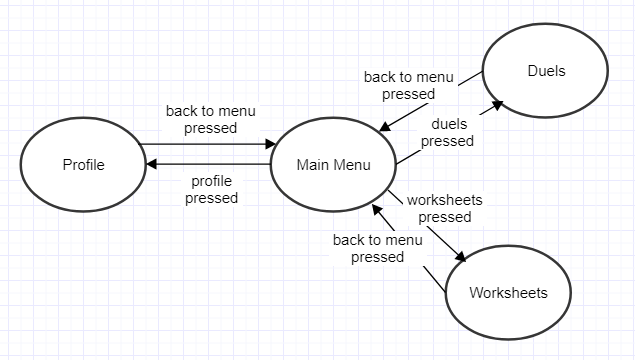
(ellipses represent a change of state for the root window, rectangles represent a new window being created.)

Overview of whole program

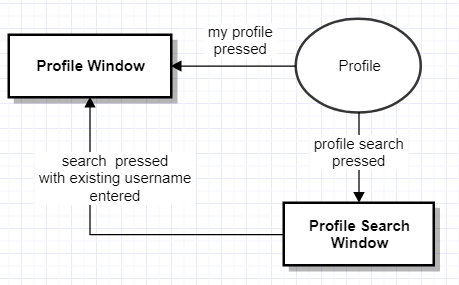
Login process

If the user sets splash\_screen\_enable = 1 before launching the program the splash screen will load, otherwise the program will load straight to the login frame where the user is required to enter a correct username and password. If the splash screen is loaded, then the login frame will not be displayed until all the main widgets of the program are loaded. When on the login frame, if the user presses the register button a window where a user can create a new account is created. If the user presses the return key on the login frame and has a correct username and password entered, the program will load the main menu on the root window.

Menu navigation

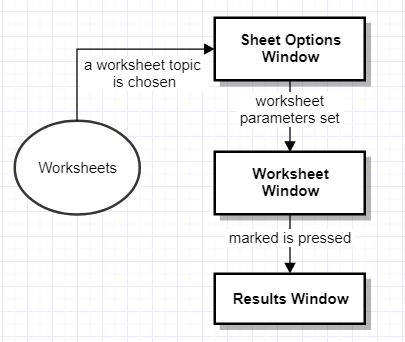
The main menu frame navigates to each sub menu frame where users can then also navigate back to the main menu.

My profile and profile search



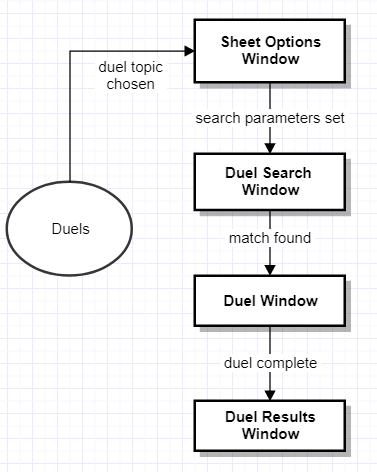
When a user is on the Profile submenu they can press the My Profile button to load a Profile window where they can view their stats and history, they can also change their profile picture. When profile search is pressed a Profile Search window is created where a user can enter the username of another existing user, a Profile window with the entered user’s stats and history is then displayed.

Worksheet generation and marking



When a user is on the worksheets submenu they can choose a topic to do a worksheet on, then they choose how many questions they wish to complete and the types of question they are going to complete on the sheet options window. Then after they press generate a worksheet with the specified questions is generated. After a user has completed a worksheet and presses mark, a window with the results of the worksheet along with the time taken is displayed.

Duel searching and completion



When a user is on the duels submenu and they can chose a topic to search for, then they chose the number of and types of question they wish to do in the sheet options window. The duel search window is then displayed whilst the matchmaking server finds another user to duel with. When a match is found the duel window will appear where the user completes a set of questions and can see their opponent’s progress. When either user completes all the questions in a duel, the duel will end and a results window that displays whether the user won or lost, their opponent’s username, elo and profile picture and the time taken.

**Calculations**

I need to research how to:

Use a competitive elo system.

Systematically generate values for fractions questions suitable for KeyStage3 students.

Systematically calculate simplified fractions using (+ - \* /) operators.

Systematically generate quadratic expressions in a simplified, expanded state where they can be factorised by keyStage3 students.

Systematically generate quadratic expressions in a simplified, factorised state where they can be expanded by keyStage3 students.

Systematically calculate simplified expanded and factorised quadratic expressions.

All these calculations are explained under the (2.2 SQL design) section.

* 1. Objectives

1. Each user will have a unique user\_id, username, password, name, email, school, postcode and competitive elo rating stored.
2. The password will be encrypted so that even if the database is compromised, the passwords are still relatively safe.
3. Make it possible to host the database separately form the client and the client connects to the database through a database server
4. Worksheet results from every worksheet users complete should be saved to the database in a separate table with information on: the time taken, topic the questions are on, types of question used, number of questions correct, number of questions total and the date which the worksheet was completed on.
5. Duel results from every duel users complete should be saved to the database in a sperate table with information on: the time taken, topic the questions are on, types of questions used, number of questions, winner’s user\_id, loser’s user\_id and the date which the worksheet was completed on.
6. Have worksheets where users practice mathematical questions.
7. Have a marking feature for the worksheets that tells the user which questions they have correct.
8. Have the marking feature show the correct answer next to incorrect answers.
9. Have the marking feature display a tick next to correct answers and crosses next to incorrect answers.
10. Have a results window appear after marking a question that displays the user’s score out of how many questions there were
11. Have the results window display a grade from A-D based on the % of correct answers they have.
12. Have the results window display the time taken to complete a worksheet.
13. Generate worksheets on fractions.
14. Generate questions for adding fractions.
15. Generate questions for subtracting fractions.
16. Generate questions for multiplying fractions.
17. Generate questions for dividing fractions.
18. Have a 1v1 duel mode
19. Have a competitive elo system for winning and losing duels
20. Have the elo of each user involved updated after a duel.
21. Have a results window at the end of a duel that displays your opponent’s username
22. Have the duel results window display whether you have won or lost.
23. Have the duel results window display your opponent’s profile picture
24. Have the duel results window display your opponent’s elo.
25. Have the duel results window display your old elo, the change in elo based on the outcome of the game and the elo of your opponent and your new elo.
26. Make the duel mode work over the internet using a client-server model
27. Make the duel mode automatically mark each question as the user completes questions
28. Make the marking feature not display a cross or the correct answer next to the question for duels
29. Have duels on the fractions topic
30. Generate worksheets on quadratics.
31. Generate questions for factorising quadratic expressions
32. Generate questions for expanding quadratic expressions
33. Have duels on the quadratics topic
34. Users can register new accounts with their own username, password, email, postcode, school code and real name.
35. Users can log in with their own account using their username and password.
36. Have a flat colour scheme
37. Make it so that the user can chose the colour theme they wish to be on the main menu and profile windows by changing the settings in a config file before launching the program.
38. Make a splash screen appear whilst the program loads all necessary widgets
39. Make it so that the splash screen can be enabled/disables by changing the settings in a config file before launching the program
40. Create a profile page where users can view their elo.
41. Have the profile page display the user’s duel results.
42. Have the profile page display the user’s worksheet results.
43. Make a click on the tabs of each column in a history table sort the table in alphabetical order of the clicked-on column
44. Make a double click on the tabs of each column in a history table sort the table in reverse alphabetical order of the clicked-on column
45. Each user should have a profile picture
46. Each user should be able to change this profile picture on their own profile by clicking on their profile picture.
47. An image of a edit symbol should appear of a user’s profile picture when a user’s cursor is over the picture.
48. Host the profile pictures on a website where each image has the end of the url as their user\_id.
49. Have the pictures uploaded by users resized to 128x128 before uploading without changing the source file chosen.
50. Have a default profile picture uploaded for new users.
51. Have a leader board where users can see their rank globally and among their classmates.
52. Have the main parts of the program unaccusable without logging in with a valid username and password.
53. Have a teacher setting work system utilising the worksheet functionality.
54. Have the teacher allocate classes to students.
55. Include a range of topics for students to practice on.
56. Include revision tools such as worked examples.
57. Include revision tolls such as links to revision material.

2.Design

2.1 Overall System Design

BillyMaths was coded in (python 3.6.2) using the (pycharm community edition 5.0) IDLE. SQLite3 was also used with (sqlitestudio 2.1.5). I used modules that come with python as well as certain external modules that enabled me to fulfil my objectives.

**internal modules:**

tkinter-

A Python binding to the Tk GUI toolkit. I use for the whole of the program’s GUI.

random-

Used to generate relatively reliable pseudo random numbers. I use to generate the numbers for questions.

fraction-

For manipulating numbers in terms of their numerators and denominators. I use for generating and marking fractions questions.

socket-

Provides access to the BSD socket interface, available on all modern Unix systems, Windows, Mac OS X, BeOS, OS/2. Used for sending and receiving data between the client and database/game servers.

ast-

Helps Python applications to process trees of the Python abstract syntax grammar. I only use ast.literal\_eval() which takes lists and tuples as a string data type and returns them as a list or a tuple data type. This is used for whenever a list/tuple is sent via a socket as a bytes data type, which then is changed to a string data type and finally a list/tuple data type using ast.literal\_eval.

time-

Provides various time-related functions. Used primarily to Return the time in seconds since the epoch as a floating point number, which i use for time taken calculations.

datetime-

Supplies classes for manipulating dates and times in both simple and complex ways. Used primarily for returning the current date that is stored as part of worksheet/duel results.

re-

Provides regular expression matching operations. Used for the validation of entry fields.

threading-

Constructs higher-level threading interfaces on top of the lower level thread module. Applied to various circumstances that require two or more functions to be running simultaneously on the same python script, such as: When the game server has duels in progress in the game() function, as well as receiving new connections, matching them up and starting new games in the matchmaking() function.

ftplib-

Creates a FTP connection. Used to upload users’ profile pictures.

os-

Provides a portable way of using operating system dependent functionality. Used for checking the existance of file paths and deleting files.

json-

Lightweight data interchange format inspired by JavaScript object literal syntax. Used for manipulating data returned by APIs.

sqlite3-

Used to connect to, query and change data in a database file.

**external modules:**

passlib-

A password hashing library, which provides cross-platform implementations of over 30 password hashing algorithms. Used to encrypt users’ passwords stored in the database.

Pillow(PIL)-

Python Imaging Library. Used for manipulating users’ profile pictures before being uploaded.

urllib.request-

Defines functions and classes which help in opening URLs. Used to load load users’ profile pictures and receive data from APIs.

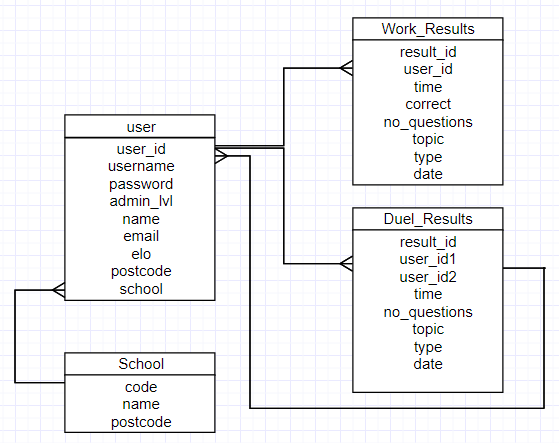
base64-

Provides functions for encoding binary data to printable ASCII characters and decoding such encodings back to binary data. Used for manipulating data returned by url requests.

|  |  |
| --- | --- |
| Class/Function (script) | Brief Description |
| class: MathsGui (client) | The frame that is applied to the root window. (essentially the main menu of the program) |
| parent class: SheetOptions (client)  child class: FractionsSheetOptions  child class: QuadraticSheetOptions | The window where the user inputs the parameters needed to generate a worksheet or search for a duel. |
| parent class: Worksheet (cleint)  child class: FractionsWorksheet  child class: QuadraticWorksheet | The window where users carry out practice questions. |
| class: DuelSearch (client) | Handles the connection to the game server whilst matchmaking and displays matchmaking UI. |
| function: matchmaking (game server) | Listens for new connections to the game server, keeps the current searching connections up to date and starts a new game thread when a pair of players searching have matching search parameters. |
| fucntion: game (game server) | Handles two client connections on a separate thread, sending the generated questions, updating each other’s’ scores. |
| parent classes: (client)  FractionsWorksheet/QuadraticsWorksheet  child classes:  FractionsDuel/QuadraticsDuel | The window where users carry out duels, it also handles the connection to the game server whilst a duel is in progress and handles the resulting elo change and database changes at the end of a duel. |
| function: db (client) | Handles the client’s connection to the database server. |
| function: connection\_handler (database server) | Handles the connection to the client and runs functions based on the client connection’s queries. |
| class: Profile (client) | A window that displays a user’s information, profile picture and worksheet/duel history. |
| class: ProfileSearch (client) | A window where the user searches for another user and if that user exists, the window is destroyed and a window with their profile on it is created. |
| class Register (client) | A window where a new user can enter all their required information to make an account. If all the fields that can be locally validated are valid, then the data entered is sent off to the database server to be further validated against existing users’ data. If all the user’s information entered is valid then the user’s information is uploaded to the database, the user is prompted and is directed back to the login window. If the information is not valid, the invalid fields are pointed out to the user. |

2.2 Data Dictionary

Entity relationship diagram between the tables in my database:



Each user can have one school, many worksheet results and many duel results.

Each school can have many users.

Each worksheet result can have one user.

Each duel result can have many users (winner and loser).

**Tables**

**User**

This table contains all the information that the user entered when they registered, as well as their current elo rating and admin level that is set to 1 for normal users. Data from this table is displayed in the user information section of a user’s profile, their elo is used for rankings on the leaderboard.

The expected number of records is however many users sign up and the expected number of users that will sign up is however many schools decide to use this program multiplied by the average amount of students in each school. So if we take the average amount of students in each school to be 400 and between 1-20 schools use the program, then the expected number of users would range between 400-8000.

The maximum number of records would be purely based on the size of the hard drive that contains the database; I have researched sources that say SQL should not cause any significant losses in performance speed due to large tables, if there is a good use of primary keys and overall data structure. My program also never has to manipulate every single piece of data from this table as that would also cause lower performance speeds.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field Name | Data Type | Size (bytes) | Key? | Validation Rules | Look Up values | Purpose/Comments/Other |
| user\_id | integer | 8 | Primary key | Required |  | unique identifier. (auto increments) |
| username | varchar | 16 |  | Required, Unique, between 3-16 characters |  | User’s username. |
| password | varchar | 80 |  | Required |  | Encrypted password using passlib. |
| admin\_lvl | integer | 8 |  | Required | 1/2 | Determines between a standard user and an admin, defaults to 1. |
| name | varchar | 33 |  | Required, between 6-33 characters |  | user’s first and second name. |
| email | varchar | 40 |  | Required |  | User’s email. |
| school | varchar | 3 | Foreign key | Required |  | User’s school code. (relates to school name and postcode in school table) |
| postcode | varchar | 10 |  | Required |  | User’s postcode. |
| elo | varchar | 8 |  | Required |  | User’s competitive skill rating, defaults to 1000. |

Size of one record for the user table above is: 8+16+80+8+8+33+40+3+10+8 = 214 bytes

Therefore, the expected table size would be between: (214 x400) - (214x8000) =

Between 85.6 Kbytes – 1.7 Gbytes

**School**

This table holds all the information on the different schools that use this program. The school that a user goes to is displayed in the user information section of their profile.

The expected number of records is however many schools decide to use this program. Which would be between 1-20.

The maximum number of records would be purely based on the size of the hard drive that contains the database; I have researched sources that say SQL should not cause any significant losses in performance speed due to large tables, if there is a good use of primary keys and overall data structure. My program also never has to manipulate every single piece of data from this table as that would cause lower performance speeds.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field Name | Data Type | Size (bytes) | Key? | Validation Rules | Look Up values | Purpose/Comments/Other |
| Code | varchar | 3 | Primary key | Required, Unique |  | Unique identifier, used so that only users from a certain school can register because only they should know their school code. |
| name | varchar | 40 |  | Required, Unique |  | Name of the school. |
| postcode | varchar | 10 |  | Required |  | Postcode of the school. |

Size of one record for the user table above is 3+40+10 = 53 bytes

Therefore, the expected table size would be between: (53 x 1) - (53 x 20) =

Between 53bytes – 1.06 Kbytes

**Work\_Results**

Whenever a user marks a worksheet the information regarding their completion is stored in this table. Users can access their results at the worksheet history section of their profile.

The expected number of records is proportional to however many worksheets are completed, so this number will increase over time. The amount that this number will increase by over time is proportional to how many users use the program. So if we have the expected amount of users to be 400-8000 and each user completes an average of 5 worksheets a week, then the number of records in this table would increase by 2000-40000 each week. After a year the number of records in this table would be 104000-2080000.

The maximum number of records would be purely based on the size of the hard drive that contains the database; I have researched sources that say SQL should not cause any significant losses in performance speed due to large tables, as long as there is a good use of primary keys and overall data structure. My program also never has to manipulate every single piece of data from this table as that would cause lower performance speeds.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field Name | Data Type | Size (bytes) | Key? | Validation Rules | Look Up values | Purpose/Comments/Other |
| Result\_id | integer | 8 | Primary key | Required, unique |  | Unique identifier. |
| User\_id | integer | 8 | Foreign key | Required |  | User that completed the worksheet. |
| Time | Numeric(float) | 8 |  | Required |  | Time taken to complete the worksheet. |
| Correct | integer | 8 |  | Required | Between 1-99 | Number of questions user got correct. |
| No\_questions | integer | 8 |  | Required | Between 1-99 | Number of questions total in the worksheet. |
| Topic | varchar | 20 |  | Required |  | The topic that the worksheet was about. |
| type | varchar | 40 |  | Required |  | The different types of questions in the worksheet. |
| date | varchar | 40 |  | Required |  | The date and time at which the worksheet was completed on |

Size of one record for the user table above is: 8+8+8+8+8+20+40+40 = 140 bytes

Therefore, the expected table size after a year would be between: (140 x 104k) - (140 x 20800k) =

Between 14.5 Gbytes – 291 Gbytes

**Duel\_Results**

Whenever two users complete a duel the information regarding the results is stored in this table. Users can access their results at the duel history section of their profile.

The expected number of records is proportional to (however many duels are completed / 2), so this number will increase over time. The amount that this number will increase by over time is proportional to how many users use the program. So if we have the expected amount of users to be 400-8000 and each user completes an average of 10 duels a week, then the number of records in this table would increase by 2000-40000 each week. After a year the number of records in this table would be 104000-2080000.

The maximum number of records would be purely based on the size of the hard drive that contains the database; I have researched sources that say SQL should not cause any significant losses in performance speed due to large tables, as long as there is a good use of primary keys and overall data structure. My program also never has to manipulate every single piece of data from this table as that would cause lower performance speeds.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Field Name | Data Type | Size (bytes) | Key? | Validation Rules | Look Up values | Purpose/Comments/Other |
| Result\_id | integer | 8 | Primary key | Required, unique |  | Unique identifier. |
| User\_id1 | Integer | 8 | Foreign key | Required |  | User that won the duel. |
| User\_id2 | integer | 8 | Foreign key | Required |  | User that lost the duel. |
| Time | Numeric(float) | 8 |  | Required |  | Time taken to complete the duel |
| No\_questions | integer | 8 |  | Required | Between 1-20 | Number of questions in the duel |
| Topic | varchar | 40 |  | Required |  | The topic that the duel was about. |
| Type | varchar | 40 |  | Required |  | The different types of questions in the duel. |
| date | varchar | 40 |  | Required |  | The date and time at which the duel was completed on |

Size of one record for the user table above is: 8+8+8+8+8+40+40+40 = 160 bytes

Therefore, the expected table size after a year would be between: (160 x 104k) - (160 x 20800k) =

Between 22.4 Gbytes – 488 Gbytes

Some of these table sizes appear rather large, however because the database is hosted by me, the database would have a dedicated hard drive and each client does not need the database file.

**SQL design**

Username check for login

|  |  |
| --- | --- |
| Purpose | Check entered username exists in the database for login. |
| Used where | db\_login() in the database server |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| SELECT | username | Check against all usernames |
| FROM | user | In the user table |

Password check for login

|  |  |
| --- | --- |
| Purpose | Check that a password entered for a certain username is correct. If the password matches, then the whole record is sent to the client with the password taken out for safety. |
| Used where | db\_login() in the database server |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| SELECT | \* | Get all the data |
| FROM | user | In the user table |
| WHERE | username = (username user entered) | Only where the username is the same as the username entered. |

Username and email check for registering

|  |  |
| --- | --- |
| Purpose | Check if a user already has the entered username and email. |
| Used where | Db\_register() in the database server |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| SELECT | username, email | Check against every username and email. |
| FROM | user | In the user table |

Create new account

|  |  |
| --- | --- |
| Purpose | Create a new account with the data that the user entered. |
| Used where | Db\_register() in the database server |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| INSERT INTO | User (username, password, name, email, school, admin\_lvl, postcode, elo) | We wish to enter data into the user table under the fields: username, password, name, email, school, admin\_lvl, postcode, elo |
| VALUES | username, password, name, email, school, 1, postcode, 1000 | The values that the user entered for: username, password, name, email, school and postcode. Admin\_lvl is set to 1 and elo is set to 1000 |

Get the user\_id of newly created account

|  |  |
| --- | --- |
| Purpose | Get the user\_id of newly created account to upload their placeholder profile picture to the right webpage. |
| Used where | Db\_register() in the database server |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| SELECT | User\_id | Get only the user\_id. |
| FROM | user | In the user table. |
| WHERE | Username = (username of the newly created account) | Only for the record that has the same username as the newly created account. |

Save worksheet results

|  |  |
| --- | --- |
| Purpose | Save the data related to a completed worksheet. |
| Used where | Db\_save\_results() in the database server |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| INSERT INTO | Work\_results (user\_id, time, correct, no\_questions, topic, type, date) | We wish to enter data into the work\_results table under the fields: user\_id, time, correct, no\_questions, topic, type and date. |
| VALUES | user\_id, time, correct, no\_questions, topic, type, date | The data related to a completed worksheet. |

Save duel results

|  |  |
| --- | --- |
| Purpose | Save the data related to a completed duel. (Only the winner runs this query) |
| Used where | Db\_save\_duel\_results() in the database server |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| INSERT INTO | duel\_results (user\_id1, user\_id2, time, no\_questions, topic, type, date) | We wish to enter data into the duel\_results table under the fields: user\_id1, user\_id2, time, no\_questions, topic, type, date. |
| VALUES | user\_id1, user\_id2, time, no\_questions, topic, type, date | The data related to a completed duel. |

Update elo

|  |  |
| --- | --- |
| Purpose | Update the elo of a user after a duel with their newly calculated elo. |
| Used where | Db\_update\_elo() in the database server |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| UPDATE | user | We wish to update data within the user table. |
| SET | Elo = (new elo) | Change the elo to the newly calculated elo. |
| WHERE | User\_id = (user\_id of user) | Only in the record that has the same user\_id as the specific user. |

Get user’s elo.

|  |  |
| --- | --- |
| Purpose | Get a user’s elo so that it can be displayed to them while they queue for a duel |
| Used where | \_\_init\_\_() in DuelSearch in the client |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| SELECT | elo | Get only the elo |
| FROM | user | From the user table |
| WHERE | User\_id = (user\_id of currently logged in user) | Only get data form the record that has the same user\_id as the currently logged in user |

Get opponent’s information

|  |  |
| --- | --- |
| Purpose | After a duel get the opponent’s information to be displayed in the results window. |
| Used where | \_\_init\_\_() in GameOver in the client |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| SELECT | Username, elo | Get only the username and elo |
| FROM | user | From the user table |
| WHERE | User\_id = (user\_id of opponent) | Only get data form the record that has the same user\_id as the user’s duel opponent. |

Get username of winner

|  |  |
| --- | --- |
| Purpose | After a duel get the winner’s username is returned so that it can be saved in the duel\_results table. (Only the winner runs this query) |
| Used where | \_\_init\_\_() in GameOver in the client |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| SELECT | Username, elo | Get only the username |
| FROM | user | From the user table |
| WHERE | User\_id = (user\_id of currently logged in user) | Only get data from the record that has the same user\_id as the winner of the duel. |

Get username of loser

|  |  |
| --- | --- |
| Purpose | After a duel get the loser’s username is returned so that it can be saved in the duel\_results table. |
| Used where | \_\_init\_\_() in GameOver in the client |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| SELECT | Username, elo | Get only the username |
| FROM | user | From the user table |
| WHERE | User\_id = (user\_id of opponent) | Only get data from the record that has the same user\_id as the loser of the duel. (Only the winner runs this query) |

Get worksheet history data

|  |  |
| --- | --- |
| Purpose | Get worksheet history data related to the user whose profile is being viewed so that is can be displayed in the worksheet history treeview. |
| Used where | \_\_init\_\_() in Profile in the client |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| SELECT | Topic, type, correct, no\_questions, time, date, result\_id | Get only the Topic, type, correct, no\_questions, time, date, result\_id fields. |
| FROM | Work\_results | From the work\_results table. |
| WHERE | User\_id = (user\_id of whose profile is being viewed) | Only get data from the record that has the same user\_id as whose profile is being viewed. |

Get duel history data

|  |  |
| --- | --- |
| Purpose | Get duel history data related to the user whose profile is being viewed so that is can be displayed in the duel history treeview. |
| Used where | \_\_init\_\_() in Profile in the client |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| SELECT | Topic, type, user\_id1, user\_id2, time, date, result\_id | Get only Topic, type, user\_id1, user\_id2, time, date, result\_id fields. |
| FROM | Duel\_results | From the duel\_results table. |
| WHERE | User\_id = (user\_id of whose profile is being viewed) | Only get data from the record that has the same user\_id as whose profile is being viewed. |

Get school name

|  |  |
| --- | --- |
| Purpose | Get school name of the user whose profile is being viewed to be displayed in the user information section. |
| Used where | \_\_init\_\_() in Profile in the client |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| SELECT | name | Get only the name field. |
| FROM | school | From the school table. |
| WHERE | Code = (school code of the user whose profile is being viewed) | Only get data from the record that has the same code as the school code of whose profile is being viewed. |

Get user data (profile search)

|  |  |
| --- | --- |
| Purpose | Get user data about a user that has been searched for. |
| Used where | Search() in ProfileSearch in the client |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| SELECT | User\_id, username, admin\_lvl, name, school | Get only User\_id, username, admin\_lvl, name and school fields. |
| FROM | user | From the user table. |
| WHERE | Username = (username of searched for user.) | Only get data from the record that has the same username as the username being searched for. |

Get elo and username for leader board

|  |  |
| --- | --- |
| Purpose | Get user data about all users that can be used for the leader board |
| Used where | Db\_get\_leaderboard() in the database server |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| SELECT | Username, elo | Get only the username and elo fields. |
| FROM | user | From the user table. |
| ORDER BY | elo | Order all the data in relation to elo. |
| DESC |  | Order in descending order. |

Get number of games won

|  |  |
| --- | --- |
| Purpose | Get the number of games a specific user has won. |
| Used where | Db\_get\_leaderboard() in the database server |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| SELECT COUNT | Result\_id | Count only fields with result\_ids. |
| FROM | Duel\_results | From the duel\_results table. |
| WHERE | User\_id1 = user\_id of specific user | Only get data from the records that have the same user\_id as the specific user’s user\_id |

Get number of games lost

|  |  |
| --- | --- |
| Purpose | Get the number of games a specific user has lost. |
| Used where | Db\_get\_leaderboard() in the database server |

|  |  |  |
| --- | --- | --- |
| SQL code | Parameters | Comments |
| SELECT COUNT | Result\_id | Count only fields with result\_ids. |
| FROM | Duel\_results | From the duel\_results table. |
| WHERE | User\_id2 = user\_id of specific user | Only get data from the records that have the same user\_id as the specific user’s user\_id |

**Variables**

|  |  |  |
| --- | --- | --- |
| Variable | Data Type | Purpose |
| Host | Constant string | The IP address of where the database and game servers are being hosted. For testing purposes, it is set to the IPv4 of the same machine the client is running on, however in the real word it would be the public IP address of where the servers are hosted. |
| Root | Tkinter.Tk  object | The root window of the program that contains the menu UI and what all subsequent windows are bound to. |
| w (set\_window) | Integer | The desired width of the window. |
| h (set\_window) | Integer | The desired height of the window. |
| Window (set\_window) | tkinter.Toplevel object | The desired window to be resized and positioned. |
| X (set\_window) | Integer | The x co-ordinate of the user’s screen that would place the window in the middle of the user’s screen with the desired width. |
| Y (set\_window) | Integer | The y co-ordinate of the user’s screen that would place the window in the middle of the user’s screen with the desired width. |
| Entry (limit\_size) | tkinter.Entry  object | The entry widget that is to have the size of its contents limited. |
| Limit (limit\_size) | Integer | The desired size limit of the contents of the entry. |
| Value (limit\_size) | String | The contents of the entry before being limited. |
| Func (db) | String | Indicates the function you wish the database server to run. |
| \*args (db) | List | Includes all the required data that is to be processed by the database server. |
| S (db) | Socket.socket  object | Create a new socket. |
| Port (db) | Integer | The desired port that will connect to the correct server at the host IP address. |
| Msgr (db) | String | The data received from the database server. |
| Msgs (db) | Bytes | The data sent to the database server in bytes because you can only send data over sockets in the bytes data type. |
| Text (db) | String | The string format of what is to be sent to the database server. |
| Master/self.master (all classes) | Tkinter.Tk  object | The root window. |
| Self.sc\_image (SplashScreen) | Tkinter.PhotoImage  object | The BillyMaths logo with transparent background colour. |
| Sef.sc\_img  (SplashScreen) | Tkitner.Label  object | The widget that the logo will be within. |
| Self.progress  (SplashScreen) | Tkinter.ttk.Progressbar  object | The progress bar widget that indicates how much of the program has loaded. |
| Self.setup\_info\_text  (SplashScreen) | Tkinter.StingVar  object | The variable text that will change to the bit of the program that is being loaded. |
| Self.setup\_info  (SplashScreen) | Tknter.Label  object | The label that the above variable sting will be inside. |
| Self.colour | String | The hexadecimal RGB value of the spash screen window background colour. |
| Self.increase  (SplashScreen) | Boolean | This is set to True when the green value of the RGB background colour is meant to increase (making the background change to orange) and is False when the green value is meant to decrease (making the background change to red). |
| Green  (SplashScreen) | String | The new hexadecimal value for the green component of the RGB value for the background. |
| Self.progress[“value”]  (SplashScreen) | Integer | The percentage of the progress bar filled in. |
| Number  (FractionQuestionFrame) | Integer | The specific question number to be displayed in this question frame. |
| Func, self.function  (FractionQuestionFrame) | String | The function that this question is going to have (+ - \* /) This is used to determine the correct answer and display the correct function. |
| Mode/self.mode  (FractionQuestionFrame) | String | Specifies if this question is for a worksheet or a duel. If the question is for a worksheet then when the question is marked the correct answers are shown (because the only time a worksheet question is marked is when the worksheet is complete). However, a duel question is marked every 0.5 seconds during the duel so the correct answers are not shown. |
| Self.marked  (FractionQuestionFrame) | Boolean | Indicates of the question has already been marked and corrected. |
| Self.function\_var  (FractionQuestionFrame) | Tkitner.StringVar  Object | The operator that is to be displayed in the question. (“+ or”-”or”\*”or”/”) |
| Self.x1, self.y1, self.a1,  (FractionQuestionFrame) | Tkinter.StringVar  Object | These values are the numerators of the 1st and 2nd fraction, as well as the numerator of the answer. |
| Self.x2, self.y2, self.a2,  (FractionQuestionFrame) | Tkinter.StringVar  Object | These values are the denominators of the 1st and 2nd fraction, as well as the denominator of the answer. |
| Decimal\_answer  (FractionQuestionFrame) | Float | The decimal answer calculated from the two fractions. |
| Answer  (FractionQuestionFrame) | String | The decimal answer in a fraction format where it is spit into a numerator and denominator. |
| Data/self.data  (FractionduelFrame) | List | The list of numerators and denominators that is to be applied to the set of questions. (supplied by the game server) |
| Self.start\_time  (Worksheet) | Float | The time since epoch used to calculate time taken to complete a worksheet. |
| Frames  (Woksheet) | List | A list of all the frames generated depending on the amount of questions in a worksheet. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Calculations**

**Elo rating calculations:**

A user’s elo rating is a dynamic way to show their rank as you do not get a flat elo gain/loss from winning/losing. Instead the amount of elo you gain or lose is based on the elo of the player you are against. First, the expected outcome of a game for player A against a player B is calculated (1 is a win and 0 is a loss therefore if the calculated value is say 0.71 then they should have a 71% chance of winning.).



Where EA is the chance of player A winning against player B.

RB is the elo rating of player B

RA is the elo rating of player A.

Then the new elo rating of player A is calculated.



Where RA’ is the new elo rating of player A.

RA is the elo rating of player A before the game.

K is the K-factor (value that is usually 16 or 32, I chose 32 as it seemed more interesting for kids to see larger rank increases)

SA is the actual outcome of the game for player A (1 for a win, 0 for a loss)

EA is the expected outcome of the game for player A calculated before.

I will round all values up so that 1 is the minimum elo gain/loss and 32 is the maximum

Examples:

1)

A player with 1500 elo playing against a player with 500 elo should have an expected outcome of 0.99, so if they win the game they will only have an elo increase of 1, if they lose the game they will have an elo decrease of 32.

2)

A player with 1000 elo playing against a player with 1000 elo would have an expected outcome of 0.5, so if they win the game they will have an elo increase of 16, if they lose the game they will have an elo decrease of 16.

3)

A player with 900 elo playing against a player with 1100 elo would have an expected outcome of 0.24, so if they win the game they will have an elo increase of 25, if they lose the game they will have an elo decrease of 8.

**Systematically generate quadratic expressions in a simplified, expanded state where they can be factorised by keyStage3 students:**

For a Keystage3 student to be able to factorise an expanded quadratic the numbers both in the question and in the answer, should be integers and the fraction should be factorizable into two brackets.

For a quadratic expression to have 2 roots and therefore be factorizable into two brackets, the discriminant of the expression should be > 0.

Quadratic expression where a, b and c are all integers:

a(x^2) + bx + c

where a is the coefficient of (x^2). (integer)

where b is the coefficient of x. (integer)

where c is the constant. (integer)

Discriminant calculation and inequality:

Discriminant = b^2 – (4 \* a \* c), Discriminant > 0

For a quadratic expression to be factorised using all integers the product of a and c should have a factor pair that adds up to make b. also the square root of the discriminant must be an integer.

Factorizable using integers calculation:

ac = a \* c, sum of factor pair of ac = b

Examples:

1)

2(x^2) + 10x + 12, discriminant = 10^2 – (4 \* 2 \* 12) = 4, 4 > 0, sqrt(4) = 2 = integer

ac = 2 \* 12 = 24, factor pair = (6 and 4), 6 + 4 = 10 = b, Therefore the expression can be used.

2(x^2) + 10x + 12 = (2x + 6) (2x + 2)

2)

3(x^2) + 5x + 2, discriminant = 5^2 – (4 \* 3 \* 2) = 1, 1 > 0, sqrt(1) = 1 = integer

ac = 3 \* 2 = 6, factor pair = (3 and 2), 3 + 2 = 5 = b, Therefore the expression can be used.

3(x^2) + (x + 2 = (1x + 1) (3x + 2)

3)

2(x^2) +3x + 12, discriminant = 3^2 – (4 \* 2 \* 12) = -39, -39 < 0, Therefore the expression cannot be used as the quadratic has no real roots.

4)

1(x^2) + 9x + 16, discriminant = 9^2 – (4 \* 1 \* 16) = 17, 17 > 0, sqrt(17) = 4.12… = non integer

ac = 1 \* 16 = 16, factor pairs = (1 and 16), 1 + 16 = 17, (2 and 8), 2 + 8 = 10, (4 and 4), 4 +4 = 8, none of which = b, Therefore the expression cannot be used because the answer will not consist of all integers.

**Systematically calculate simplified factorised quadratic expressions:**

Get the factor pairs of the product of a and c:

2(x^2) + 7x + 3

ac is 2 \* 3 = **6** and b is **7**

So, we want two numbers that multiply together to make 6, and add up to 7

In fact, **6** and **1** do that (6 \* 1 = 6, and 6 + 1 = 7)

Then split b into two terms with the coefficients of each term being a member of a factor pair that adds up to b:

2(x^2) + 6x + 1x + 3

Then partially factorise the (x^2) term with one of the split terms and partially factorise the (constant c) with the other split term:

(2(x^2) + 6x) (1x + 3)

Then bring out a factor of each partially factorised term leaving a common factor that will be one of the final brackets and add these factors that have been taken out together to be the other final bracket:

2x(x + 3) + 1(x + 3)

(2x + 1) (x + 3)

**Systematically calculate simplified expanded quadratic expressions:**

To get an expression from the form:

(Px + Q) (Rx + S)

to the form:

a(x^2) + bx + c

a = P \* R

b = (P \* S) + (R \* Q)

c = Q \* S

The HCF (highest common factor) of a, b and c is then calculated (see algorithm in 2.5) then a, b and c are divided by this HCF to get the final simplified answer.

HCF = highest common factor of (a, b, c)

a / HCF

b / HCF

c / HCF

Examples:

(3x + 4) (2x + 6)

A = 3 \* 2 = 6

B = (3 \* 6) + (2 \* 4) = 26

C = 4 \* 6 = 24

HCF = 2

a / 2 = 6 / 2 = 3

b / 2 = 26 / 2 = 13

c / 2 = 24 / 2 = 12

Therefore (3x + 4) (2x + 6) = 3(x^2) + 13x + 12

**Systematically generate values for fractions questions suitable for KeyStage3 students and Systematically generate quadratic expressions in a simplified, factorised state where they can be expanded by keyStage3 students:**

All this is done using random integers that do not need to correlate with each other, therefore no calculations are necessary.

**systematically calculate simplified fractions using (+ - \* /) operators:**

This is calculated mostly using an eternal module.

Pseudo code explaining how all calculations are executed is in 2.5.

2.3 Validation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Where | Validation rule | Error Message and actions | When to check for error |
| Username | Register window | Username must not exist in the database, have 3-16 characters. | Invalid entry: Username | When the register button is pressed/ when return is pressed. |
| First name | Register window | Only letters, have 1-16 characters. | Invalid entry:  First name | When the register button is pressed/ when return is pressed. |
| Second name | Register window | Only letters, have 1-16 characters. | Invalid entry:  Second name | When the register button is pressed/ when return is pressed. |
| Email | Register window | Be in the form example@email.com | Invalid entry:  Email | When the register button is pressed/ when return is pressed. |
| Password | Register window | At least 8 characters, at least one number, at least one special character. | Invalid entry:  Password | When the register button is pressed/ when return is pressed. |
| Re-enter password | Register window | Must be the same as the password field. | Invalid entry:  Re-enter password | When the register button is pressed/ when return is pressed. |
| School code | Register window | Must be a valid school code within the school table. | Invalid entry:  School code | When the register button is pressed/ when return is pressed. |
| Post code | Register window | Must be a valid UK postcode. (checked against a regularly updated database that the API that is used, uses.) | Invalid entry:  Post code | When the register button is pressed/ when return is pressed. |
| Number of questions | Sheet options window | Presence check, must be between 1-99, must be an integer | Please make sure all fields are filled in and correct | When the generate button is pressed. |
| Functions | Sheet options window | Presence check | Please make sure all fields are filled in and correct. | When the generate button is pressed. |
| Profile | Profile search window | Username must exist in the database. | Username does not exist | When the search button is pressed/ when return is pressed. |
| Username | Login window | Username must exist in the database. | Invalid Username | When the login button is pressed/ when return is pressed. |
| password | Login window | Password must match the password in the record that has the same username entered. | Invalid Password | When the login button is pressed/ when return is pressed. |
| Profile picture | Profile window | Presence check, file must be an image file. | “File not selected” or “invalid file type” |  |

My program only ever validates fields when a whole form is complete because my client found it rather annoying to have error messages appear when they lost focus on a field. Generally, if the data a user has input into a field does not comply with the validation rules, an error message is displayed with the incorrect field pointed out to them before the data is manipulated by the program (as that may cause it not to function properly or crash) or before the data is saved to the database.

Pseudo code for general validation:

Input data

If validation == True:

Manipulate/ save data.

Else:

Display error message.

Pseudo code for validating length:

Input data, min\_length, max\_length

If min\_length < len( data) < max\_length:

Manipulate/ save data.

else:

Display error message

Endif

Pseudo code for validating number\_of\_questions is an integer:

Input number\_of\_questions

Try:

Turn number\_of\_questions into integer data type

Generate worksheet with the number\_of\_questions

Except error returned when number\_of \_questions is not an integer data type (ValueError):

Display error message

Pseudo code for if username exist in the database to sign up:

Input username

Query the database for records with username in them

If query is empty:

Save username to database

Else:

Display error message

Pseudo code for validating if a profile picture selected:

Input profile picture chosen

Try:

Resize image to 128x128

Upload to profile picture section of website

Except error returned when no file is selected(AttributeError):

Display error message: “File not selected”

Except error returned when a file that is not an image file is selected(OSError):

Display error message: “invalid file type”

Regular expression to validate email address before registering:

([a-z]|[A-Z]|[0-9])+@([a-z]|[0-9])+\.([a-z]|.)+

([a-z]|[A-Z]|[0-9])+ ensures that the email starts out with one or more alphanumeric characters.

@ ensures there is an @ afterwards.

([a-z]|[0-9])+ ensures the first part of the domain is alphanumeric and lowercase.

\. Ensures there is a “.” Afterwards.

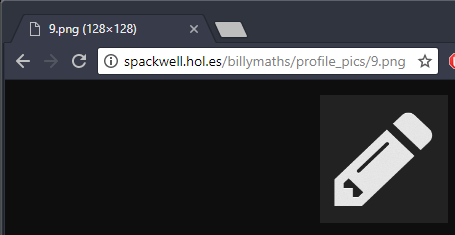
([a-z]|.)+ ensures the last part of the domain consists of one or more letters and “.”s.

2.4 Files

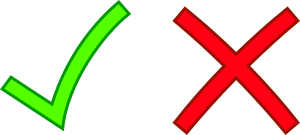
image files-

All profile pictures are uploaded (using FTP) and read from (using a url request) my website at: <http://spackwell.hol.es/billymaths/profile_pics/>

each directory is a user’s profile picture corresponding to their user id. Each image is resized to 128x128 before uploading.



tick.png and cross.png used for marking questions.



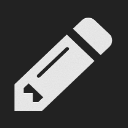
A.png, B.png, C.png, D.png used for grading worksheets.



Logo used for splash screen (sc2.png)



Logo used for icon and profile picture placeholder that is uploaded when a new user is registered. (favicon2.ico, profile\_placeholder2.png)



Edit symbol that appears over profile picture when cursor is over it (edit.png)



scripts-

Client.py- The main program that a pupil would use.

db\_server.py- Handles data requests and commits from the client. (to be hosted elsewhere)

game\_server.py- Handles clients searching for a duel and pairs them up, then runs the game using multi-threading.

Others-

database file (SQLite3)- holds user information, school information, worksheet and duels results.

config.txt- User can change variables in the config file to enable/disable the splash screen and change the colour theme of the program. Each line of the file has a setting assigned to a value, for instance: (enable\_splashscreen) = (1/0 and colour\_theme = “black”/”yellow”/”red”/”#26f23e”/”#ffffff”…).

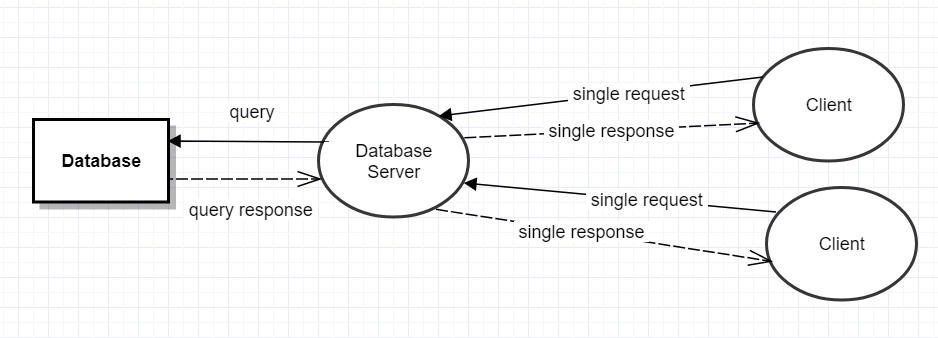
Static\_funcs.py- A library that holds all the functions of my program that only take parameters and do not need to be within classes.

Validation.py- A library that holds all the functions necessary to validating entry fields.

2.5 Algorithms

**Server-Client Related Algorithms:**

Database Server-Client structure:



Each time a user needs to contact the database for saving or receiving data a temporary connection to the database server is established, the server then handles the specified query(s) with the parameters sent by the client and sends a response with the necessary data.

**Pseudo code for algorithms involving the data server:**

**Handling connection and queries:**

Client-

Procedure db\_connection\_handler

Input procedure for database server to perform

Input parameters for that function

Try:

Connect to database server socket

Except:

Display error saying the database server cannot be reached.

Return “error”

Endproc

Send server password

Receive response

If response reived says password is correct:

Send procedure and parameters to database server

Receive response with data returned by database server

Close connection to server socket

Return received data

Endproc

Else:

Close connection to server socket

Display error saying the database server cannot be reached.

Return “error”

Endproc

Server-

Procedure client\_connection\_handler

Loop:

Wait for new connection

Tell client a connection has been made successfully

Receive client password attempt

If password is correct:

tell client password is correct

receive procedure and parameters to execute

execute desired procedure with desired parameters

tell client the returned values from the function run

else:

tell client password is incorrect

**Pseudo code for registering a new user:**

Client-

Input username, first name, second name, email, password, re-enter password, school code, postcode

Validate fields that do not require the database.

If fields are valid:

Send all inputs to the database server through the database connection handler to run the… …register procedure

If the database connection handler says the fields validated against the database are valid:

Display message saying the registration process is complete

Endproc

Else:

Display error message specifying the incorrect fields

endproc

Else:

Display error message specifying the incorrect fields

endproc

Server-

Connection handler receives message from client to run the register procedure

Check If username exists in the database already

Check if the school code entered is in the list of correct school codes

If username and school code are valid:

Encrypt password

Save all necessary fields to the database under a new record in the user table

endproc

Else:

Return incorrect fields

Endproc

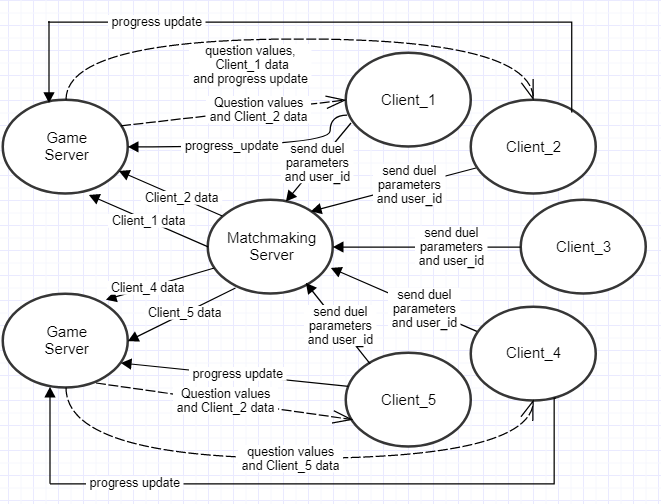
Summary of above pseudo code:

The user inputs their

Game Server-Client structure example:

Clients connect to the matchmaking server and if their search parameters are the same as another client’s search parameters whom is currently searching for a game, then the two clients are put into a game where a set of questions are generated based on their search parameters. These questions as well as the opposing client’s user\_id are then sent back to each client. The duel is then started and each time a user gets another question correct, the amount of questions they have correct is sent to the game server where it is then sent to their opponent’s client for both clients to be constantly up to date with each other’s progress. When a user wins the duel, the opposing client is told, and the game server is closed. The winning client then updates the database with the results of the duel and each client updates the database with their changed elo rating.

The matchmaking aspect and game aspects of the game server are modelled as separate servers here; however, they are not separate files. This is because for every “Game server” there is, a new thread is created so that the matchmaking can continue whilst there are multiple simultaneous duels ongoing.



In the example above client\_1 would have joined the matchmaking queue and not found a match because there is no one else searching for a game, then client\_2 would have connected with the same search parameters as client\_1 so they are matched together, and their connections would be routed to a new “Game Server” that is created. The “Game Server” would then generate a duel based on the search parameters of the clients and will continuously receive and send progress updates to client\_1 and client\_2. Client\_3 would then connect and not find a match as there is no one else searching for a game. Client\_4 would then connect and not find a match because it’s search parameters do not match any search parameters of the currently connected clients in the matchmaking queue (Client\_3). Then Client\_5 would have joined and been matched with Client\_4 because they have identical search parameters. A new “Game Server” will then be created and the process described before will occur. The two “Game Servers” will close once they have winners and Client\_3 will remain in the matchmaking queue until a client with matching search parameters joins or the user of client\_3 decides to stop searching for a game.

Pseudo code for the matchmaking:

This matches clients with identical search parameters and starts a new game with them both. It also removes any dead connections where clients have closed the duel search before finding a match.

Game Server-

Searching Connections = empty list

Loop:

Wait for new connection

Tell client a connection has been made successfully

Receive client password attempt

If password is correct:

tell client password is correct

receive client’s search parameters

dead connections = empty list

for each searching connection in the searching connections list:

if searching connection is alive:

if search parameters of searching connection match the parameters of the new connection:

add searching connection to dead connections list

start new game thread with the new connection and… …searching connection

end for loop

else:

add searching connection to dead connections list

for dead connection in the dead connections list:

remove dead connection from the searching connections list

if a game was not found:

add new connection to searching connections list

else:

tell client password is incorrect

Pseudo code for a game:

This generates a set of identical questions for each client in a duel and updates each clients’ progress as the duel goes on.

Procedure game:

Input number of questions, topic, types, connection and user\_id of client2, connection and user\_id of client 2

Generate duel questions based on the specified topic, type and number of questions

Send the user\_id of client 2 and generated questions to client 1

Send the user\_id of client 1 and generated questions to client 2

Loop:

Receive progress of client 1

Send received progress of client 1 to client 2

Receive progress of client 2

Send received progress of client 2 to client 1

If message received form either client says that the game has ended:

Endproc

Calculation algorithms:

Generate quadratic expressions in a simplified, expanded state where they can be factorised by keyStage3 students:

Procedure set values for factorising:

Loop:

Set a to a random integer between 1-4

Set b to a random integer between 1-10

Set c to a random integer between 1-20

Try:

If the square root of the discriminant of a, b and c is an integer:

Ac = a \* c

Get the factor pairs of ac

For each pair in the set of factor pairs of ac:

If the sum of the pairs == b:

Valid pair = pair

Endproc

Except:

Do nothing

Calculate simplified factorised quadratic expressions:

Procedure calculate factorised quadratic:

Input a, c and valid pair

Factor1 = highest common factor of (a and valid pair[1])

Divide (a + valid pair[1]) through by factor1

A = a / factor1

Valid pair[1] = valid pair[1] / factor1

Factor2 = highest common factor of (c and valid pair[2])

Set answers to (factor1x +factor2) (ax + valid pair[1])

Calculate simplified expanded quadratic expressions: (q1x + q2) (q3x + q4)

Input q1, q2, q3, q4

A = q1 \* q3

B = (q1 \* q4) + (q2 \* q3)

C = q2 \* q4

Factor = highest common factor of (a, b and c)

A = a / factor

B = b / factor

C = c / factor

Elo rating change:

Input opponent’s user\_id, my elo,

Opponent’s elo = Query database for opponent’s elo using their user\_id

Win chance = 1 / (1 + (10 ^ ((opponent’s elo – my elo) / 400)))

If this client won:

New elo = my elo + (32 \* 1 – win chance)

Else:

New elo = my elo + (32 \* -win chance)

2.6 Data Structures

**Lists/arrays:**

All of the data sent between the servers and the client are in a list format so that when they are converted from a bytes data type to a string data type they can then be converted using ast.literal\_eval() to a list data type.

Args (db(), client): The list that holds all the data that is to be sent off to the database server.

Msgr (connection\_handler(), database server): holds the function which is to be run, as well as the parameters to use.

Msgr (matchmaking(), game server): Holds all the information regarding the duel search parameters of the client.

questions (set\_frac\_numbers(), game server): A list of numerators and denominators that is sent to both clients in a duel so that they can generate duels with identical questions.

Self.questions (FractionsWorksheet, client): A list holding all the FractionQuestionFrame widgets to be gridded.

Error\_list (db\_register(), database server), (Register, client): List containing all the invalid fields a user entered when a user tries to register

Self.user\_data (MathsGui, client), (Profile, client): list of all the information about a user in their user table record apart from their password.

**Classes:**

**(Class diagrams included in analysis)**

**Client:**

**MathsGui**

|  |  |
| --- | --- |
| Purpose | The menu frame that is loaded on the root window. This is where the user navigates to all other aspects of the program. |
| Methods | **Setup()-** Calls all the create\_widgets() functions and updates the splash screen progress if the “enable\_splashcreen = 1” is in the config.txt file.  **Load\_config()-** checks if there are valid options in the config.txt file, if there are then the appropriate attributes are changed, otherwise they remain as their defaults.  **Create\_login\_widgets(), create\_main\_menu\_widgets(), create\_worksheet\_widgets(), create\_profile\_widgets(), create\_duel\_widgets()-** Creates all the widgets for the different menus.  **Gotofrom(dest\_frame, curr\_frame)-** ungird the current frame and grid the desired frame, also change the window size if required.  **Login(user, passw)-** Checks if the fields in the login\_frame are correct, if not display an error message with the incorrect fields, if they are correct, set self.user\_data to the user data of the logged in user and change the login frame to the main\_menu frame. |
| Attributes | User\_data, master, validation, main\_menu\_frame, worksheets\_frame, duels\_frame, profile\_frame, login\_frame, config |

**Profile**

|  |  |
| --- | --- |
| Purpose | A window that displays a user’s duel/worksheet history, the user’s profile picture, general information about a user including their name, elo and school. If the user is viewing their own profile, then they can also change their profile picture. |
| Methods | **Upload\_profile\_pic()-** Called when a user clicks on their profile picture. A file selector window appears where the user choses an image file that they wish to be their profile picture, a temporary image file that is a resized 128x128 version of their selected image is made. That image is uploaded to the profile picture section of the website and then deleted.  **Set\_tree\_worksheets(), Set\_tree\_duels()-** creates the tree view widgets and sets their columns.  **Change\_table\_from\_to(table1, table2)-** changes the currently displayed table to the one the user choses by pressing the duel/worksheet history buttons.  **Set\_data(tree, orderby, mode)-** Sets the data inside the specified tree to be ordered by the specified column. |
| Attributes | User\_data, woksheet\_data, duel\_data, profile\_pic\_img and other GUI widgets. |

**ProfileSearch**

|  |  |
| --- | --- |
| Purpose | Window where the user can search for other users’ profiles by typing in a user’s username. |
| Methods | **Search()-**  If the username entered exists then it calls the Profile class with the searched for user’s user\_id. If the user does not exist, then an error message is displayed. |
| Attributes | GUI widgets. |

**Register**

|  |  |
| --- | --- |
| Purpose | Window where a new user can create an account. |
| Methods | **Register()-** Validates all fields entered that can be validated locally, if they are not valid the invalid fields are pointed out, otherwise the fields are sent of to the database server for further validation. If the fields are found to be valid by the database server then a new record in the user table is inserted with the details of the new user, the register window displays a message saying: “Registration complete” and the register window is destroyed. If the further validation finds an error in the fields then the invalid fields are pointed out in the register window. |
| Attributes | Username\_entry, firstname\_entry, secondname\_entry, email-entry, password\_entry, re\_enter\_password\_entry, school\_code\_entry, post\_code\_entry, other GUI widgets. |

**GameOver**

|  |  |
| --- | --- |
| Purpose | Window that appears at the end of a duel with the duel results. |
| Methods | **None** |
| Attributes | Win\_chance, opponent\_info, time, uid, opponent, elo, win\_state, topic, qn, funcs, GUI widgets. |

**DuelSearch**

|  |  |
| --- | --- |
| Purpose | Window that displays the current time taken to find a duel, the user’s profile picture and current elo. Also handles the connection to the game server whilst in matchmaking. |
| Methods | **Update\_time()-** Updates the time\_taken label widget every second.  **Safe\_connection\_shutdown(\*\*kwargs)-** This method is called whenever the duel search window or actual duel window are closed/crashed while connected to the game server or when a duel has ended. It safely closes the connection to the game server to prevent both the game server and the client from crashing. An error message is also displayed necessary.  **Status\_update(duel, opponent)-** continuously sends the status and amount of questions correct that the current client has to the game server and also receives the status and amount of questions correct of the opponent’s client from the game server. It also sets the state attribute of FractionsDuel to “finished” someone has found to have won the duel.  **Connect()-** handles the connection to the game server whilst in matchmaking. |
| Attributes | Uid, topic, qn, funcs, elo, time, s, GUI widgets |

**FractionsDuel**

|  |  |
| --- | --- |
| Purpose | Window where the user carries out questions in the quickest time possible to beat another user that is completing the same set of questions. |
| Methods | **Set\_state()-** Used to change the state to “finished” when either the user wins the duel or their opponent wins the duel. The time taken to complete the duel is also calculated.  **Check\_answers()-** Every 0.5 seconds mark all the questions in the duel to see if any more answers have been answered correctly. If the “state” attribute of the class == ”finished” then the remaining incorrect questions have the correct answers shown next to them. |
| Attributes | Question\_no, op1, op2, op3, op4, game\_data, correct, time, state, GUI widgets. |

**WorkSheet**

|  |  |
| --- | --- |
| Purpose | The base class that other topic variants can inherit from. Includes the base attributes and methods of for a worksheet. |
| Methods | **Mark(user\_id, questions, window, menu\_items, topic, type)-** Marks all the questions in the worksheet, displays a window with the results and saves the results to the work\_results table in the database.  **Change\_page(destpage, frames)-** ungirds the currently displayed frame and girds either the next or previous page based o the button they pressed. |
| Attributes | Start\_time, finish\_time, correct, GUI widgets. |

**SheetOptions**

|  |  |
| --- | --- |
| Purpose | The base class that other topic variants can inherit from. Includes the base attributes for a sheet options window. |
| Methods | **None** |
| Attributes | Question\_no, question\_no\_label, question\_no\_entry, question\_nomenu |

**FractionsSheetOptions**

|  |  |
| --- | --- |
| Purpose | Window that is used to select the parameters needed to generate a fractions worksheet/ search for a fractions duel |
| Methods | **Generate()-** calls either the FractionsWorksheet class or the DuelSearch class with all the user selected parameters. |
| Attributes | Mode, user\_id, add\_var, sub\_var, mult\_var, div\_var, GUI widgets. |

**QuadraticsSheetOptions**

|  |  |
| --- | --- |
| Purpose | Window that is used to select the parameters needed to generate a quadratics worksheet/ search for a quadratics duel |
| Methods | **Generate()-** calls either the FractionsWorksheet class or the DuelSearch class with all the user selected parameters. |
| Attributes | Mode, user\_id, fact\_var, exp\_var, GUI widgets. |

**FractionQuestionFrame**

|  |  |
| --- | --- |
| Purpose | Each of these frames contains a single fractions question that can have 4 different types of operator. One or more of these frames are gridded to make a worksheet. |
| Methods | **Set\_values()-** Set the values of each numerator and denominator  **Get\_correct()-** set the values of the correct numerator and denominator  **Mark()-** If the entered numerator and denominator match the correct numerator and denominator then the question is considered marked, as well as displaying an image of a tick next to the question. For a duel if the answer is incorrect nothing happens, for a worksheet show\_answers() is run.  **Show\_answers()-** displays the correct answer as well as an image of a cross next to the question. |
| Attributes | Mode, marked, x1, y1, x2, y2, a1, a2, function\_var, … GUI widgets. |

**QuadraticsQuestionFrameFactorise**

|  |  |
| --- | --- |
| Purpose | Each of these frames contains a single quadratics question to do with factorising. One or more of these frames are gridded to make a worksheet. |
| Methods | **Set\_values()-** Set the values of each numerator and denominator  **Get\_correct()-** set the values of the correct numerator and denominator  **Mark()-** If the entered numerator and denominator match the correct numerator and denominator then the question is considered marked, as well as displaying an image of a tick next to the question. For a duel if the answer is incorrect nothing happens, for a worksheet show\_answers() is run.  **Show\_answers()-** displays the correct answer as well as an image of a cross next to the question. |
| Attributes | Mode, marked, a, b, c, a1, a2, a3, a4, … GUI widgets. |

**QuadraticsQuestionFrameExpand**

|  |  |
| --- | --- |
| Purpose | Each of these frames contains a single quadratics question to do with expanding. One or more of these frames are gridded to make a worksheet. |
| Methods | **Set\_values()-** Set the values of each numerator and denominator  **Get\_correct()-** set the values of the correct numerator and denominator  **Mark()-** If the entered numerator and denominator match the correct numerator and denominator then the question is considered marked, as well as displaying an image of a tick next to the question. For a duel if the answer is incorrect nothing happens, for a worksheet show\_answers() is run.  **Show\_answers()-** displays the correct answer as well as an image of a cross next to the question. |
| Attributes | Mode, marked, q1, q2, q3, q4, a, b, c, … GUI widgets. |

**FractionsDuelFrame**

|  |  |
| --- | --- |
| Purpose | Each of these frames contains a single fractions question that can have 4 different types of operator. One or more of these frames are gridded to make a duel. |
| Methods | **Set\_values()-** Set the numerators and denominators to the ones generated by the game server. |
| Attributes | Number, func, data, x1, x2, y1, y2, a1, a2… GUI widgets. |

**QuadraticsDuelFrameFactorise**

|  |  |
| --- | --- |
| Purpose | Each of these frames contains a single quadratics question to do with factorising. One or more of these frames are gridded to make a duel. |
| Methods | **Set\_values()-** Set the numerators and denominators to the ones generated by the game server. |
| Attributes | Number, data, a, b, c, a1, a2, a3, a4, … GUI widgets. |

**QuadraticsDuelFrameExpand**

|  |  |
| --- | --- |
| Purpose | Each of these frames contains a single quadratics question to do with expanding. One or more of these frames are gridded to make a duel. |
| Methods | **Set\_values()-** Set the numerators and denominators to the ones generated by the game server. |
| Attributes | Number, data, q1, q2, q3, q4, a, b, c, … GUI widgets. |

**FractionsWorksheet**

|  |  |
| --- | --- |
| Purpose | Window that has a specified number of fractions questions where the user can fill in the worksheet. When the user is done they mark the worksheet and have the questions corrected where needed. A window with their results is then shown. |
| Methods | **None** |
| Attributes | Question\_no, master, op1, op2, op3, op4, frames, … GUI widgets. |

**QuadraticsWorksheet**

|  |  |
| --- | --- |
| Purpose | Window that has a specified number of quadratics questions where the user can fill in the worksheet. When the user is done they mark the worksheet and have the questions corrected where needed. A window with their results is then shown. |
| Methods | **None** |
| Attributes | Question\_no, master, op1, op2, frames, … GUI widgets. |

**SplashScreen**

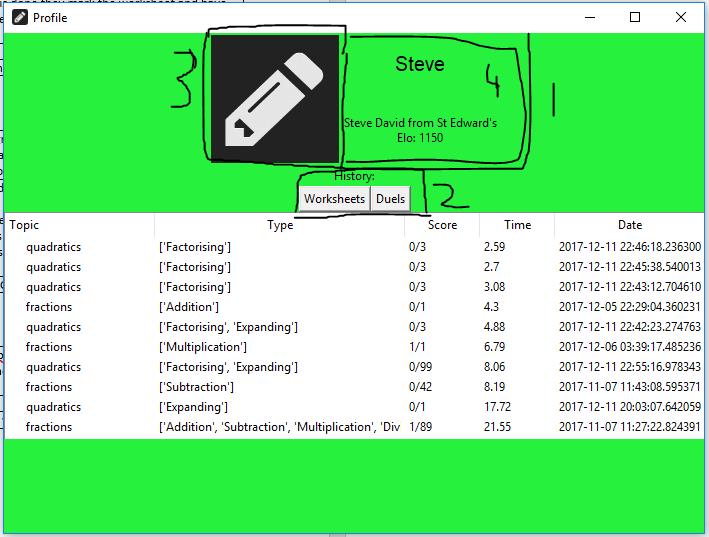
|  |  |
| --- | --- |
| Purpose | Window that appears while the program is loading (only if enable\_splashscreen = 1 is in the config file). A progress bar with what is currently being loaded is shown, as well as the BillyMaths logo and colour pulsing background. |
| Methods | **Colour\_pulse-** pulses the background colour from red to orange to red continuously.  **Set\_progress-** change the percentage of the filled in section of the progress bar and the text overlapping it to what is currently being loaded.  **Set\_progress\_done-** set the progress bar to full and change the text overlapping it to “done” and deiconify the splash screen. |
| Attributes | Setup\_info\_text, colour, increase, … GUI widgets. |

**Leaderboard**

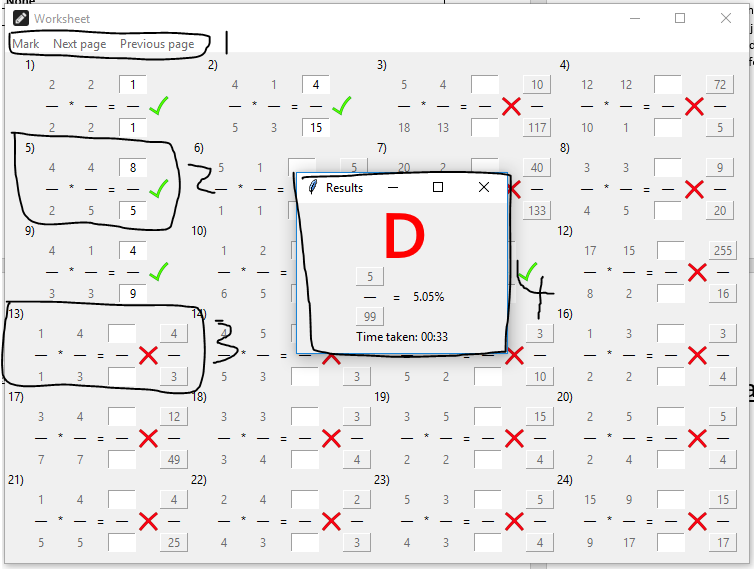
|  |  |
| --- | --- |
| Purpose | Window that appears when leaderboard is pressed on the duel submenu. It shows the elo rating, username, games won and win % of every user in order of descending elo. |
| Methods | **None** |
| Attributes | Leaderboard\_data, tree … other GUI widgets. |

2.7 User Interface

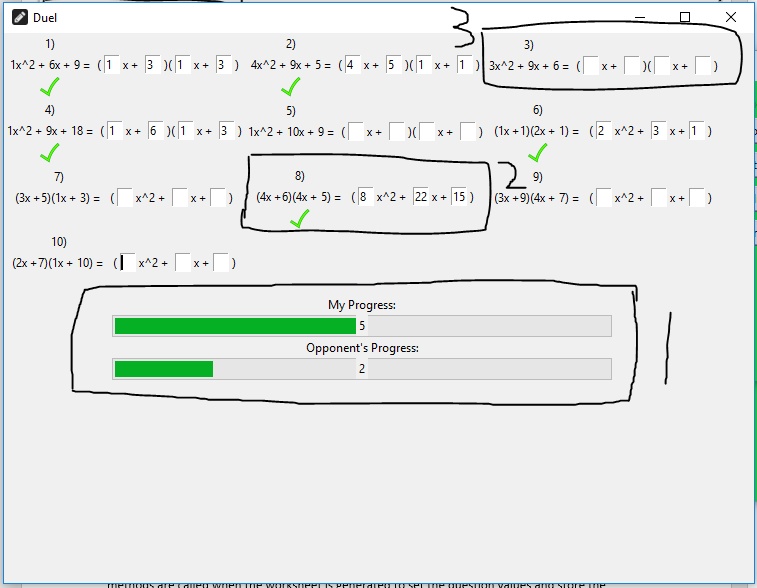
Complex UIs:

Profile page:

To get the structure of this page, there are 4 separate frames and a tree view. Frame 1 consists of the profile picture frame(3) and the text frame(4), these needed to be separate in order to have the edit symbol appear over the profile picture when the cursor is over it because I bound the ENTER event to the profile picture frame, not the label. The profile picture frame has a row span of 2 to allow the username and user info to grid neatly beside it. Frame(2) consists of two buttons and a label that are gridded just below frame(1) but in the same column. The tree view that contains the user’s worksheet and duel history has a column span of 3 and both frame2 and frame1 are gridded in the middle column for them to appear in the middle of the window.

Fractions worksheet after being marked:

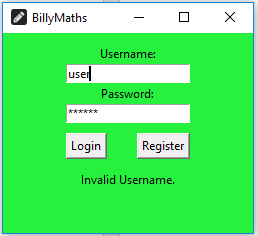
Each worksheet consists of a maximum of 5 pages as the maximum number of questions per page is 24 and the maximum number of questions you can have is 99. Each page consists of 6 rows and 4 columns. Section 1 contains the buttons for changing the pages and marking the worksheet when the user thinks they are done. Every question is an individual FractionsQuestion class with set\_values(), get\_correct, mark() and show\_answers() methods. The set\_values() and get\_correct() methods are called when the worksheet is generated to set the question values and store the correct answers as attributes, the mark() and show\_answers() methods are called when mark is pressed. This checks the user’s entries against the correct answers and if they are the same, a tick is displayed, if not a cross is displayed, and the correct answer is shown (section 2 and 3 are examples of these).The window in section 4 also appears after the worksheet has been marked and displays the user’s score as a fraction and percentage, as well as grading the user A-D based on their % score.

Ongoing quadratics duel:

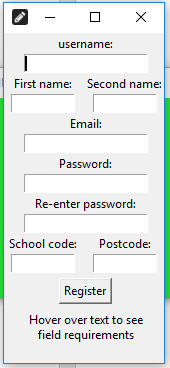
This may look very similar to a quadratics worksheet, that is because a duel inherits from a worksheet, however it handles the question frames in a very different manor. As you can see section 2 includes a question frame that appears marked, however on the same window, the question frame in section 3 appears not marked. This is because every question is being marked every 0.5 seconds, but only the correct answers get indicated with an image. The only time the questions will be marked wrong is when the duel is over, and the losing client will have its remaining answers corrected. Section 1 includes a progress bar for the current client and a progress bar for the opponent’s client. This should be updated every 0.5 seconds (+ any server lag which is most probably negligible) by the connection to the game server. The quadratics duels and worksheets also have 24 questions maximum per page, 3 questions per row and 8 questions per column.

Other UIs:

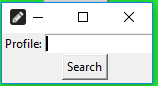
Login:



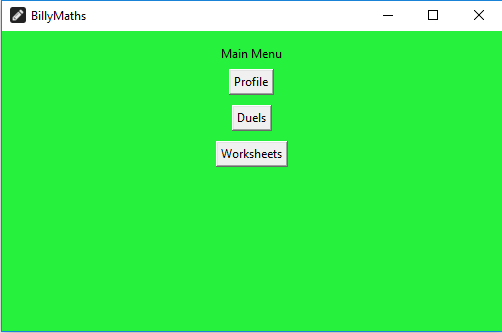
Register:



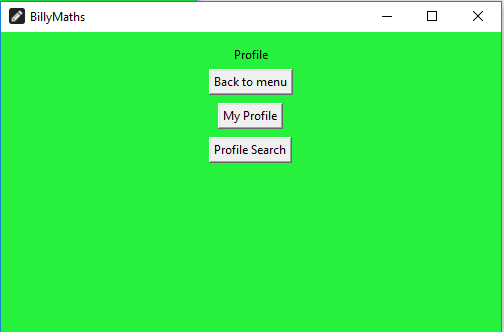
Profile Search:



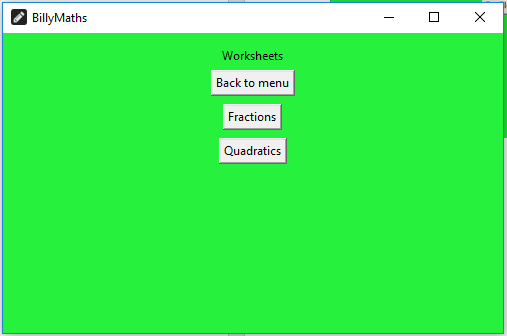
Main menu:



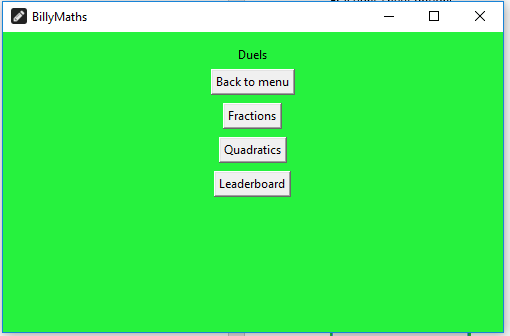
Profile submenu



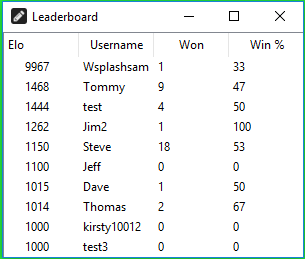
Worksheet submenu:



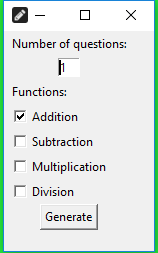
Duel submenu:



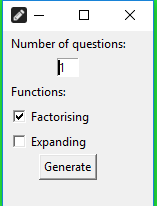
Leaderboard:



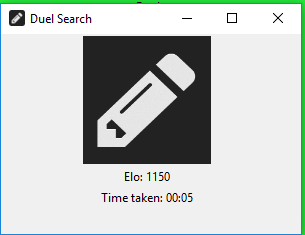
Fractions sheet options:



Quadratics sheet options:



Duel search:



3.Technical solution

4.0 Testing

4.1 Test Plan

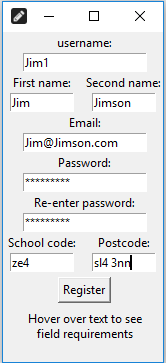
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test No** | **Objective tested** | **Test Data and description** | **Test Type** | **Expected Results** | **Actual Results** | **Comments** |
| 1 | 1- Store data on each individual user. | Register with valid data (Jim1, Jim, Jimson, [Jim@jimson.com](mailto:Jim@jimson.com), Password1, Password1, ze4, sl4 5en) | **Normal data** | Data accepted. | As expected. |  |
| 2 | 1- Store data on each individual user. | Register with valid data (Jim1, Jim, Jimson, [Jim@jimson.com](mailto:Jim@jimson.com), Password1, Password1, ze4, sl4 5en) | **Functional** | All the data entered will be added to the database along with a unique user\_id, encrypted password and 1000 elo. | As expected. |  |
| 3 | 1- Store data on each individual user. | Register with invalid data (Test3, test, test, testattesting.com, password, password, ze4, f2k, fl4) | Normal data | Data not accepted and incorrect data fields displayed (Email, Password, Postcode). No data added to the database. | As expected. |  |
| 4 | 1- Store data on each individual user. | Register with invalid data(Averyverylongname, test, Null, [test@testing.com](mailto:test@testing.com), Password1, pass, ze4, sl4 5en) | normal data | Data not accepted and incorrect data fields displayed (Username, re-enter password, second name). No data added to the database. | As expected. |  |
| 5 |  | Try logging in with no connection to the database (no internet connection) | robustness | An error message should appear saying: “Cannot connect to database, check connection and try again.”. Then nothing else should happen. | As expected. |  |
| 6 |  | Try loading profile with no connection to the database. (press the my profile button in the profile submenu when the database server is not running) | robustness | An error message should appear saying: “Cannot connect to database, check connection and try again.”. Then nothing else should happen. | As expected. |  |
| 7 |  | Generate a fractions worksheet (no fields filled in) | validation | Incomplete fields pointed out. | As expected. |  |
| 8 |  | Generate a fractions worksheet (99 questions, with addition, subtraction, multiplication and division checked) | Functional/on boundary | A worksheet with 99 questions, an even distribution of functions(25+, 25-, 25\*, 24/) | As expected. |  |
| 9 |  | Complete a fractions worksheet with 5 multiplication questions and 0 answers correct. | functional | All the questions should be corrected. A window with the user’s score and time taken is displayed. Result should be stored in the worksheet results table. | As expected. |  |
| 10 |  | Test average speed of loading a leaderboard with 100, 1000, 10000, 100000, 1000000 users in the database. The limit of time is based on the time it takes to quicksort the data because the time it takes to load from the database is negligible. | Volume test | Should sort fast enough for small amounts of data but would become impractical for larger amounts of data. | average time in seconds:  0.00081, 0.0434, 3.52, 388, 1million pieces of data took too long to wait for to complete the test (>10minutes) | The times for a low ammount of data is resonable, i doubt that this system would be used by >10000 users and if it was, then regeonal leaderboards could be made based on the users' postcode. |
| 11 |  | Load my profile (by pressing my profile button in the profile submenu) | functional | Correct profile picture, information and history displayed . | As expected. |  |
| 12 |  | Search for another user’s profile with a username that doesn’t exist (qwertyuiop) | validation | Error to be displayed and no profile loaded. | As expected. |  |
| 13 |  | Search for another user’s profile with a username that does exist (Steve) | functional | Search window is destroyed and a profile window with correct profile picture, information and history displayed for the specific user. | As expected. |  |
| 14 |  | Login with incorrect username. (qwertyuiop) | Validation | Error saying that an incorrect username was input. | As expected. |  |
| 15 |  | Login with correct username and incorrect password. (qwertyuiop, 123) | Validation | Error saying that an incorrect password was input. | As expected. |  |
| 16 |  | Login with correct username and password. (Steve, Test) | functional | Login window is destroyed, menu window is displayed. | As expected. |  |
| 17 |  | Complete a fractions worksheet with 5 multiplication questions and 4 answers correct. | functional | All the incorrect questions should be corrected. A window with the user’s score and time taken is displayed. Result should be stored in the worksheet results table. | As expected. |  |
| 18 |  | Generate a quadratics worksheet with a letter in the number of questions entry (t9) | Validation | Error displayed, and no worksheet generated | As expected. |  |
| 19 |  | Generate a quadratics worksheet with no checkboxes filled in and 5 questions. | Validation | Error displayed, and no worksheet generated | As expected. |  |
| 20 |  | Generate a fractions worksheet with nothing in the number of questions box and all checkboxes filled in | Validation | Error displayed, and no worksheet generated | As expected. |  |
| 21 |  | Press the next page button on a fractions worksheet with 10 questions. (multiplication) | robustness | Nothing should happen | As expected. |  |
| 22 |  | Press the next page button on a fractions worksheet with 50 questions (subtraction) | Functional | The current page of questions is removed, and the next page of questions is displayed. | As expected. |  |
| 23 |  | Press the next page button at the last page of a worksheet (fractions, 50 questions, multiplication) | Functional | The current questions are removed, and the first page of the worksheet is displayed. | As expected. |  |
| 24 |  | Press the previous page button with 5 questions on a quadratics worksheet. (Expanding) | Functional | Nothing should happen | As expected. |  |
| 25 |  | Press the previous page button after pressing the next page button. (quadratics worksheet, 99 questions, expanding.) | Functional | The current page of questions is removed, and the previous page of questions is displayed. | As expected. |  |
| 26 |  | Press the previous page button at the first page of a worksheet (quadratics worksheet, 99 questions, expanding.) | Functional | The current questions are removed, and the last page of the worksheet is displayed. | As expected. |  |
| 27 |  | Complete a worksheet with at least one field having an invalid character. (quadratics worksheet, 99 questions, expanding.), invalid character (&) | Validation | The question should be marked as wrong and the window should not crash. A window with the user’s score and time taken is displayed. Result should be stored in the worksheet results table. | As expected. |  |
| 28 |  | Press each tab of the history tree on a profile page with 5 rows of history. | Functional | After each tab is pressed for the first time the table data should be ordered in descending order of the corresponding data below the tab pressed. | As expected. |  |
| 29 |  | Press each tab of the history tree on a profile page with 20 rows of history twice. | Functional | After each second press the table data should be ordered in descending order of the corresponding data below the tab pressed. | As expected. |  |
| 30 |  | Search for a fractions duel with functions: +-\*/ with one client | Functional | A search window with a counting time elapsed label appears (no game should be found). | As expected. |  |
| 31 |  | Search for a fractions duel with functions: +-\*/ with two clients | Functional | A search window with a counting time elapsed label appears for the first client and when the second client searches the two should be put into a game. | As expected. |  |
| 32 |  | Generate a worksheet with 100 questions (fractions, addition) | Just out of boundary | Only 2 characters should be accepted in the entry with any attempt to type more doing nothing. (trying to type 100 into the entry should result in 10 being in the entry box.) | As expected. |  |
| 33 |  | Generate a worksheet with -1 questions (fractions, addition) | Just out of boundary | The invalid question number entry should be pointed out no worksheet should be generated. | As expected. |  |
| 34 |  | copy and paste 100 into the question number entry box. | Just out of boundary/ validation | 10 should be input into the box because no more than 2 characters are allowed in the box. | As expected. |  |
| 35 |  | Search for a fractions duel with parameters (10 addition and subtraction). Leave the search open for 146seconds. | functional | time taken should increase every second and the time format should be correct (minutes:seconds). The user’s elo and profile picture is correctly displayed. | As expected. |  |
| 36 |  | search for different duel parameters on 2 separate clients. (5 multiplication) and (1 multiplication) | functional | Both connections to the matchmaking server should be stored in the searching clients list. The two clients should not be matched to each other.. | As expected. |  |
| 37 |  | search for identical duel parameters on 2 separate clients. (5 multiplication) and (5 multiplication) | functional | Both connections should be removed from the searching clients list and a set of questions are generated server side, the game is then loaded client side with identical questions. | As expected. |  |
| 38 |  | search with different duel parameters for 11 clients and close the connection to the 3rd after searching with the 4th client, close the connection to the 6th and 9th clients after searching with the 10th client. Also a 12th client should search for identical parameters to one of the 10 other clients. | volume | The check connection function should remove any dead connections from the currently searching clients list after every new connection. If a match is found both the matched connections should be removed from the currently searching clients list. length of currently searching connections list after each search is initiated: 1, 2, 3, 4, 4, 5, 6, 7, 8, 9, 8, 7 | As expected. |  |
| 39 |  | complete a fractions duel with 5 multiplication questions. | functional | The user that completes the set of questions first should gain elo and the loser should lose elo based on current elo rating of both players. The time taken and the loser’s remaining questions should be displayed. The users’ elo should be updated in the database and the results of the duel are saved to the database. | The time taken displayed for each client is different by one second sometimes. | This is a result of very slight server lag where one client is told the game has ended before the other. This change could be as minute as 0.001 seconds, however if that makes the time taken for one client to be 1.499 seconds and 1.5 seconds for the other the times taken would be rounded up and down to make their time taken be off by 1 second. This happends so rarely and does not effect anything really, because only one time is saved to the database. |
| 40 |  | play a duel with 20 quadratic expansion questions | functional | The enemy’s score should be displayed and updated frequently, the duel should end once a player has answered all questions correctly, the questions should be marked instantly after being correct. | As expected. |  |
| 41 |  | mark a worksheet with 20 quadratic expansion questions | functional | The correct questions should have a tick next to them, the incorrect ones should have a cross and the correct answer next to them. The user should also not be able to mark the worksheet more than once. | As expected. |  |
| 42 |  | close the duel search window whilst searching for a game. With (with 20 quadratic expansion questions) | robustness | The connection to the game server should be closed and neither the program nor the game server will crash. | As expected. |  |
| 43 |  | Close one of the clients connected to a duel in progress with (with 20 quadratic expansion questions) | robustness/functional | Neither the program nor the game server will crash. The remaining player is declared the winner and the two players’ elos should be updated along with the results begin saved to the database with NA for the time. | Can sometimes cause game server to crash. | To make the duel system more robust I must have a better use of socket.set\_blocking() . |
| 44 |  | Open the leaderboards window. | functional | A leaderboard of users with their games won and win % should be displayed. | As expected. |  |
| 45 |  | Search for a game with no way of connecting to the game server. | robustness | An error message should appear saying: “Cannot connect to game servers, check connection and try again.”. Then no duel search should commence | As expected. |  |
| 46 |  | Log in with no way of connecting to the database server. | robustness | An error message should appear saying: “Cannot connect to database, check connection and try again.”. Then nothing else should happen. | As expected. |  |
| 47 |  | Register with no way of connecting to the database server With all fields that are locally validated filled out. (username = Steve, firstname = Steve, secondname = Steve, email = [fsa@fsa.com](mailto:fsa@fsa.com), password = kkkkkkkk!1, password re-enter = kkkkkkkk!1, school code = f, postcode = sl4 5en) | robustness | An error message should appear saying: “Cannot connect to game servers, check connection and try again.”. Then nothing else should happen. | As expected. |  |
| 48 |  | Search for Steve in the profile search window with no way of connecting to the database server. | robustness | An error message should appear saying: “Cannot connect to game servers, check connection and try again.”. Then nothing else should happen. | As expected. |  |
| 49 |  | Complete a duel with a player of elo rating 500 winning against a player of elo rating 1500, with 1 quadratic expansion question | calculation | (1-(1/(1+(10\*\*((1500-500)/400)))))\*32= 31.89…  (-(1/(1+(10\*\*((500-1500)/400)))))\*32= -31.89...  . Therefore the Winner should have their elo increased by 32 and the loser should have their elo decreased by 32. | The Winner had their elo increased by 31 and the loser had their elo decreased by 32. | There is clearly a rounding error here where the winner has their increased elo rounded down. |
| 50 |  | Complete a duel with a player of elo rating 1260 winning against a player of elo rating 832, with 1 quadratic expansion question | calculation | (1-(1/(1+(10\*\*((832-1260)/400)))))\*32= 2.51…  (-(1/(1+(10\*\*((1260-832)/400)))))\*32= -2.51...  The Winner should have their elo increased by 3 and the loser should have their elo decreased by 3. | The Winner had their elo increased by 2 and the loser had their elo decreased by 3. | There is clearly a rounding error here where the winner has their increased elo rounded down. |
| 51 |  | Lose internet connection whilst searching for a duel | robustness | An error message should appear saying: “Cannot connect to game servers, check connection and try again.”. Then the duel search window should close. | As expected. |  |
| 52 |  | Hover the mouse over the profile picture on “My Profile” window, then take the mouse away. | functionality | An edit icon should appear over the profile picture when hovering over the profile picture, then should disappear after the mouse is taken away. | As expected. |  |
| 53 |  | click on profile picture in “My Profile” window, then try and upload a non-image file type.(A text file) | robustness | An error message should appear saying: “Invalid file type”. Then nothing should happen. The profile picture is not uploaded to the website. | As expected. |  |
| 54 |  | click on profile picture in “My Profile” window, then exit the file selector window. | robustness | An error message should appear saying: “FIle not selected”. Then nothing should happen. The profile picture is not uploaded to the website. | As expected. |  |
| 55 |  | click on profile picture in “My Profile” window, then chose an image file. Reload the “My Profile” window to check the image was uploaded successfully. | functionality | After selecting an image, a a message box asking if the user would like to reload the profile window appears. Also, a resized (128x128) image of the image file selected is uploaded to http://spackwell.hol.es/billymaths/profile\_pics/ under the correct user\_id. When the user reloads the page, the correct resized image is displayed as their profile picture. | As expected. |  |
| 56 |  | Hover the cursor over and click on the profile picture in a profile window loaded form a profile search. (Search for Tommy on the account: Steve) | functionality | No edit icon should appear over the profile picture when the cursor is hovering over it and no file selection window should appear when the profile picture is clicked. | As expected. |  |
| 57 |  | Press the duels button on a profile. | functionality | A table with details of the user’s duel history should be displayed. | As expected. |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 58 |  | Press the worksheet button on a profile after pressing the duel button. | functionality | A table with details of the user’s duel history should be displayed. | As expected. |  |

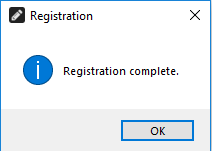
4.2 Test Evidence

1 and 2)

input:

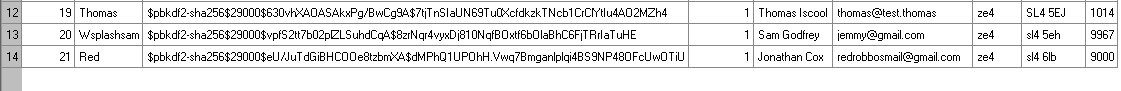


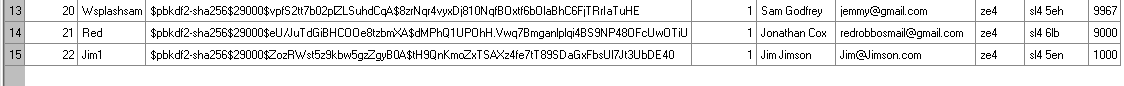
Message box displayed after input:



After this message box is closed the registration window is closed and the login window is left.

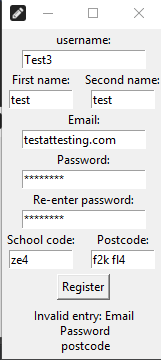
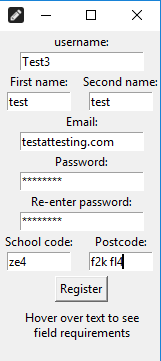
End of user table before input:



End of user table after input:

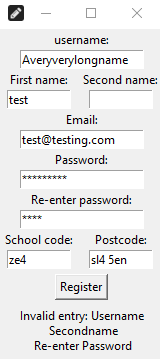
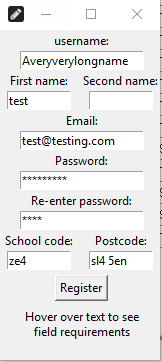
3)

Before register is pressed: After register is pressed:

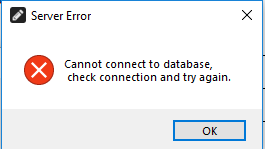


4)

Before register is pressed: After register is pressed:



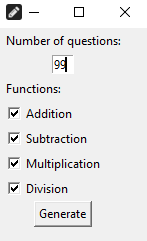
5)



Once this window is closed the program does not crash and the user can try to login again. No restart of the program is necessary because the database is accessed when needed, the connection is not constant.

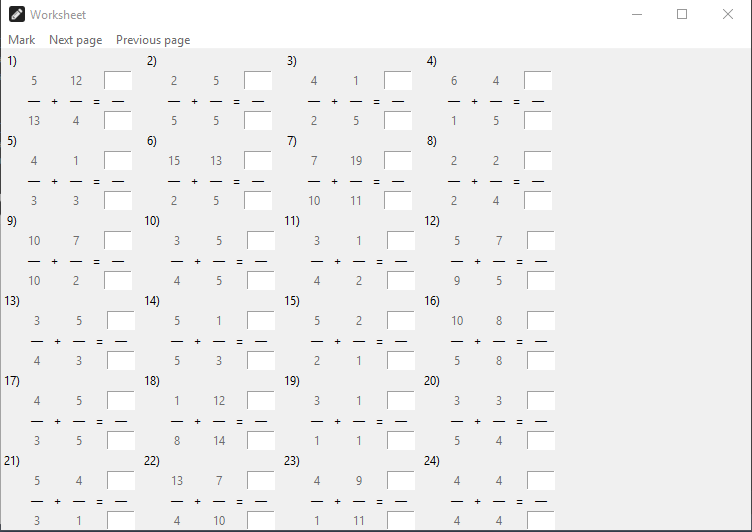
8)

parameters entered:

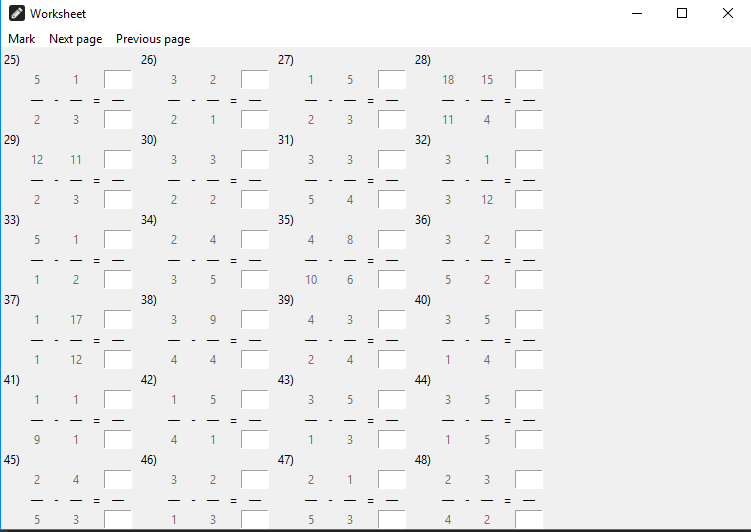


worksheet generated:

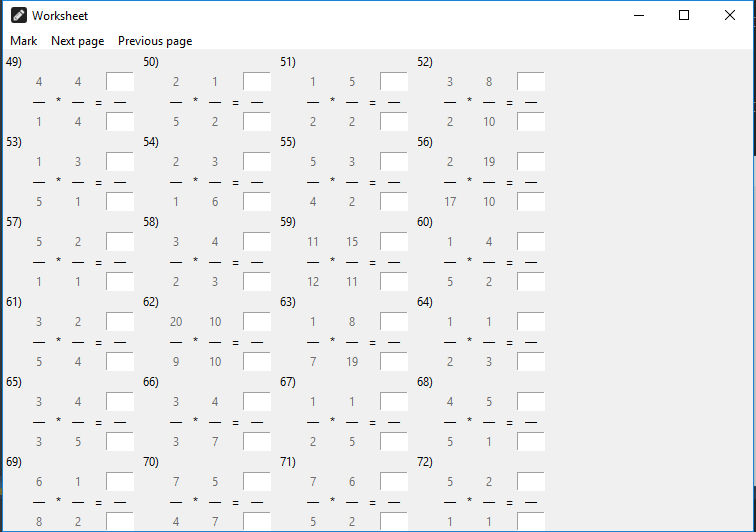
page1:



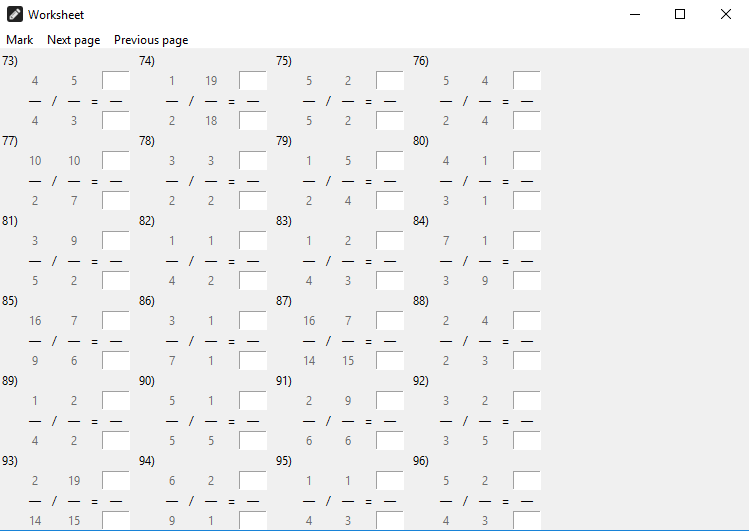
Page2:



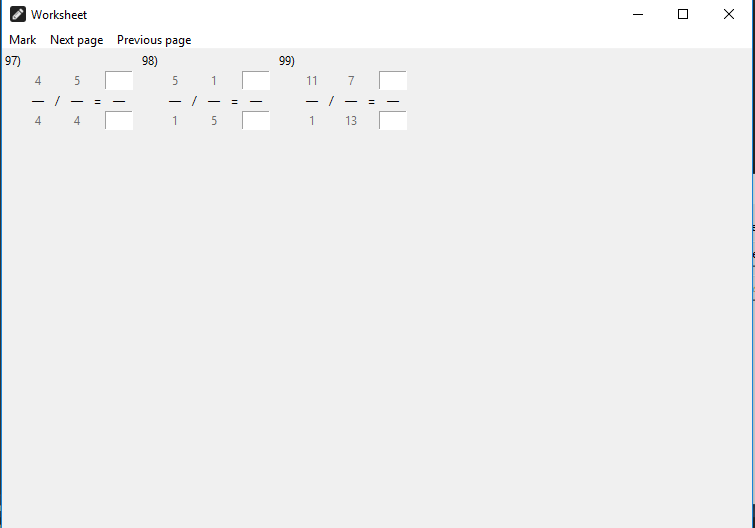
Page3:



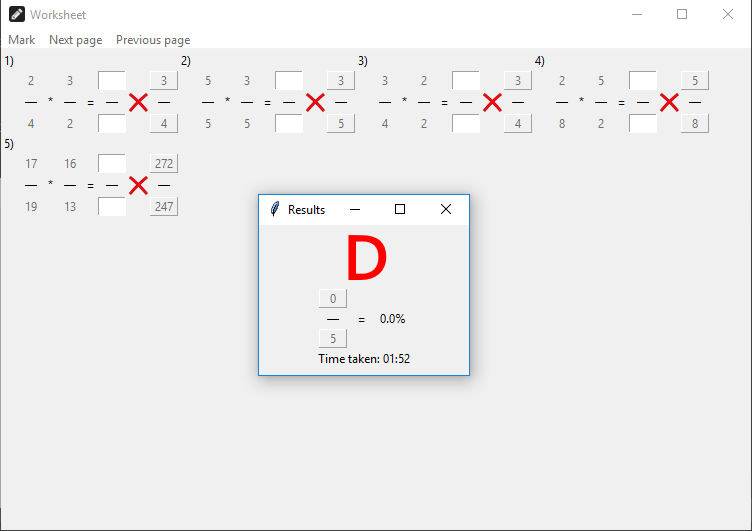
Page4:



page5:

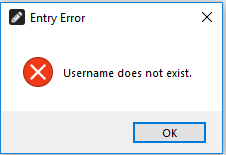
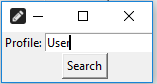


9)



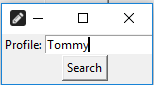
13)

input: result:

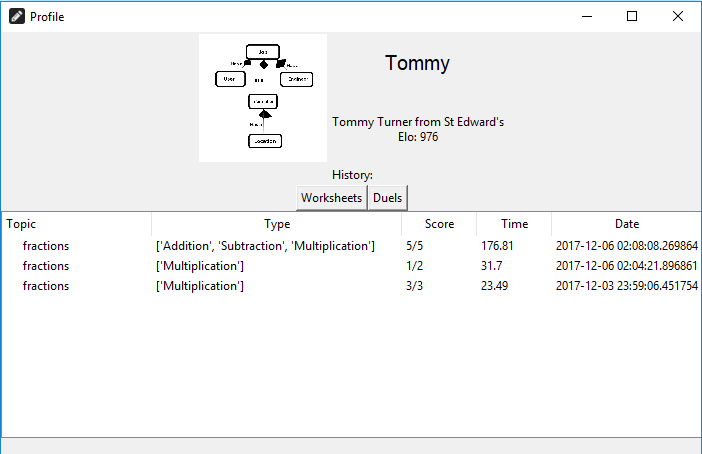


14)

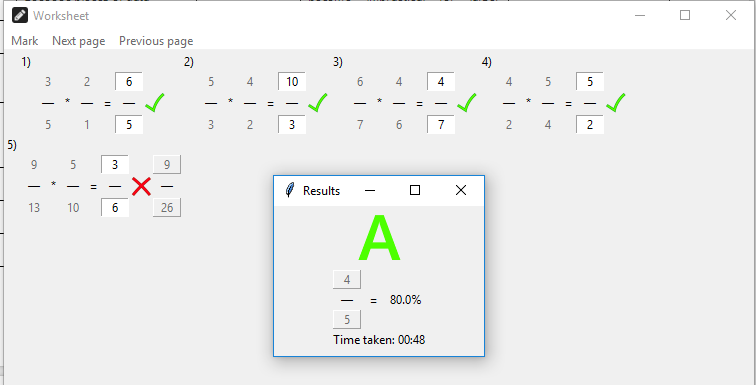
input:



result:



17)

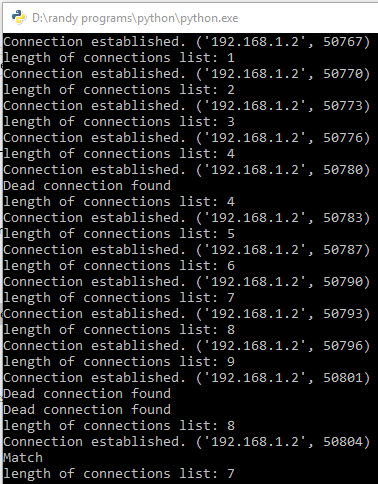


14(0:00), 15(0:06), 16(0:10), 37(0:18), 38(0:42), 40(0:53), 49(1:32), 11(2:20)

This video shows a user logging in with an incorrect username, then an incorrect password, then successfully logging in. Then the two clients complete a duel. the I did in fact mix up the clients when the duels were generated and overlapped, but everything functions as expected. Play with sound for commentary: https://youtu.be/HDVTawTQFCM

39)

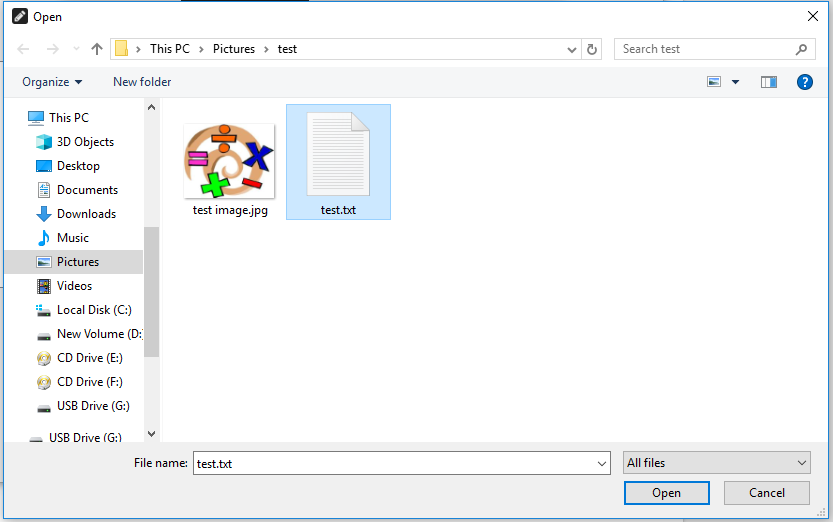
Game Server console output after all searches have been made:



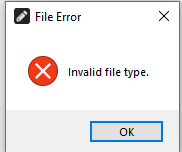
Each time a connection is made between a client and the game server the connection is printed. If the server discovers a dead connection in the connections list “Dead connection found” is printed, then the dead connection(s) are removed from the connections list. If the server finds matching search parameters between the new connection a connection in the connections list “Match” is printed. If there are no dead connections found or matches found nothing is printed. The length of the connections list is printed out after all connections have been checked.

53)

File selected:

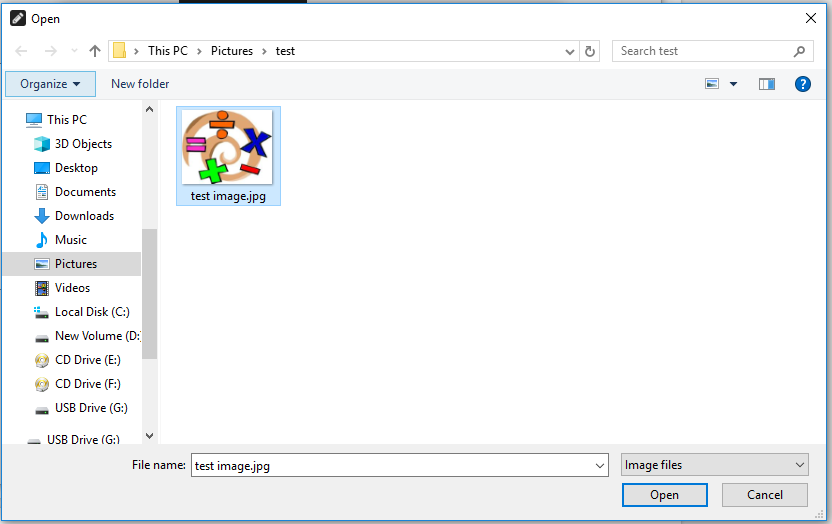


Result:

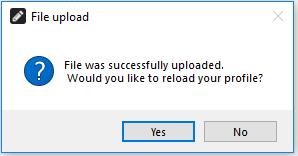


55)

File selected (the image is 250x207):



result:



Profile info before upload: Profile info after upload:



Image on website before upload:

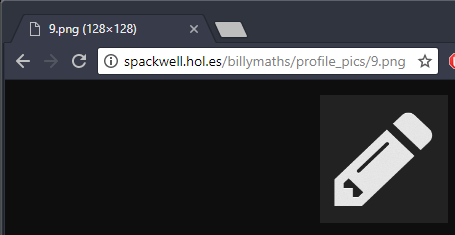
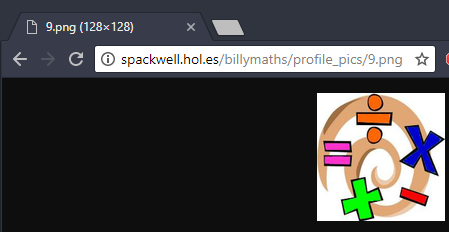
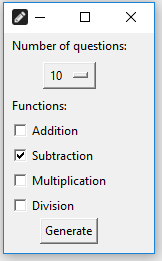


Image on website after upload:

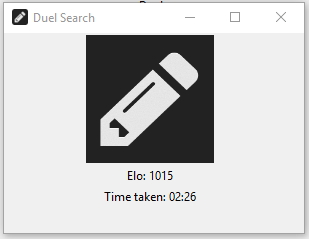
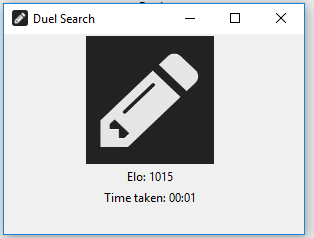


35)

input:



result: result after 146 seconds:



10)

I devised a quicksort algorithm and wanted to test its speed for vast quantities of data, therefore I also created an automated test which creates a list of random integers ranging between 1-10 with x number of elements, then the time the quicksort takes to sort the list is calculated. 3 tests are done, and an average of the times is calculated and printed.

def test(tests, volume):

times = []

for run in range(tests):

li = []

for i in range(volume):

li.append(random.randrange(10))

start\_time = time.time()

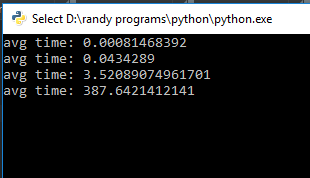
quick\_sort(li)

end\_time = time.time()

times.append(end\_time - start\_time)

print("avg time: " + str(sum(times)/tests))

test(3, 100), test(3, 1000), test(3, 10000), test(3, 100000), test(3, 1000000)



5.Evaluation

5.1 objectives

1. Each user will have a unique user\_id, username, password, name, email, school, postcode and competitive elo rating stored.

Achieved in full. (testing)

1. The password will be encrypted so that even if the database is compromised, the passwords are still relatively safe.

Achieved in full.

1. Make it possible to host the database separately form the client and the client connects to the database through a database server

Achieved in full.

1. Worksheet results from every worksheet users complete should be saved to the database in a separate table with information on: the time taken, topic the questions are on, types of question used, number of questions correct, number of questions total and the date which the worksheet was completed on.

Achieved in full.

1. Duel results from every duel users complete should be saved to the database in a sperate table with information on: the time taken, topic the questions are on, types of questions used, number of questions, winner’s user\_id, loser’s user\_id and the date which the worksheet was completed on.

Achieved in full.

1. Have worksheets where users practice mathematical questions.

Achieved in full.

1. Have a marking feature for the worksheets that tells the user which questions they have correct.

Achieved in full.

1. Have the marking feature show the correct answer next to incorrect answers.

Achieved in full.

1. Have the marking feature display a tick next to correct answers and crosses next to incorrect answers.

Achieved in full.

1. Have a results window appear after marking a question that displays the user’s score out of how many questions there were

Achieved in full.

1. Have the results window display a grade from A-D based on the % of correct answers they have.

Achieved in full.

1. Have the results window display the time taken to complete a worksheet.

Achieved in full.

1. Generate worksheets on fractions.

Achieved in full.

1. Generate questions for adding fractions.

Achieved in full.

1. Generate questions for subtracting fractions.

Achieved in full.

1. Generate questions for multiplying fractions.

Achieved in full.

1. Generate questions for dividing fractions.

Achieved in full.

1. Have a 1v1 duel mode

Achieved in full.

1. Have a competitive elo system for winning and losing duels

Achieved in full.

1. Have the elo of each user involved updated after a duel.

Achieved in full.

1. Have a results window at the end of a duel that displays your opponent’s username

Achieved in full.

1. Have the duel results window display whether you have won or lost.

Achieved in full.

1. Have the duel results window display your opponent’s profile picture

Achieved in full.

1. Have the duel results window display your opponent’s elo.

Achieved in full.

1. Have the duel results window display your old elo, the change in elo based on the outcome of the game and the elo of your opponent and your new elo.

Achieved in full.

1. Make the duel mode work over the internet using a client-server model

Achieved in full.

1. Make the duel mode automatically mark each question as the user completes questions

Achieved in full.

1. Make the marking feature not display a cross or the correct answer next to the question for duels

Achieved in full.

1. Have duels on the fractions topic

Achieved in full.

1. Generate worksheets on quadratics.

Achieved in full.

1. Generate questions for factorising quadratic expressions

Achieved in full.

1. Generate questions for expanding quadratic expressions

Achieved in full.

1. Have duels on the quadratics topic

Achieved in full.

1. Users can register new accounts with their own username, password, email, postcode, school code and real name.

Achieved in full.

1. Users can log in with their own account using their username and password.

Achieved in full.

1. Have a flat colour scheme

Achieved in full.

1. Make it so that the user can chose the colour theme they wish to be on the main menu and profile windows by changing the settings in a config file before launching the program.

Achieved in full.

1. Make a splash screen appear whilst the program loads all necessary widgets

Achieved in full.

1. Make it so that the splash screen can be enabled/disables by changing the settings in a config file before launching the program

Achieved in full.

1. Create a profile page where users can view their elo.

Achieved in full.

1. Have the profile page display the user’s duel results.

Achieved in full.

1. Have the profile page display the user’s worksheet results.

Achieved in full.

1. Make a click on the tabs of each column in a history table sort the table in alphabetical order of the clicked-on column

Achieved in full.

1. Make a double click on the tabs of each column in a history table sort the table in reverse alphabetical order of the clicked-on column

Achieved in full.

1. Each user should have a profile picture

Achieved in full.

1. Each user should be able to change this profile picture on their own profile by clicking on their profile picture.

Achieved in full.

1. An image of an edit symbol should appear of a user’s profile picture when a user’s cursor is over the picture.

Achieved in full.

1. Host the profile pictures on a website where each image has the end of the url as their user\_id.

Achieved in full.

1. Have the pictures uploaded by users resized to 128x128 before uploading without changing the source file chosen.

Achieved in full.

1. Have a default profile picture uploaded for new users.

Achieved in full.

1. Have a leader board where users can see their rank globally and among their classmates.

Achieved partially. The leader board system is not fleshed out as much as I would have liked, there is no option for searching via different schools, there is only a global leader board. To achieve this objective fully, I must make more features related to the leader board such as having a more appealing UI, more options for ordering the table and fix some elo ordering errors that occur with lower elo.

1. Have the main parts of the program inaccessible without logging in with a valid username and password.

Achieved in full.

1. Include a range of topics for students to practice on.

Achieved partially. I currently only have two topics to practice on (fractions and quadratics), although they are done in relatively good depth, more topics would defiantly improve the overall functionality of my program.

1. Have a robust client-server model with the database server

Achieved in full.

1. Have a robust client-server model with the game server

Achieved partially. There is still an error that can cause the game server to crash when closing the duel window before completing a duel.

1. Users should be able to lookup other users’ profiles

Achieved in full.

Summary

I believe that what my program does it does it to a good degree of quality, however it does lack in overall content such as: more topics for worksheets and duels, also a good competitive leader board system with proper search functionality. The overall profile design and uses are pretty much fully developed and done in decent detail like the profile picture and worksheet/ duel history features.

5.2 An evaluation of the whole solution compared with the start

To begin with the hardest part of developing the project was initially figuring out how my OOP structure was going to work and how all my frames and windows linked to one another. After a core structure was formed it became much easier to inherit from different existing objects to create large volumes of content in smaller amounts of time and overall effort. The hardest thing that I found about the project was using tkinter to create any kind of semi-decent looking user-interface, however this too became increasingly easier to do as I had more experience with it and I believe now that my user-interface is clear and simple enough for any kind of user to navigate through and utilise. The client-server models I use I believe are efficient and robust enough for small volumes of traffic, however for larger volumes of traffic on the game server there may be issues and multiple game servers would probably need to be used to spread the load.

.

5.3 User Feedback

Jasper Morgan

Student

07447003633

39 devereux road

Windosr

Sl4 1jj

|  |  |  |
| --- | --- | --- |
| Main objective | completed | comments |
| 6-Have worksheets where users practice mathematical questions. |  | The worksheets are very robust and function well to supply a large volume of questions, a difficulty setting would be good. |
| 7-Generate worksheets on quadratics. |  | This feature works relatively well, however there is an issue regarding the entries when using tab to switch between them that makes some correct answers be marked wrong. |
| 13-Generate worksheets on fractions. |  | This feature is very robust and fleshed out. |
| 53- Include a range of topics for students to practice on. |  | There is not really a wide enough range of topics maybe 2 or 3 more make this a very functional program. |
| 55-Have a robust client-server model with the game server |  | I could not find any ways to crash either the database server or the client. |
| 40- Create a profile page for each user |  | The profile pages are clear and have a good amount of information on them. The personalisation of the profile picture is nice as well. |
| 3- Make it possible to host the database separately form the client and the client connects to the database through a database server |  | This has been achieved fully and no noticeable errors occurred. |
| 1. Make it so that the user can chose the colour theme they wish to be on the main menu and profile windows by changing the settings in a config file before launching the program. |  | The functionality of having colour themes base on hexideimal RGB values as well as names is nice. |
| Have a 1v1 duel mode |  | This mode is very fun and can be quite tense when playing against someone of a similar skill level. |
| 19-Have a competitive elo system for winning and losing duels |  | This feature is developed nicely with no noticeable errors, it does create somewhat of a competitive atmosphere when seeing other people’s profiles and their elo. |
| 51-Have a leader board where users can see their rank globally and among their classmates. |  | I wish that the leader board system worked better and with more features as that would add a lot to the competitive feel of he duels. |

5.4 Analysis of user feedback

The user is happy with the overall functionality of the duels and worksheets as well as the profile features, however they lack some finishing touches, like certain issues that I neglected to test regarding the entries not working well when using tab to change entry on a quadratics worksheet. The leader board system is mentioned to have been desired as it would further the competitive feel of completing duels and gaining elo compared to other users. There should also be more topics to do questions on.

Issues regarding robustness can be fixed relatively easily with further testing and further feedback from different people that may notice things that I would not when testing on my own. The leader board system should not be too much work as the database server is easy to intact with and I have a good method of dealing with tkinter tree views. I do not believe a difficulty setting is necessary, however I would consider implementing it as it does not seem too complicated.

5.5 Extensions and improvements

Extensions I could add would involve a system that utilises the worksheets more, like a teacher setting work for students on a specific topic with certain parameters. There would be a due date and a system where the teacher could monitor students’ progress. A new database table would need to be added and a sperate client for teachers where they have UI that deals with allocating classes/ individual students with certain worksheets.

Improvements I could make in order of priority would be: making content on more topics like sequences, square numbers, Pythagoras theorem, rounding and graphs, these topics can probably be easily implemented due to my OOP structure making it easy to create large volumes of content. I would just need to create algorithms that can calculate suitable questions and accurate answers systematically.

leader board system improvements would involve better tools to search for say users by school and by age possibly. These would require a greater use of SQL queries and more client-database server interaction. The UI for the leader boards would also need to be clearer and include more information about each user.

6.Appendix

6.1 Appendix transcripts

Interview-1 (Sarah Elderfield 20/01/2017)

1- Q: What responsibilities does your role within the school you work at involve?

A: I work as a teaching assistant at a middle school covering years 7-8.

2- Q: What area of your school would benefit from being computerised?

A: Giving the pupils another alternative revision tool to use.

3- Q Would your colleagues be willing to use the program to make their jobs easier? If so, then would they be willing to test and evaluate my program?

A: I could show them and am confident that they would give useful feedback.

4- Q: Would a program that creates procedurally generated activities be easier to set work for rather than having to come up with mathematical questions manually?

A: We don’t necessarily always come up with questions manually, we do use online resources already, to print off homework. However, it would be easier in that printing off 30 worksheets and then having to manually mark them all would be eliminated by automatic generation and marking of homework.

5- Q: Does your school have a similar kind of system in place at the moment? If so then what issues/ possible improvements do you have for that system?

A: We had introduced a similar concept (a website with educational games) but what we had found out was that it was rather shallow academically. We still do however use mymaths which is the opposite; in depth but pupils lack the encouragement to invest more time into it.

6- Q: Do you think this will improve the rate of students actually completing their homework?

A: I believe more students would complete their homework because there would be less complications and excuses for absences of work.

7- Q: What administrative resources would you find beneficial implementing?

A: The ability to see how long a student spent on a piece of work, the amount of extra practice students partake in, when students complete the work and being able to set work.

8- Q: What do you find most problematic with the current system?

A: Not enough students fully complete the work set and they complain about not having enough help outside of lessons that is easily accessible.

Interview-2 (Kirsty Elderfield 21/01/2017)

9- Q: What year are you in and what level of maths do you work at?

A: I am in year 8 and study key stage 3 maths.

10- Q: What topics do you struggle with the most in maths?

A: trigonometry, sequences, fractions and algebra.

11- Q: What would incentivise you more into learning maths?

A: If maths was more fun and if different topics would be explained in a simpler way for me to understand, more interesting activities that I can compare results with my friends

12- Q: What learning resources would you find beneficial implementing?

A: Play maths games, revision guides, practice questions. A summary of a topic with links to helpful webpages as well as worked examples of step by step instructions on how to solve things.

13- Q: Do you think this will improve the rate of students actually completing their homework?

A: Yes, but some students might complain that they do not have access to computers/ internet. I believe students would care more about completing work properly if they were to be compared directly against one another.

14- Q: What do you find most problematic with the current system?

A: Some worksheets ask questions that do not fully explain what to do. The lack of support we receive on particular topics and the low amount of incentive pupils feel towards furthering their education.

15- Q: What kind of aesthetics would you like to see on the program?

A: bright colours would be nice and interesting, but nothing too over the top.

16- Q: would your fellow students be willing to test the program?

A: Yes I believe I could get my friends to test it out and give useful information about the accessibility and usefulness of the program.

Interview-3 (Sarah Elderfield 20/02/2017)

17- Q: What information would you think would beneficial in knowing about students’ activities?

A: It would be beneficial to know how much time a student spends on individual activities as well as the amount of overall time spent on extra activities. The marks they receive on each activity, also the dates and times activities are completed.

18- Q: How often do you set work currently and what does it consist of?

A: Each week 1-2 sheets of work regarding topics that students learn in that week are set.

19- Q: How confident are you at using computers? If not, then Will i have to take this into consideration and simplify The user interface?

A: I am rather unconfident using complex programs, but i can navigate through nicely laid out user interfaces with relative ease.

20- Q: (Question related to Q-4) Would this increase the amount of work that a teacher is able to set for a class?

A: Yes i think that a teacher would be able to set more work for a class due to the added ease, but i also think that the time a teacher saves because of this is more important because it allows them more time to focus on preparations for the lesson rather than stressing about homework.

21- Q:What do you like the most about my proposed program?

A: The feature to monitor the progress of each student seems very simple and appealing.

22- Q: Do you think that i should implement a private profile function for those who want to play the games without being compared to other students?

A: Yes there should be, because some students that have difficulties in maths and low confidence would be put off by the idea that other pupils can see their marks. However do not make this feature block teachers’ access to progress information.

23- Q: If a student does not possess the internet or a computer system, are they able to use these resources at school?

A: Yes, in fact a homework club is ran twice a week in the computer rooms.

24- Q: Do you think it would benefit those of a lower level in maths and or with low self confidence?

A: Yes, because they’re not in a classroom atmosphere they might not feel as pressured to complete the work.

25- Q: Do you think there should be a difficulty option that can be chosen for each activity?

A: Yes, (if possible) to allow for the range of mathematical skills that key stage 3 students possess, also to let students challenge themselves if they wanted.

Interview-4 (Kirsty Elderfield 19/02/2017)

26- Q: What categories do you think the leader boards should include?

A: There should be a leader board for each year group and different schools.

27- Q: Do you think there should be a difficulty option that can be chosen for each activity?

A: Yes, because people with different abilities might things too hard or easy so it is good to have a range.

28- Q: If a student does not possess the internet or a computer system, are they able to use these resources at school?

A: Yes, there are multiple computer rooms that are available to use during breaks and after school.

29- Q: What do you like the most about my proposed program?

A: The concept of competing against my classmates whilst improving my maths skills.

30- Q: What types of games would you like to play on my program?

A: Time trialed quick fire questions, creative use of normal games with a maths twist (snap, darts...) and puzzles.

31- Q: Do you think that i should implement a private profile function for those who want to play the games without being compared to other students?

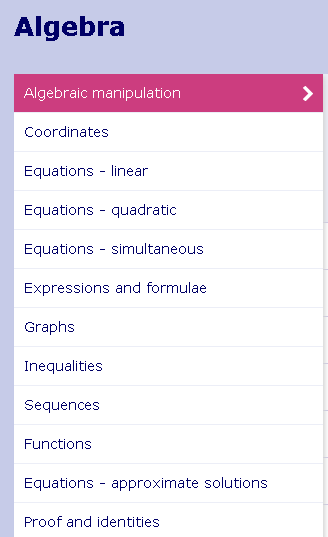
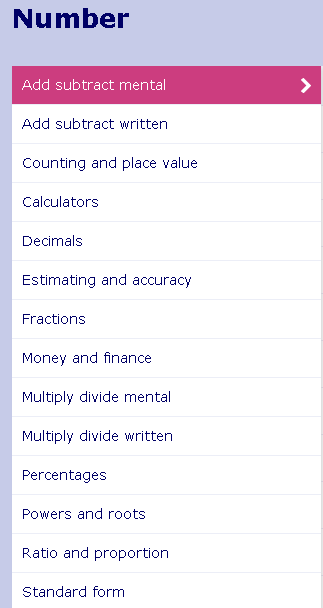
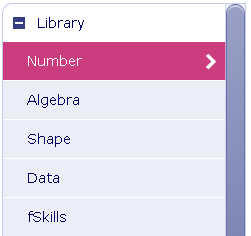
A: Yes, because some students of a lower level would not like to compare themselves to those of a higher level.

6.2 Internet search

MyMaths.co.uk

As I searched for programs/websites that do a similar thing I discovered mymaths does most of what I am trying to accomplish however it lacks certain things I believe I can add to my program that would meet the needs of my users better (which I will explain in more detail in the summary).

The website has a large library of topics to choose from and exercises that range in many different levels of maths.

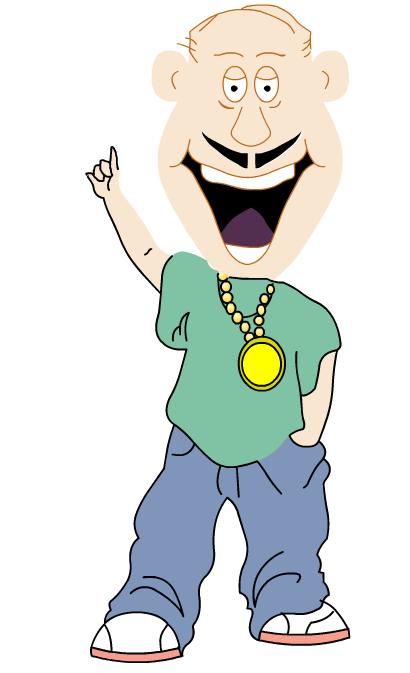


Each exercise has a rundown of the topic and a set of questions that the user can practice. Teachers can also set specific work for students to do for a deadline and have decent monitorisation over the progress for that work.

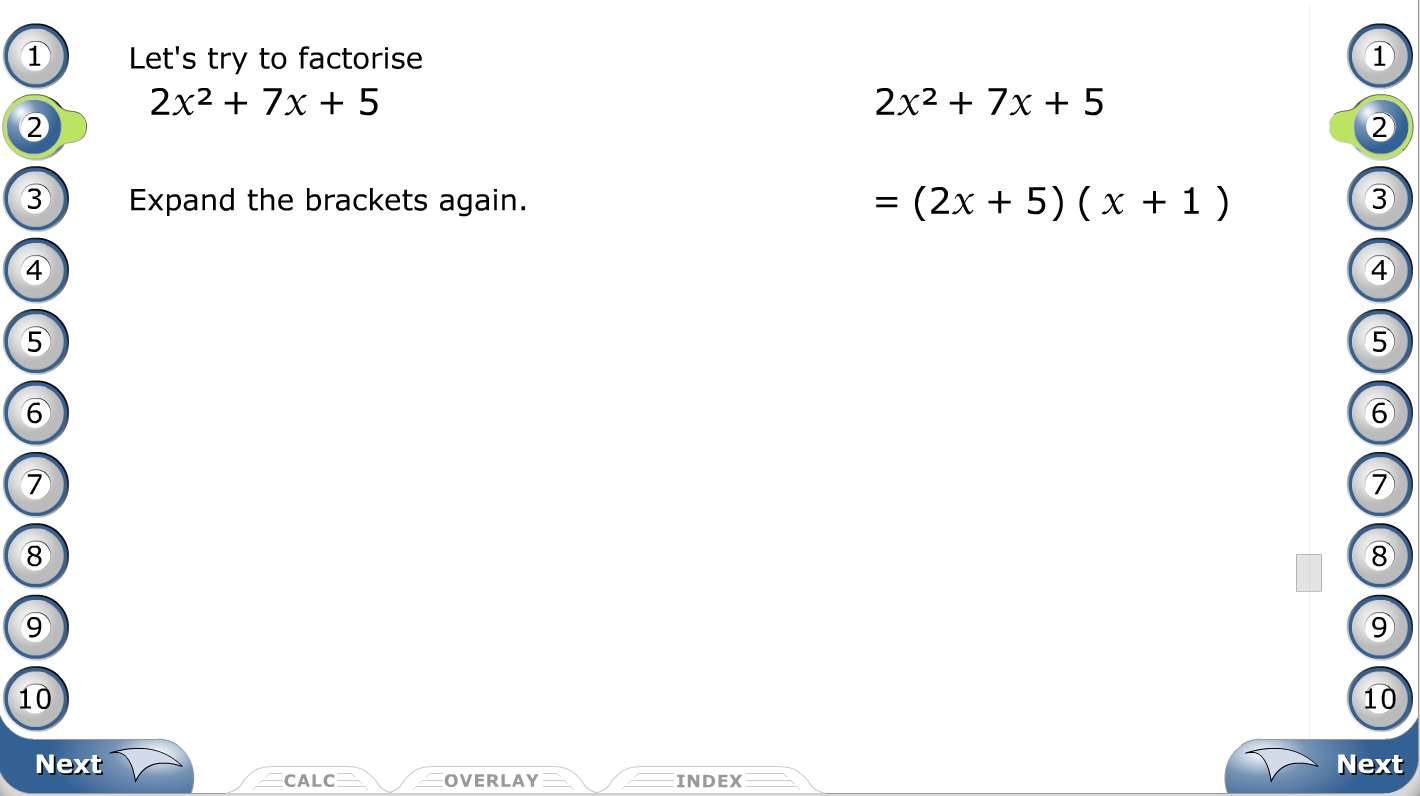


MyMaths does also have a games section, but this is where the website lacks. The games are very visual however they are designed more for primary school children and may be viewed as immature to older key stage 3 students. The games section also does not have a leaderboard system that students can compete in and engage themselves more in practicing maths.

The whole site feels childish, which certainly frustrated me slightly when I used this back in middle and secondary school.



The feature of explaining topics has a very broad scope, however the methods used have a clunky, frustrating user interface and does not describe the topic in enough detail.



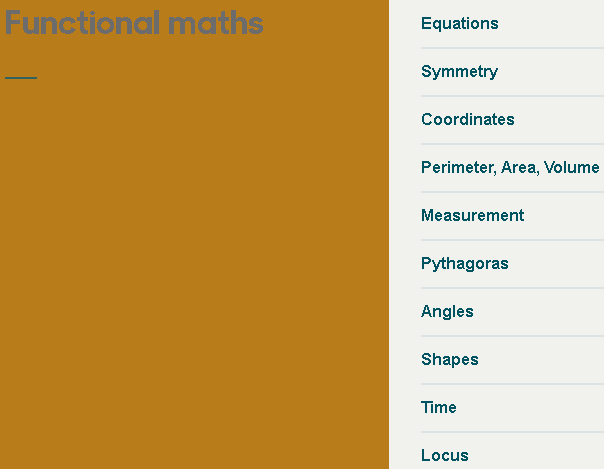
Overall MyMaths is a very good resource for practicing maths skills through routine questions about topics you are already confident on, and gives you a shallow understanding on a wide range of topics. It has a good system that enables teachers to set work, view student progress on that specific work and can use this in lessons for practice questions. It also has a large game library of simple games with limited leaderboard features.

On the other hand, MyMaths lacks in explaining topics fully with a good step by step system.It has a large range in maths levels but is missing in-depth focus. It doesn’t allow teachers to view students’ progress outside of the work that they set. The aesthetics of the site are very immature and the games are too simple.

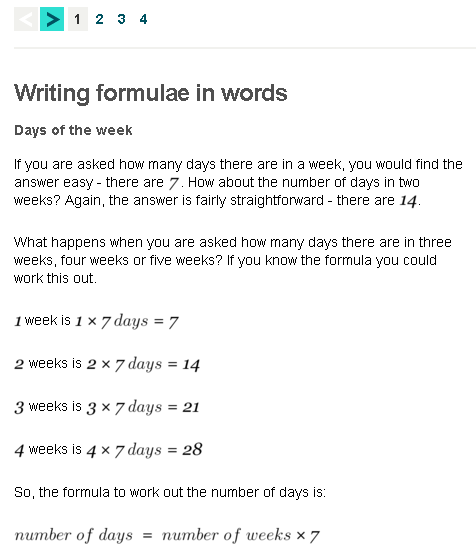
To conclude, my program will take a similar format to MyMaths as it does do a good job at practice questions and structures all their features nicely, but i will improve it to be better suited to my users by: focusing in-depth on key stage 3 topics, Making it look more mature, Making the games more interesting and fun, also have a leaderboard system. Having more monitorisation for teachers.

BBC Bitesize

BBC Bitesize is a great revision guide for key stage 3 with loads of in-depth information on every topic that a student must learn.



The site’s method of explaining the different topics is very clear and understandable. It can go into depth about topics that students find very hard to understand and can also offer just enough information that is needed to jog a student’s memory.



To conclude I like the methods BBC Bitesize use to explain the different key stage 3 topics in decent detail and will try to implement a similar structure to my explanations and will include it in my list of references to revise certain topics. I believe my program will be better than BBC Bitesize in relation to the interactivity the students can have with the activities.