AS3.46 (91637) v Develop a complex computer program for a specified task

Resource title: Student Worksheet and Answer Booklet

1. Type your name in the header of this document.
2. You have until the **end of Week two** in **Term 4** to complete the assessment. This time includes school periods, out of class time and the term 3 holiday period.
3. The test is conducted under **test conditions** in class (no talking).
4. Name your **Student Worksheet** and your **Python code sheet** "first initial, surname + AS3.46 + version number". EG **GHall_AS3.46_v3**
5. Save your completed **Student Worksheet** and your **python code** to a folder in your H: drive named "first initial, surname + AS3.46". EG **GHall_AS3.46**
6. Save your documents (including code) at regular intervals to the H: drive.
7. To hand in the assignment, you need to copy your folder to the appropriate folder in the **13CSI Dropbox**.

Learner	Name		Signature		
Result	**Not Achieved**	**Achieved**	**Merit**	**Excellence**	Date
Assessor	**Gordon Hall** Name		Signature		

Resubmit	**Not Achieved**	**Achieved**	**Merit**	**Excellence**	Date
Assessor	**Gordon Hall** Name		Signature		

Your Project

Evidence

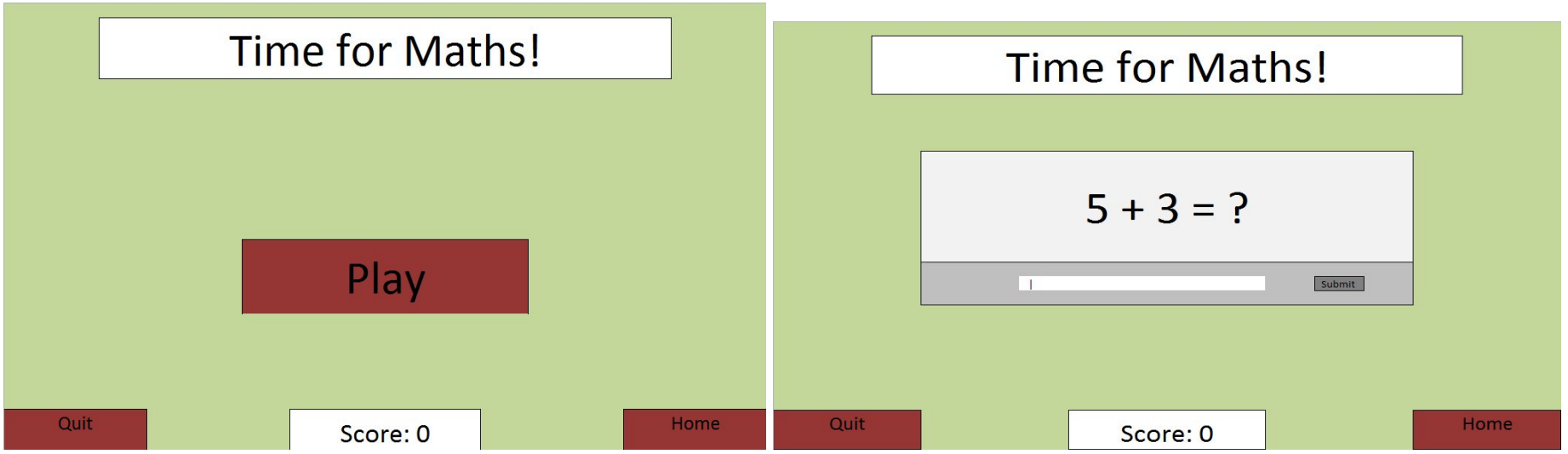
Collect and comment evidence (screenshots) documenting how you have implemented the design and build of this project. The evidence should show that you have:

1. Developed and followed a **plan** (**GUI** (Graphical user interfaces) and **Flow chart**) for building your program by saving successive versions of the **GUI, Flowchart *(Modular Structure)*** and **Python code** with added complexity in each version.
2. Worked in an organised and efficient way, using time effectively and not using a trial-and-error approach unnecessarily.

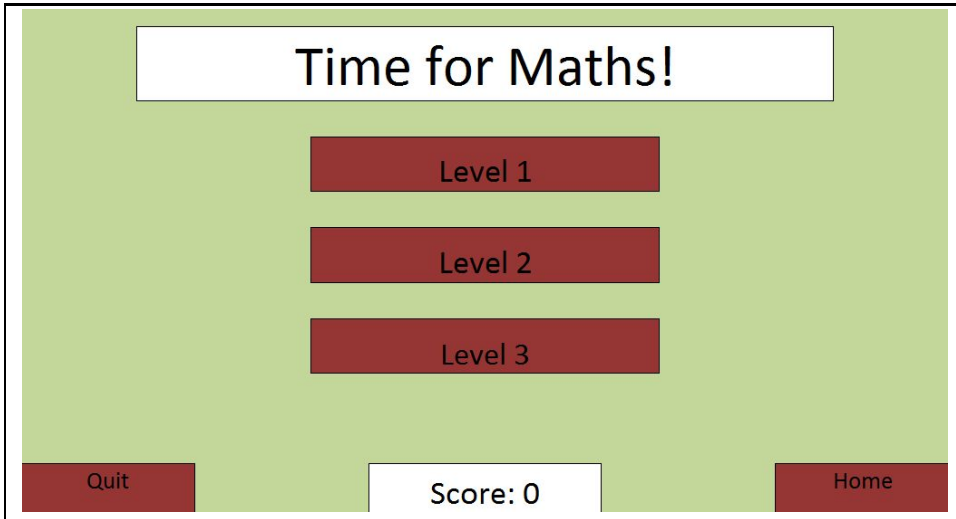
Part 1 - Planning the program

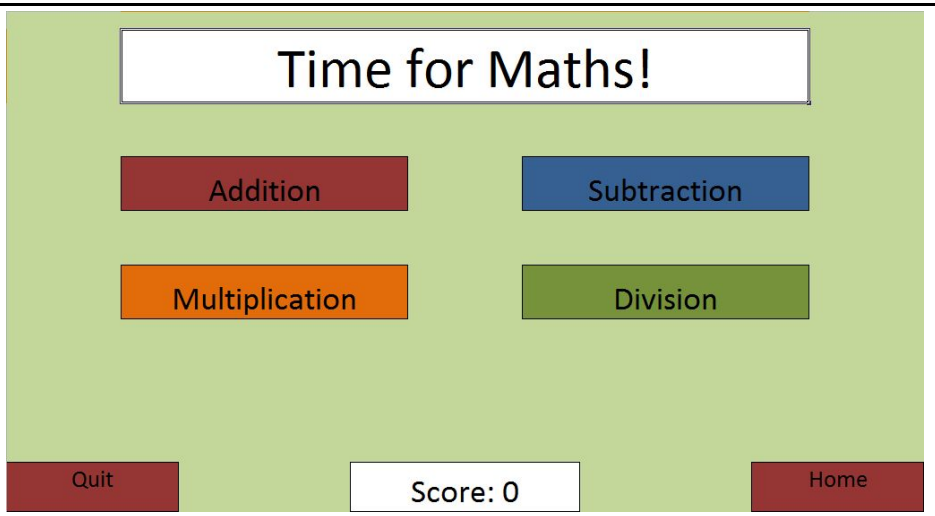
Graphical user interfaces - Version(s)

Version 1



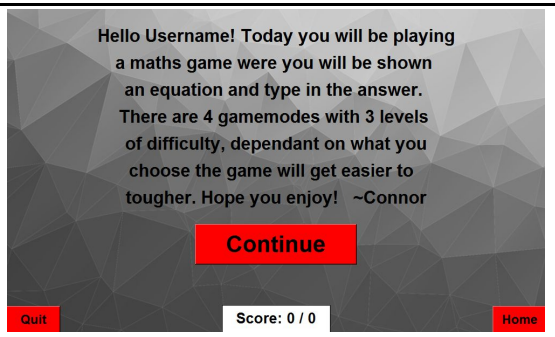
Version 2/3

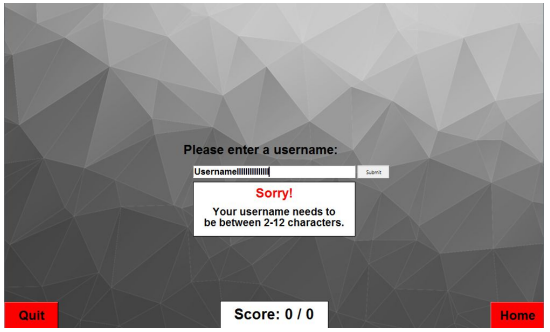




Version 2: Creates the choice of gamemode and level, Version 3: Creates an option to go to homescreen

Final Graphical user interfaces

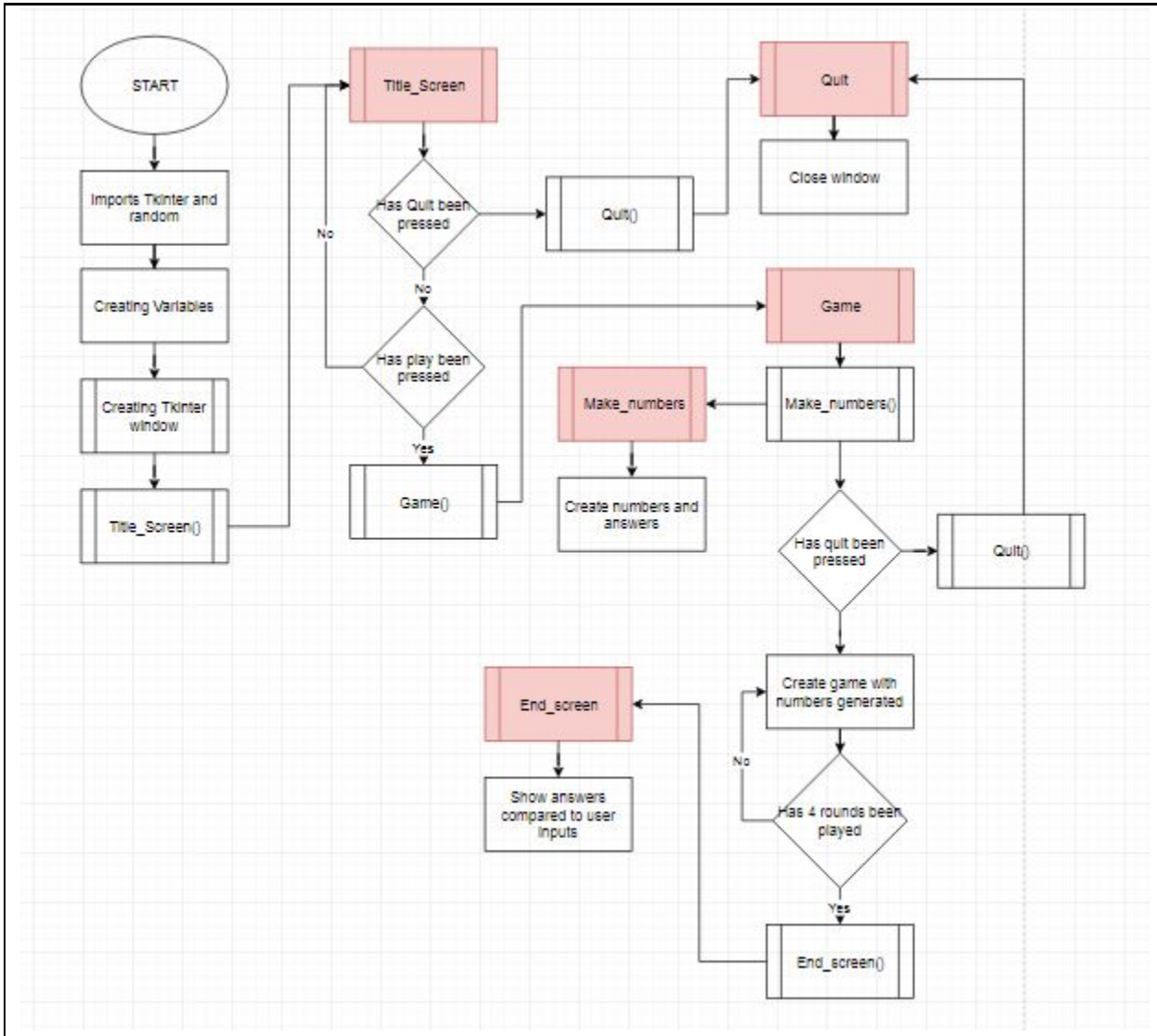




Event generating and handling components	Event generating	Event generated
How clicking and entering text generate an event. What the event is and why it is used/needed	**Place screenshot of the event. e.g. Entering text and clicking a button**	**Place screenshot of the outcome. e.g. The next screen after the button is clicked.**
The event is entering a username, If the submit button is pressed without a valid username (eg: no characters or characters out of limit 2-12) then an event will show an error message, if valid characters are used and submit is pressed then the program will generate an event and take the user to the instructions screen. It is needed to give the user more control over the program and to make sure they know who has what score		After clicking submit (with a username entered) the user will be taken to the instruction screen. The username entered will be used throughout the program.

Commented Screenshots showing your flowchart version(s):

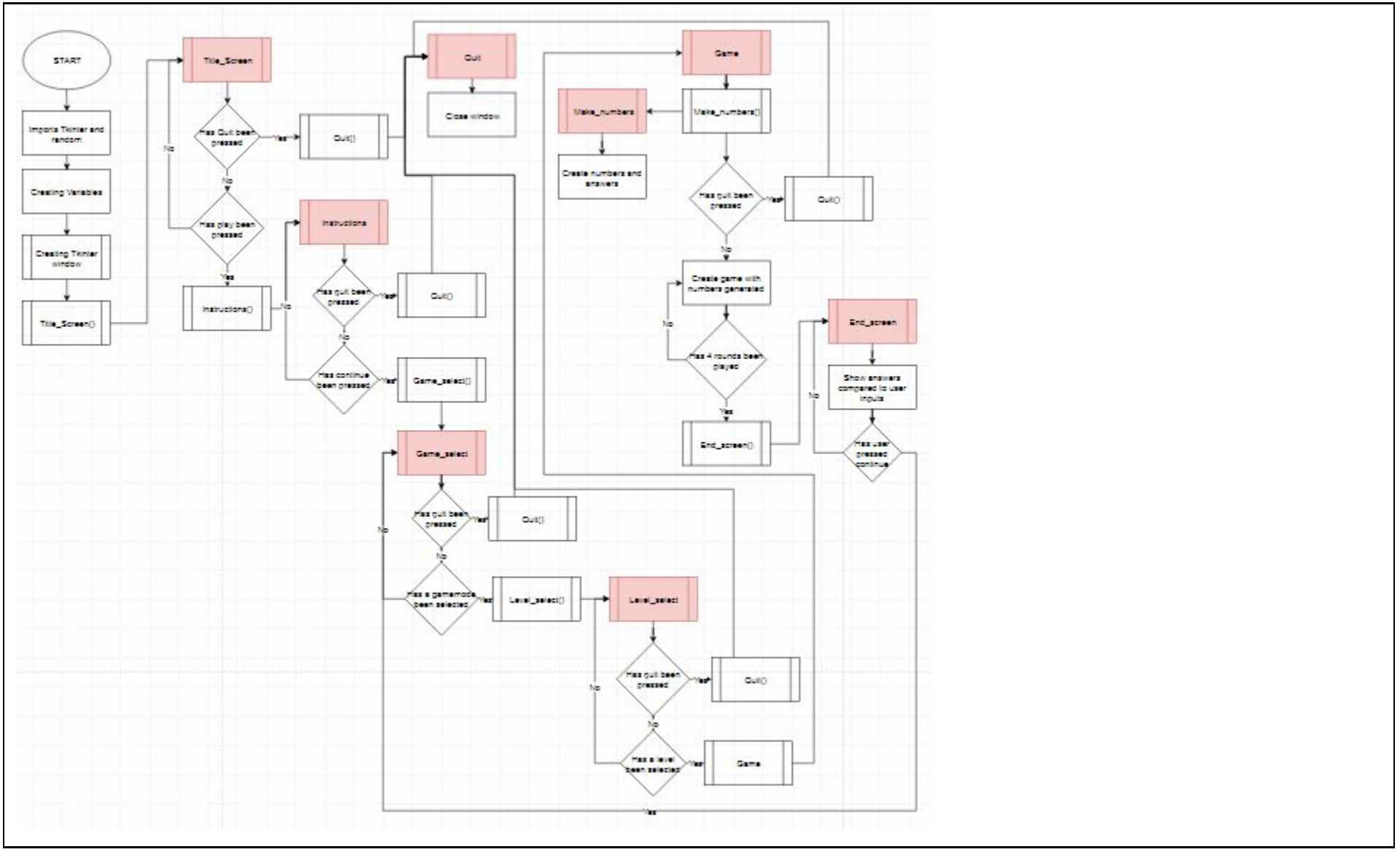
Version 1



https://drive.google.com/file/d/1g13cDz3Ot3D6e6FQ9lBzpKx6wKprY_Lf/view?usp=sharing (Commented version)

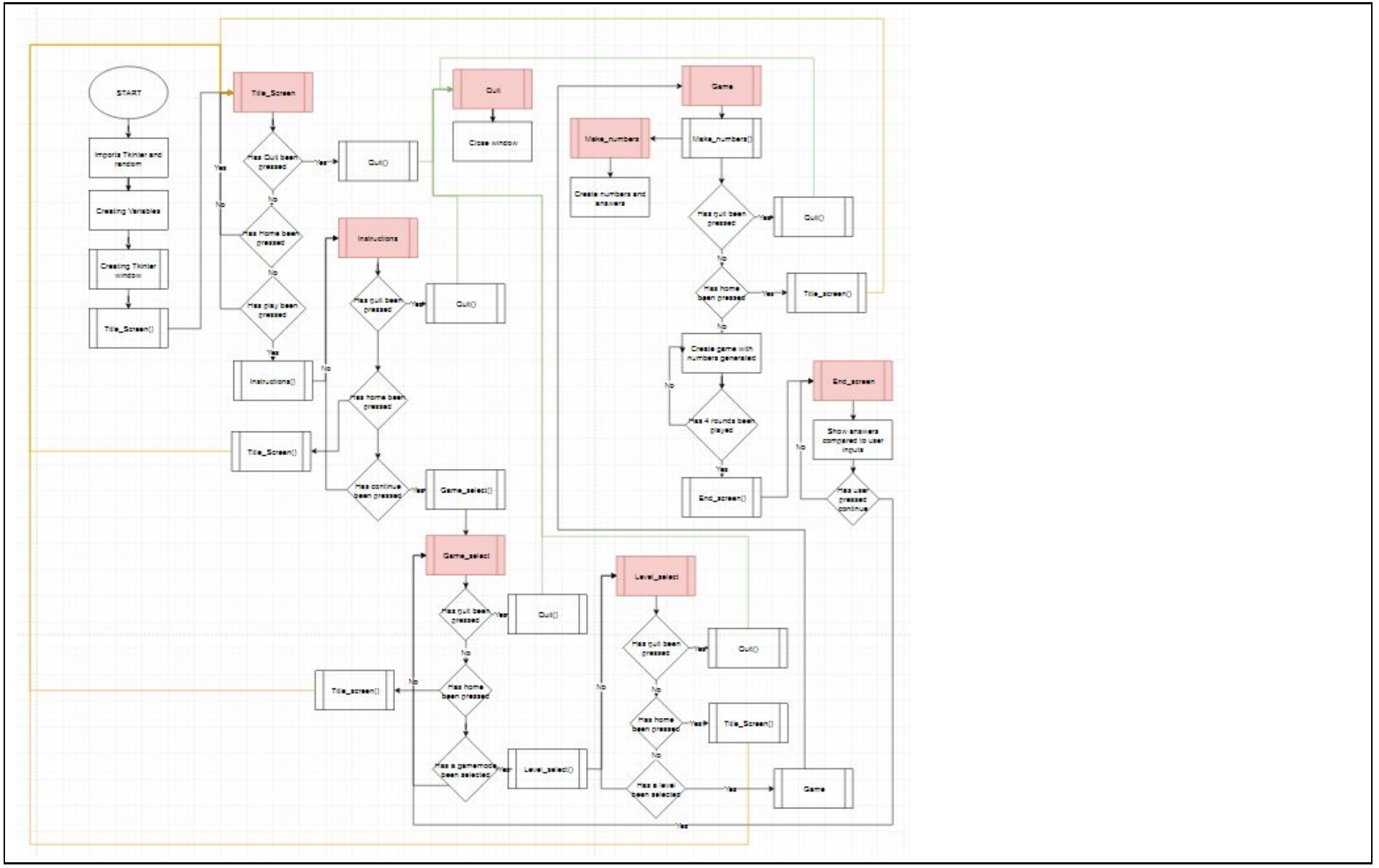
This is a basic algorithm that runs through the main program, going from title screen to the game immediately, after the game ends the user is taken to the end screen where they can see what answers they got right, there is no replay function.

Version 2


START
Imports Tkinter and random
Creating Variables
Creating Tkinter window
Title_Screen()
Title_Screen
Has Quit been pressed
Yes
No
Quit()
Has play been pressed
Yes
Instructions()
Instructions
Has quit been pressed
Yes
Quit()
No
Has continue been pressed
Yes
Game_select()
Game_select
Has quit been pressed
Yes
Quit()
No
Has a gamemode been selected
Yes
Level_select()
Level_select
Has quit been pressed
Yes
Quit()
No
Has a level been selected
Yes
Game
Quit
Close window
Game
Make_numbers
Make_numbers()
Create numbers and answers
Has quit been pressed
Yes
Quit()
No
Create game with numbers generated
Has 4 rounds been played
Yes
End_screen()
End_screen
Show answers compared to user inputs
Has user pressed continue
No
Yes

https://drive.google.com/file/d/1g13cDz3Ot3D6e6FQ9lBzpKx6wKprY_Lf/view?usp=sharing (Commented version)
The second version still contains all the functions from the first version but has an included level and gamemode section where the user cna choose a gamemode and what level of difficulty they can play at. There is also an included instructions screen that will tell the user how to play before they get to the choice of selection of game. The end screen also links back to the gamemode selection screen so the user can continue to play under same name.

Version 3


START
Imports Tkinter and random
Creating Variables
Creating Tkinter window
Title_Screen()
Title_Screen
Has Quit been pressed
Quit()
Has Home been pressed
Has play been pressed
Instructions()
Instructions
Has quit been pressed
Has home been pressed
Has continue been pressed
Game_select()
Quit
Close window
Game
Make_numbers
Make_numbers()
Create numbers and answers
Has quit been pressed
Has home been pressed
Create game with numbers generated
Has 4 rounds been played
End_screen()
End_screen
Show answers compared to user inputs
Has user pressed continue
Game_select
Has quit been pressed
Has home been pressed
Has a gamemode been selected
Level_select()
Level_select
Has quit been pressed
Has home been pressed
Has a level been selected
Title_screen()
Title_Screen()
Game
Yes
No

https://drive.google.com/file/d/1g13cDz3Ot3D6e6FQ9lBzpKx6wKprY_Lf/view?usp=sharing (Commented version)
The third version contains all the aspects of the second version but has an included home button. There are some aspects that are going to be included in the program but cannot be shown in modular structure (ie aesthetics).

Inputs:

- entry (name entry), username(stringvar in entry)
- entry2 (game entry), roundinput(stringvar in entry2)

Outputs:

- name
- symbol
- phrase, phrase2, phrase3
- Total_score
- Total_played

Variables, types and descriptions:

Name	Type	Global/local	Description/Scope
name	string	global	Saves the name thats obtained by the use of .get() in the username entry box, it is used throughout the code
symbol	string	global	When the user picks the gamemode they want (eg, subtraction) symbol will be set the a minus sign. Used in the equation page
phrase, phrase2, phrase3	string	global	All 3 variables have same pupose but contain different data. They add the name to a phrase to be printed around the program (eg: phrase 2 is “(name), Please pick a gamemode”)

intGame mode, intLevel	int	global	These variables have same jobs, when the user picks a gamemode/level then these variables get assigned a certain value used all around the code -> especially in num gen
total_scor e	int	local	This is the total score of the user, it appears in bottom of screen in middle. Whenever user gains score it is added to this value to be displayed
round_sc ore	int	global	After the round is over this variable will be assigned a value that will be added to the total_score
intRun	int	local	runs the game loop 4 times, each time intRun += 1 to make sure it doesnt run any unexpected amount of times
intColour	int	global	Runs the colour loop, 4 times so the endscreen can return red or green if they got answer right/wrong
choice	int	local	choice is only used in number generator code, random number is set as choice and appended to the number lists

Indexed data structures, types and descriptions:

Name	Type	Description/Scope
list_picked1	list	Generated by combining the 2 lists (list_picked2, list_picked3), this list contains the answers to the maths equations that the user needs to guess
list_picked2	list	Depending on what gamemode, this list has random generated numbers and will appear to be the first number in the equation (eg: is x in (x+y=z))
list_picked3	list	The same as list_picked2, all random generated, is the 2nd number in the equation (eg: is y in (x+y=z))
user_picked	list	This is what the user picks as what he thinks is the answers, appended to list and compared with actual answers

Class(es), Objects, types and descriptions:

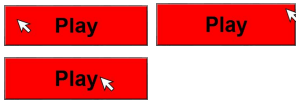
Name	Type	Information/objects Stored	Properties, Functions, Operations, Etc
Open_window	Class	N/A	N/A
def __init__	Definition	self, canvas, root	Generates Tkinter gui, creates canvas specs
Title_Screen	Class	N/A	N/A
Page_one	Definition	canvas, root	Play button to play
Instruction	Class	N/A	N/A
Instruction_screen	Definition	N/A	N/A
Instructions	Definition	N/A	Continue button to continue
Gamemode	Class	N/A	N/A
Gameselect	Definition	intGamemode, symbol	4 gamemode buttons triggering level event
addpick, subpick, mulpick, divpick	Definitions	intGamemode, symbol	Sets symbol and gamemode to corresponding button pressed
Level	Class	N/A	N/A
addlevelselect	Definition	intLevel, (Level1/level2/level3)buttons	3 buttons triggering start of game
levelbutton_colour	Definition	bg colour for level buttons	returns a colour dependant on what gamemode selected
lev1pick, lev2pick, lev3pick	Definitions	intLevel	Sets level to corresponding button pressed, runs number generator code
Container	Class	N/a	N/A
ConstantLayout	Definition	Quit_button, home_button	Runs scoreupdate, creates home/quit buttons
Game	Class	N/A	N/A
creategame	Definition	strAccept, intColour	runs games 4 times

loop	Definition	strAccept	Runs the loop_visual def 4 times
loop_visual	Definition	N/A	Creates boxes for game, runs gameask
gameask	Definition	roundinput, entry2	creates entry box for game and submit button which runs roundcheck def
nextround	Definition	user_picked, intRound, entry2	Deletes entry2, appends their guess, changes round to next round and runs loop again
changescore	Definition	round_score, intScore, round_played	Increases score dependant on whether the user got the answers right, as well as add the total amount of rounds played
end_screen	Class	N/A	N/A
showscreen	Definition	intRound, intScore, continue button	shows user answers, continue button to continue
colourchange	Definition	intColour	Changes user answer colour to green/red if they got answer right or not

Module Names, types and descriptions:		
Name	**Type**	**Description/Scope**
quit_button	Object	A button that allows a user to exit the GUI/Application so they do not have to hard exit or complete the application to the end
home_button	Object	A button that allows the user to go back to the title screen if they want to change user, resetting scores and name
entry	Object	First entry box used in collecting the username
entry2	Object	Second entry box, used in collecting the numbers user enters in the game
def scoreupdate	Definition	Updates the scores whenever the user enters a new page, displayed at the bottom of the page

def ask	Definition/Function	Creates an entry widget during the username entry page for the user to enter a name and be stored
def check_name	Definition	Error catches the name entry to make sure that the user types in a valid name before continuing to the next page
def roundcheck	Definition	Error catches the game entry to make sure the user enters a valid answer during the game phase that can be used to collect score (eg: 42 instead of forty two)
def numberpicker	Definition	Generates all 3 numbers (first number, second number and answer)
def add_finish, sub_finish, mul_finish, div_finish	Definition	Creates the answers for all the equations dependant on whether the user picked any of the 4 gamemodes

Test Plan.

Test Type	Item tested	Test inputs	Expected outcome	Comments
Expected#	username entry (entry)	"username", "USER", "UseR"	all 3 will be accepted with only the first letter capitalised	
Expected#	username entry (entry)	"abc", "abcde", "abcdefghijkl"	all 3 will be accepted as they are all between 2-12 characters	
Expected#	game entry (entry2)	"50", "800", "521"	all 3 will be accepted as they are all within 0-999 range	
Expected#	username entry (entry)	"漢字", "áá", "**Ææ**"	All 3 will be accepted as they are all valid characters	
Expected#	play_button		all 3 will be accepted as they all in the area to be pressed	This is only if one of my expected tests arent sufficient
Boundary+	username entry (entry)	"a", "ab", "abc"	"ab" and "abc" will be accepted because they are in suitable	

			range, whilst “a” is lower than 2 characters	
Boundary+	username entry (entry)	“name”, “na_me”, “na me”	“name” will be the only accepted value whilst the others have invalid characters	
Boundary+	game entry (entry2)	“-1”, “0”, “1”	“0” and “1” should be accepted whilest “-1” shouldnt	
Invalid *	username entry (entry)	“”, “1234”, “n%a$m&e”	all 3 will not be accepted as they all are invalid	
Invalid *	game entry (entry2)	“fifty”, “”, “25e”	all 3 wont be accepted as they are all not integers	
Invalid *	game entry (entry2)	“10000”, “-10000”, “500.244”	all 3 wont be accepted as they are all out of range	

#Achieved *+Merit* **Excellent*

Part 2 - Producing the program

Program version(s):

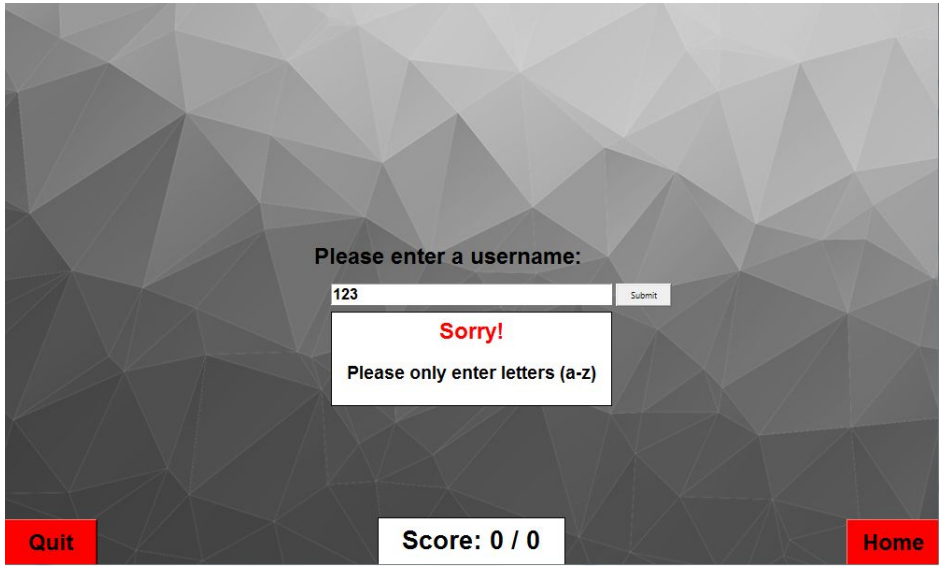
Version 1
Labelled: CdeBruyn_Achieved

Version 2
Labelled: CdeBruyn_Merit
I have added an instruction screen, gamemode selection screen and level selection screen. As well as the endscreen has the option to play again and has a change in colour of answers depending if they got them right/wrong. I added an instruction screen so that the user doesn't get confused when he/she plays with how to play. Level and gamemode selection gives the user more freedom with playing, being able to play at what level of difficulty and type of math they want to learn. Having a replay feature will allow the user to not need to restart the game if they would like to play again.

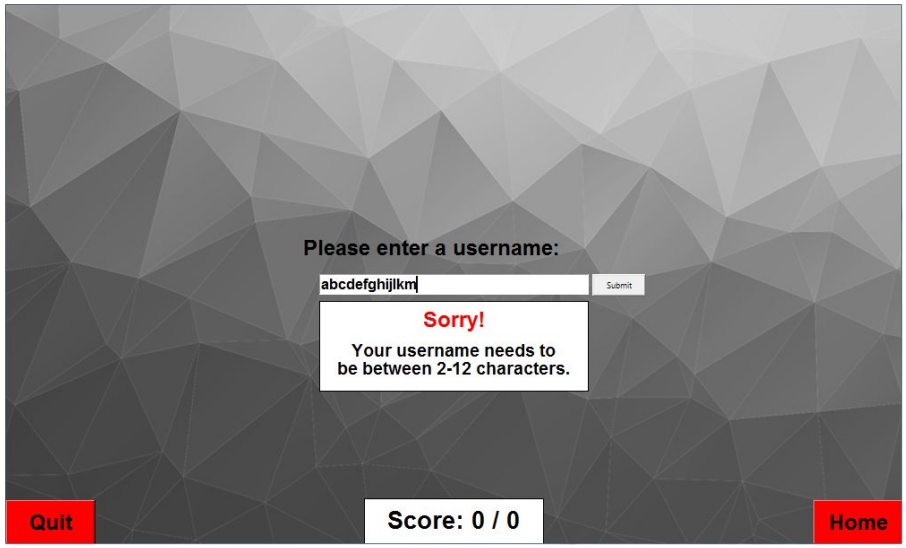
Version 3
Labelled: CdeBruyn_Excellence
I have changed the program to more pleasing visuals as well as an easier to read font. I have also implemented a "home button" which will allow the user to return to the homescreen, the score also has an added feature showing how many rounds the user has played. The home button I added to allow the user the chance to switch user without having to restart the game, This resets the score and names so that it doesn't carry on from the previous users game. The visuals and easier to read font is purely aesthetic and doesn't contribute to the game itself, the added amount of rounds played is to show the percentage of

Test Results

Comment screenshots of completed tests *(4A; 7M; 10E):*

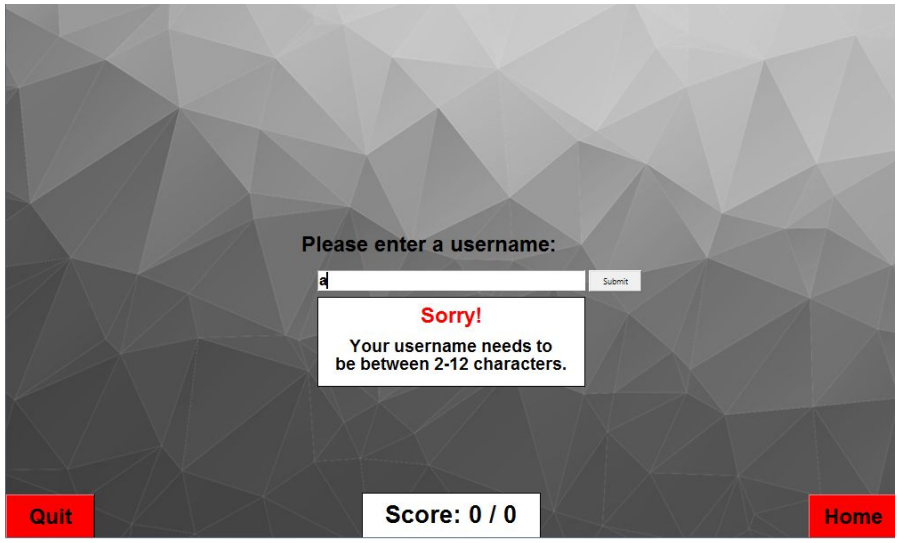
Testing if user can enter integers in name entry

The user was rejected from entering integers and an error message addresses the user what they typed wrong, No improvements needed

Testing if the user can enter more than 12 characters in name entry



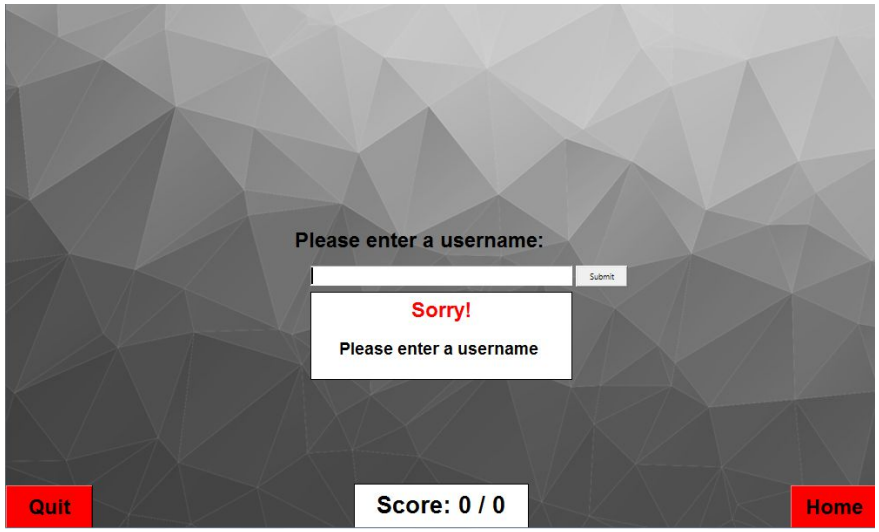
The user was rejected from entering more than 12 characters and an error message addresses the user what they typed wrong, No improvements needed

Testing if the user can type less than 2 characters in name entry



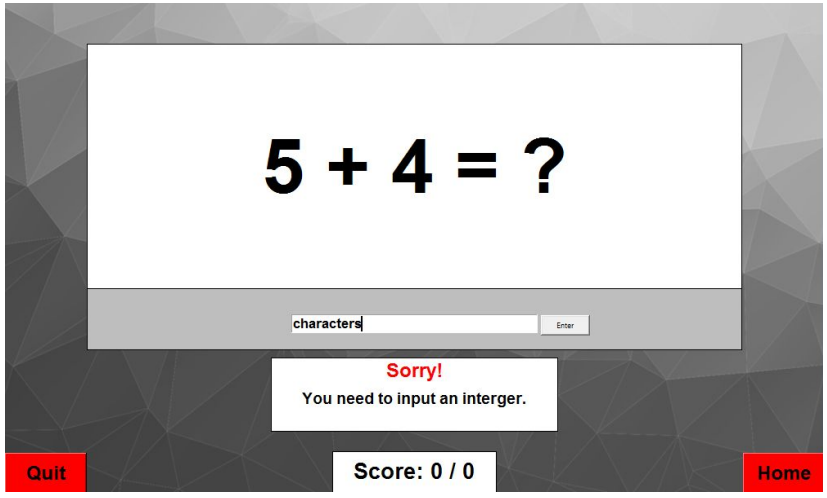
The user was rejected from entering less than 2 characters and an error message addresses the user what they typed wrong, No improvements needed

Testing if the user can continue without a username (Putting "" in the entry) in name entry



The user was rejected from not entering anything and an error message addresses the user what they typed wrong, No improvements needed

Testing if the user can enter characters in the game entry

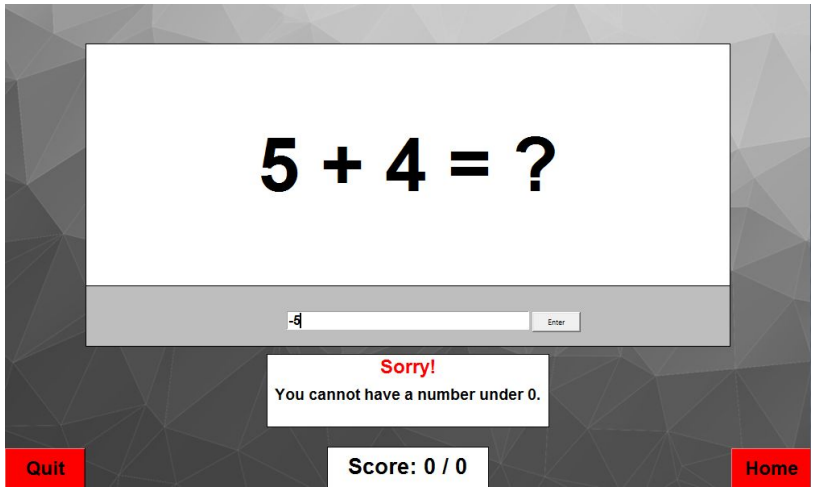


The user was rejected from entering characters and an error message addresses the user what they typed wrong, No improvements needed

Testing if the user can enter a number over 999 in the game entry

The user was rejected from entering a number over 9999 and an error message addresses the user what they typed wrong, No improvements needed

Testing if the user can enter a number less than 0 in the game entry



The user was rejected from entering a number less than 0 and an error message addresses the user what they typed wrong, No improvements needed