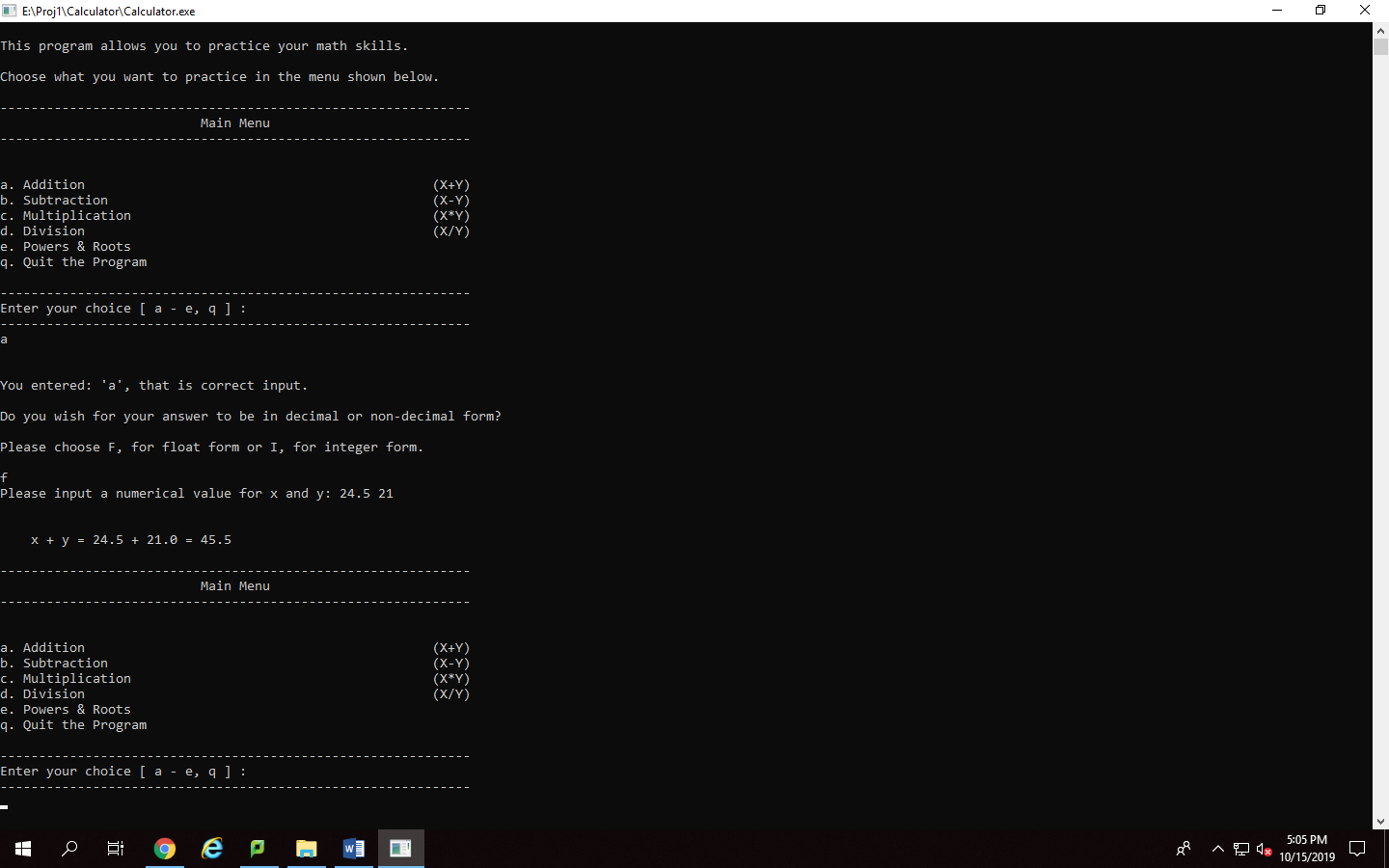
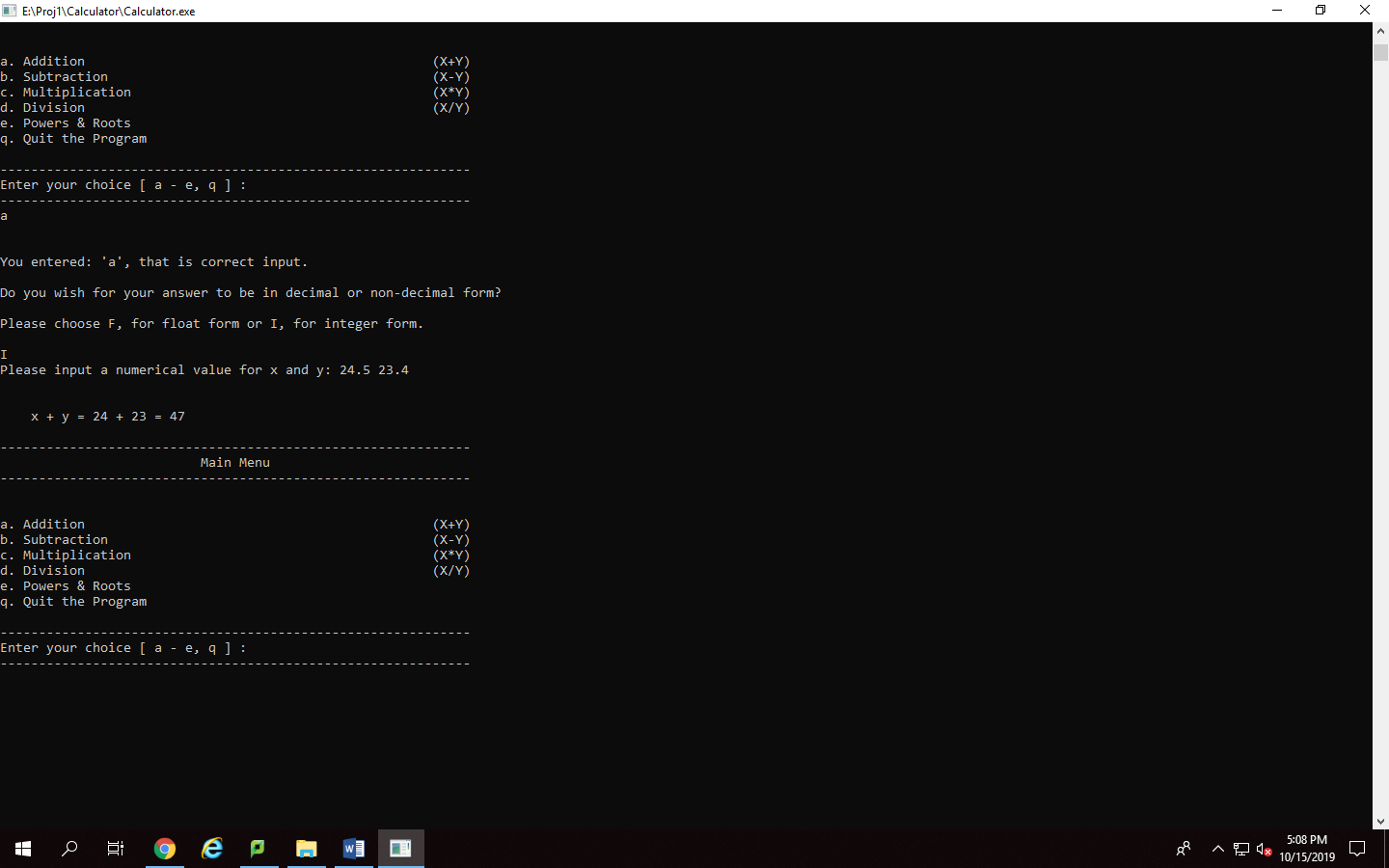
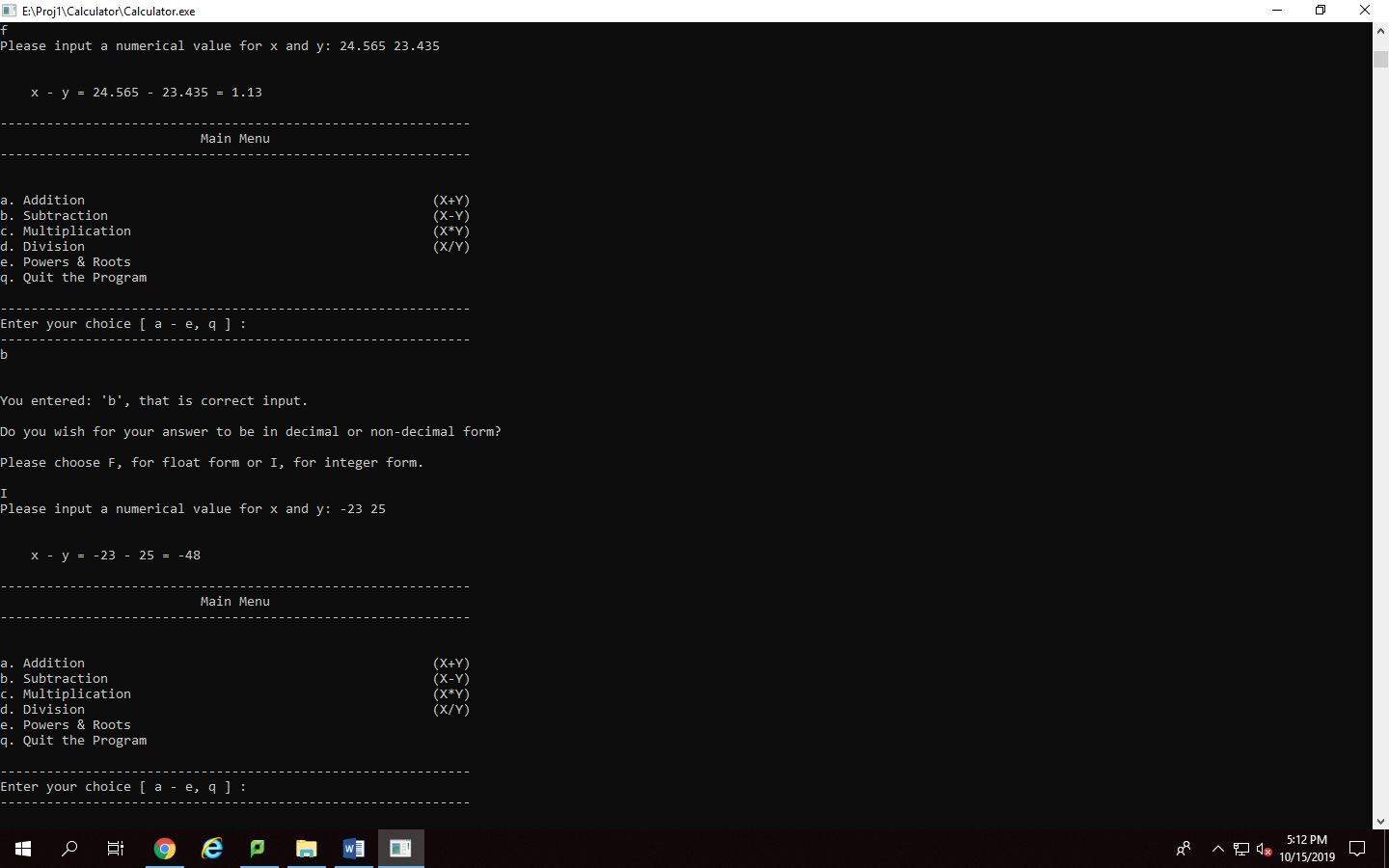
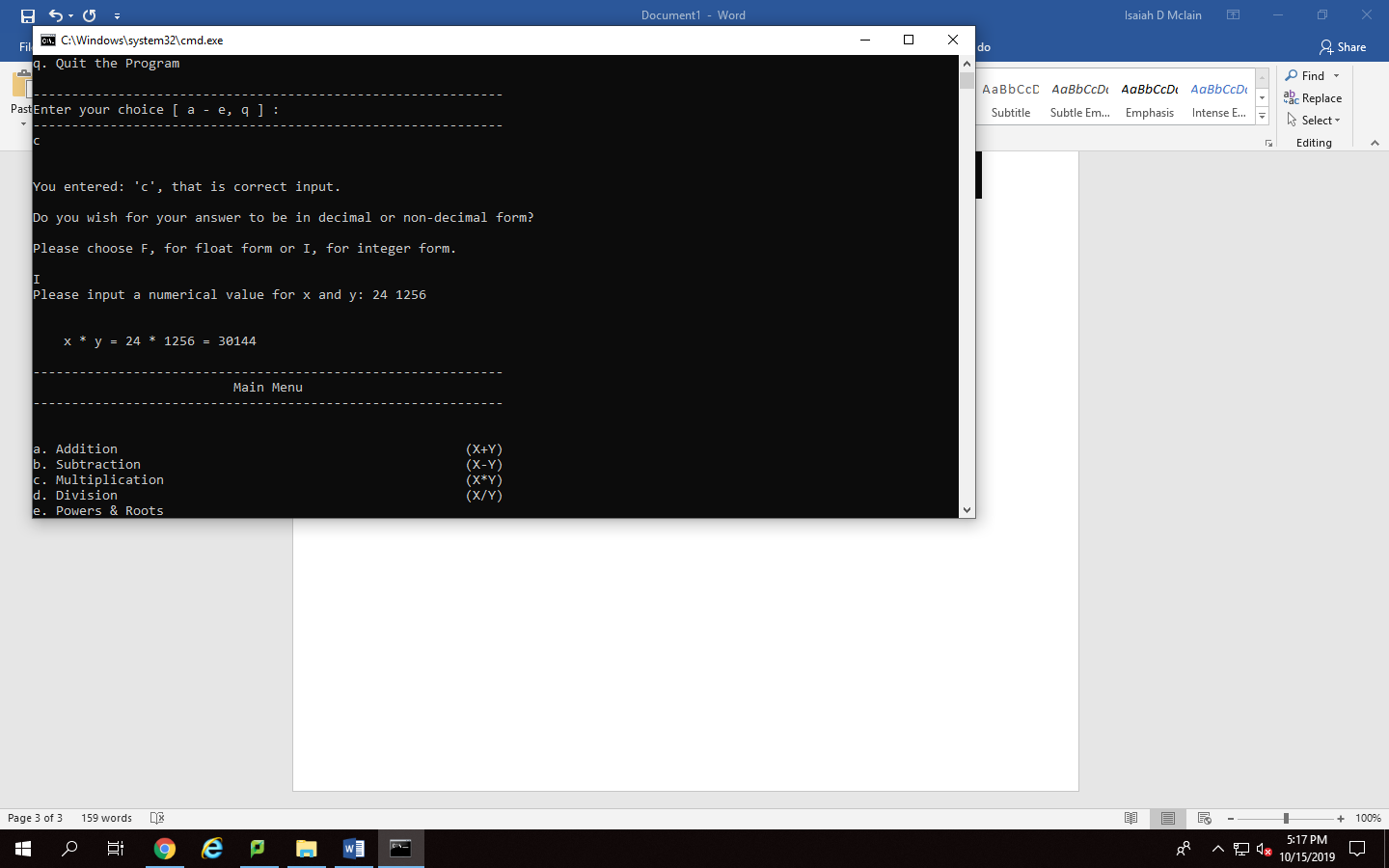
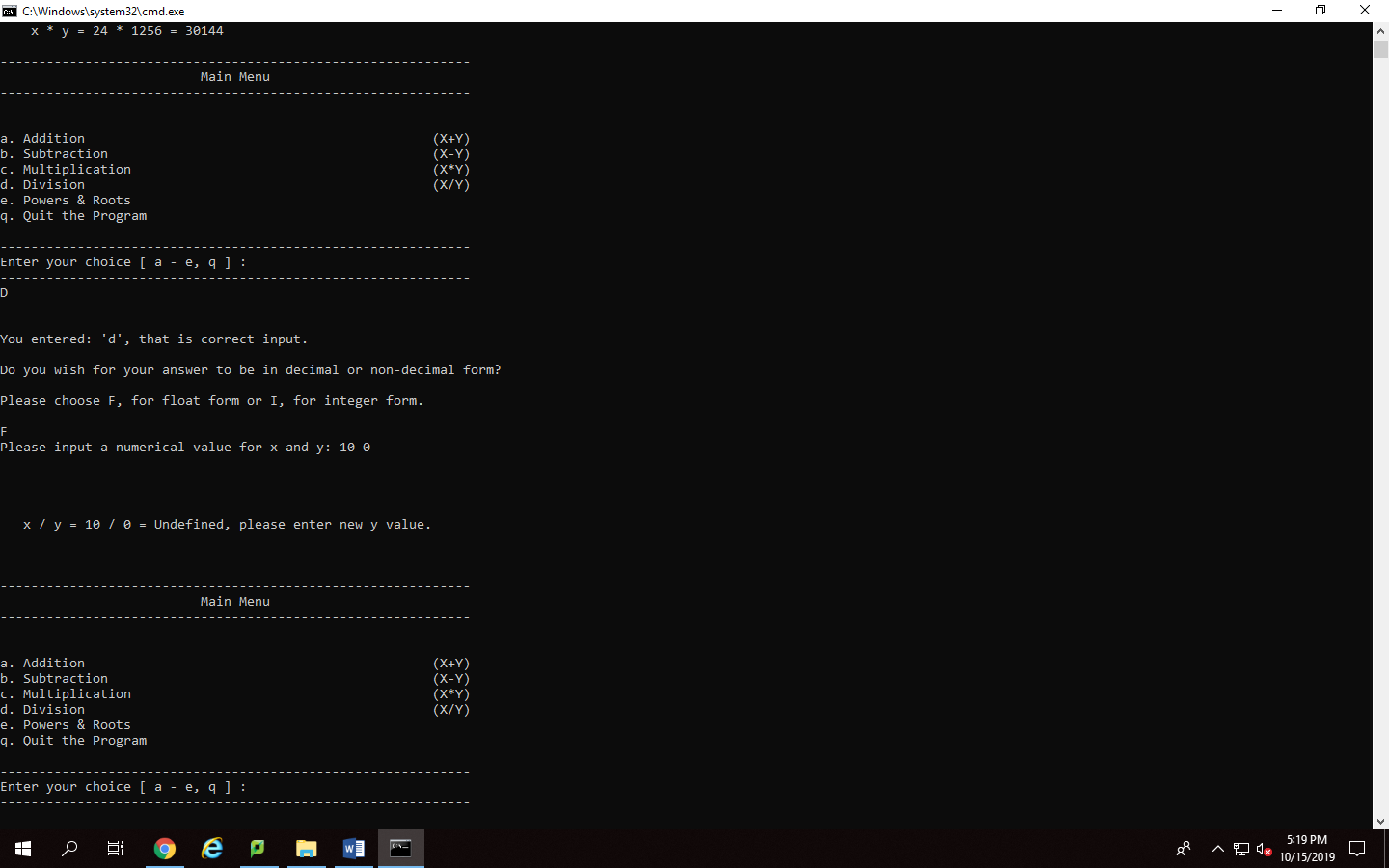
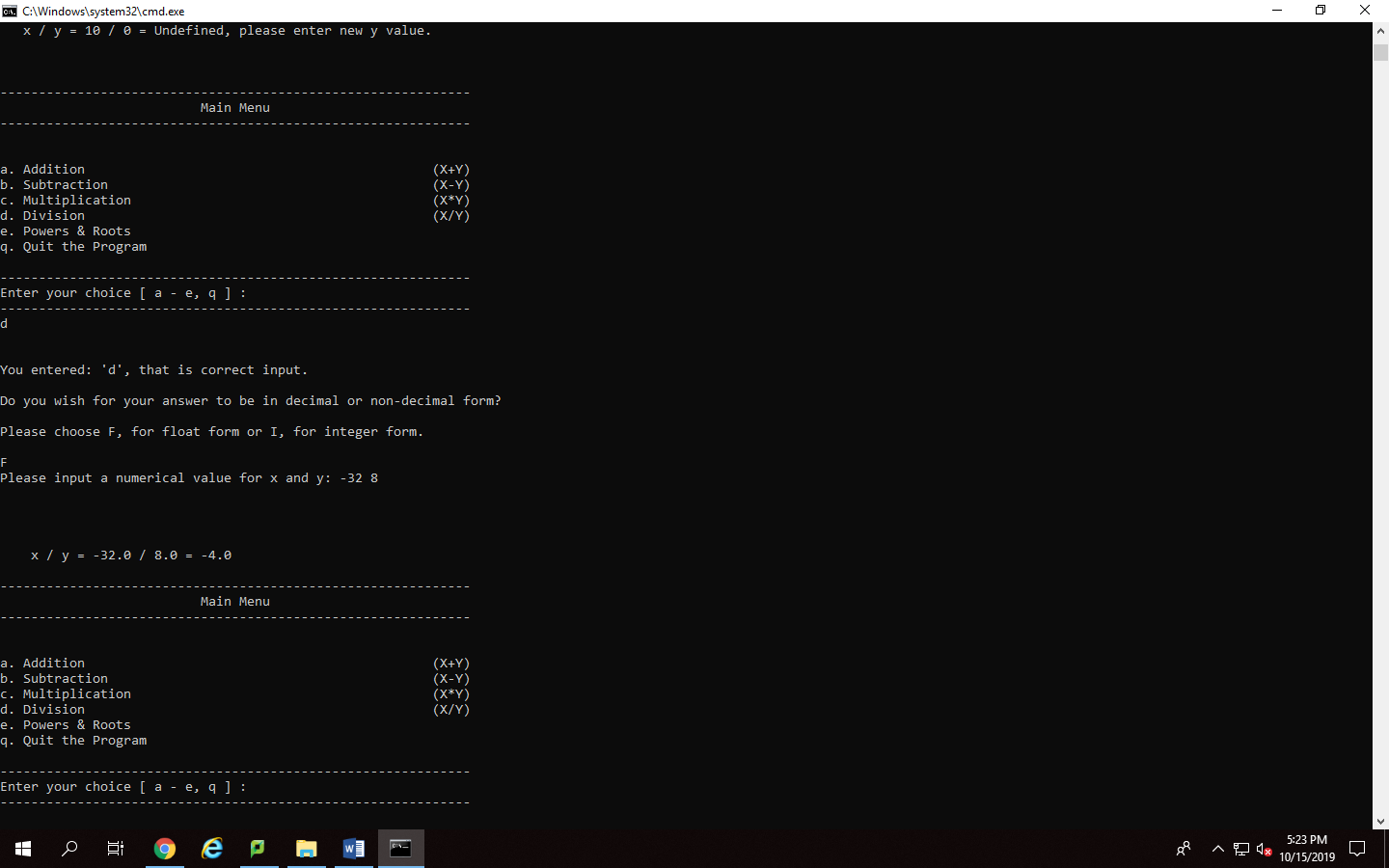
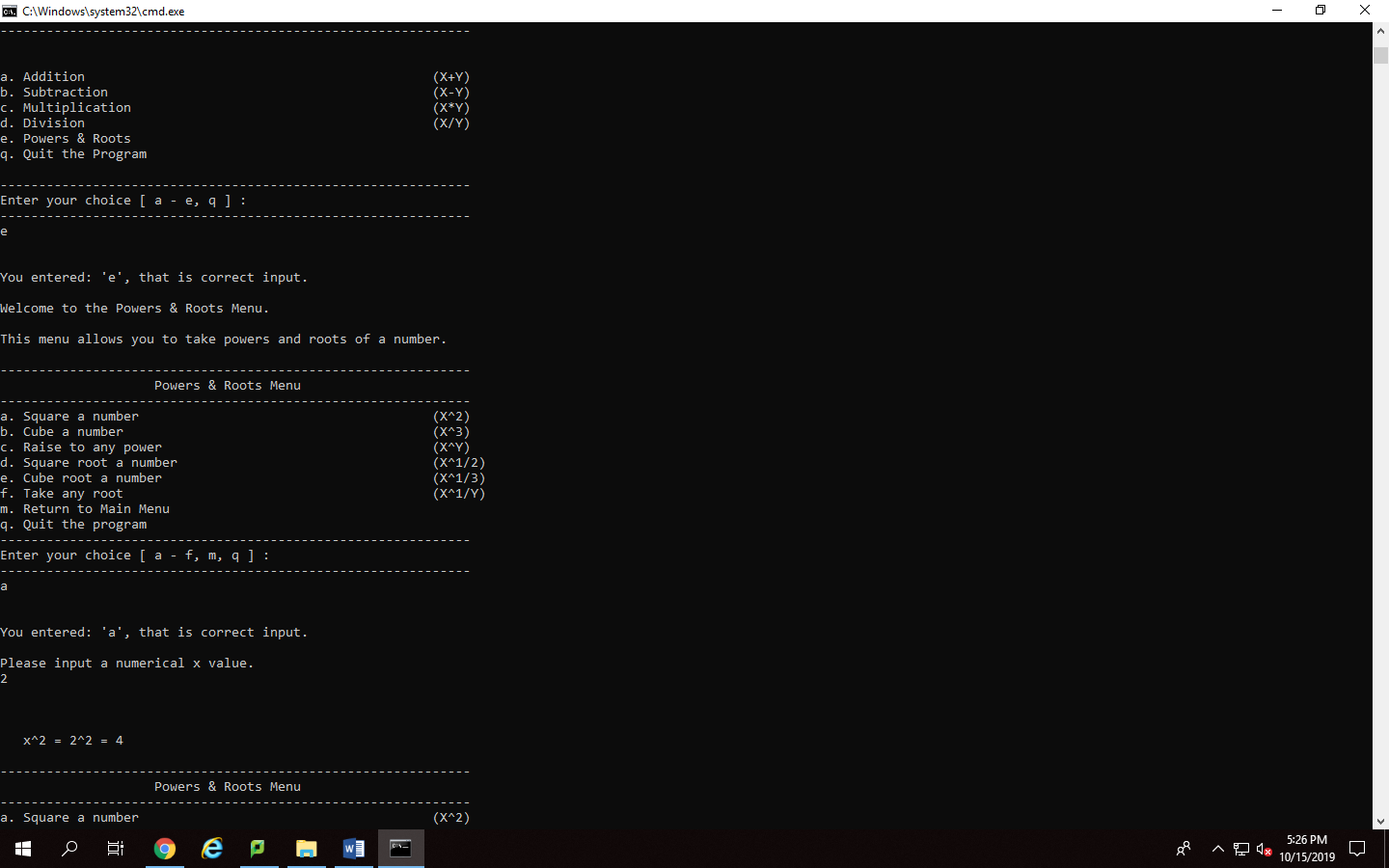
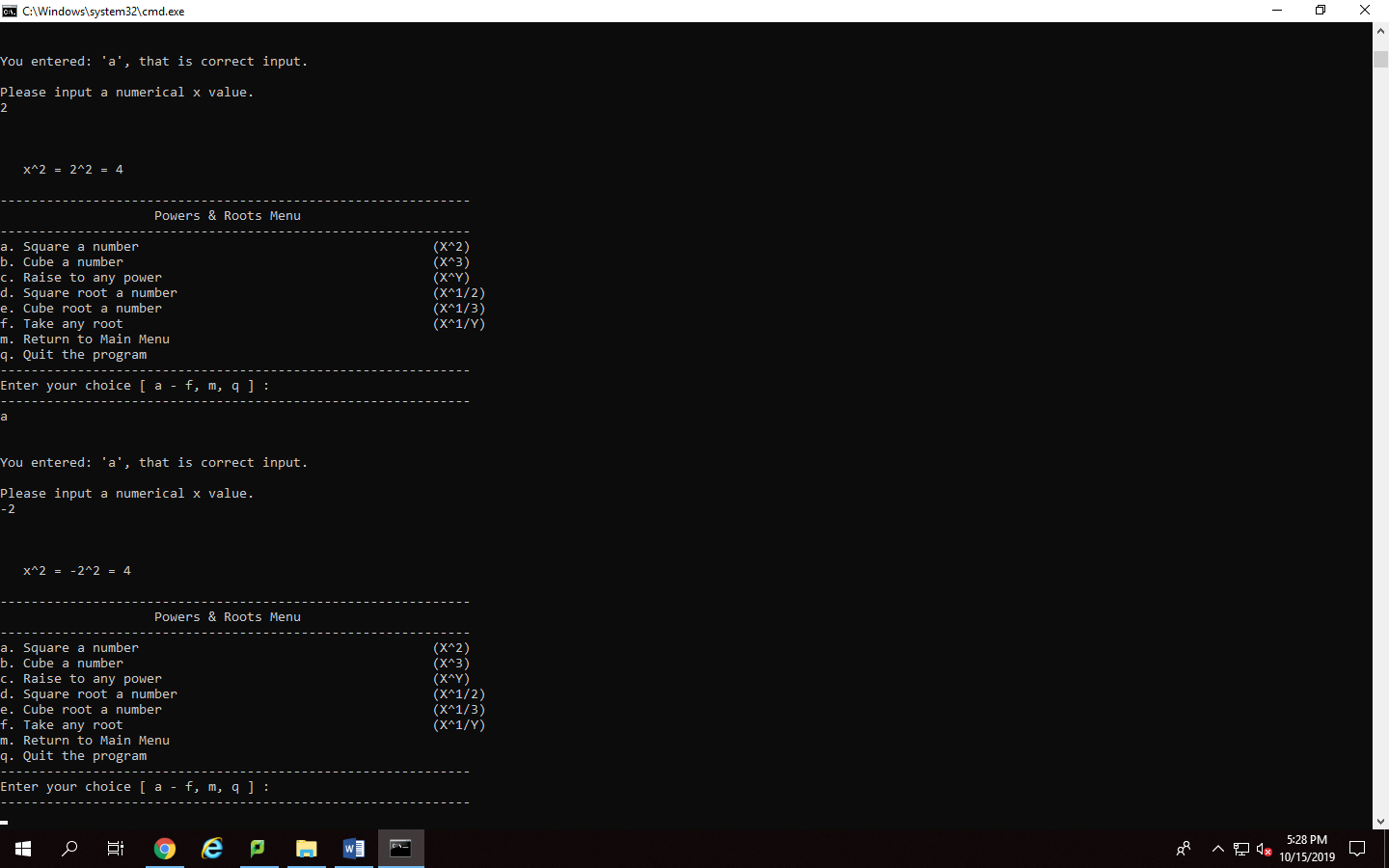
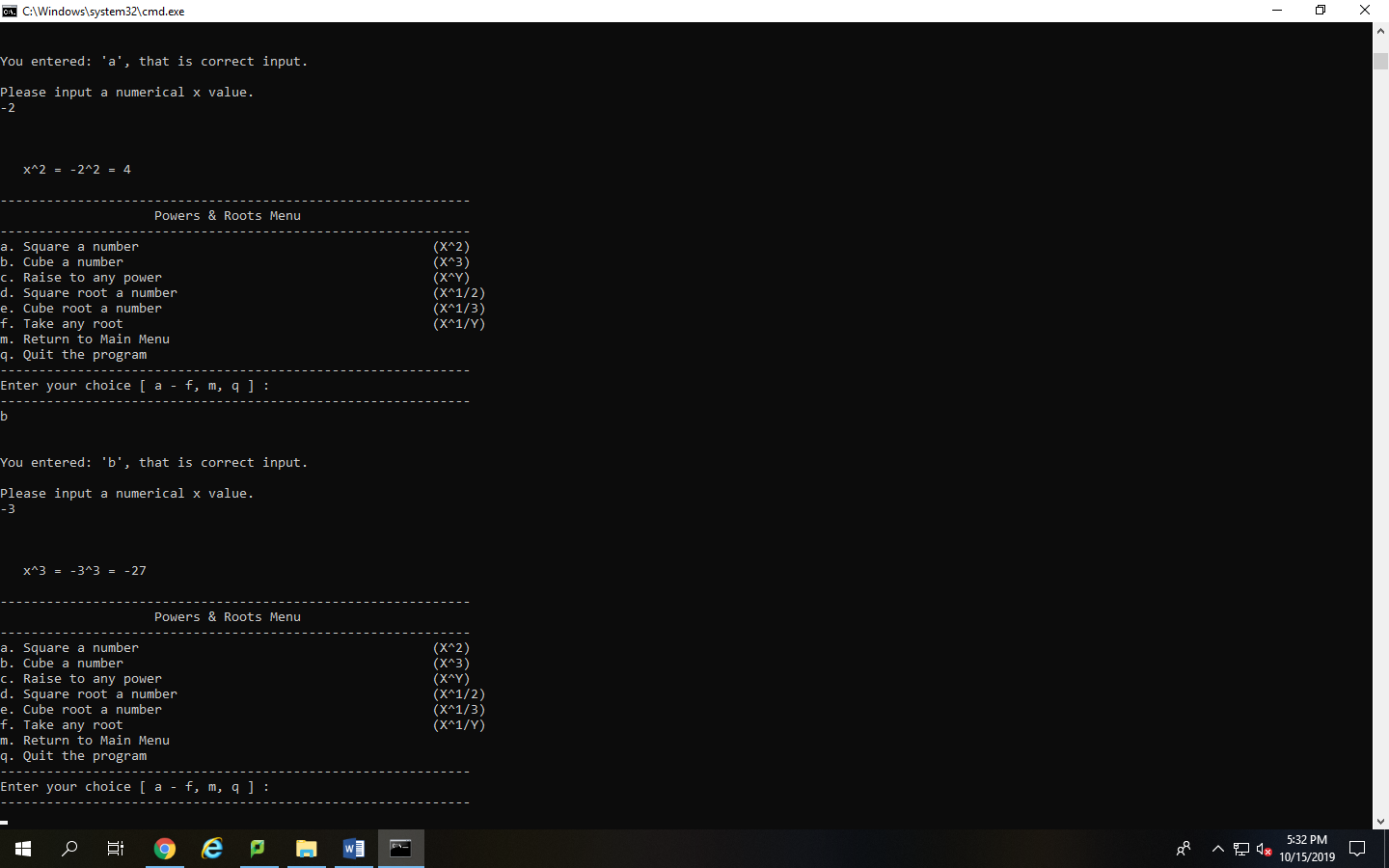
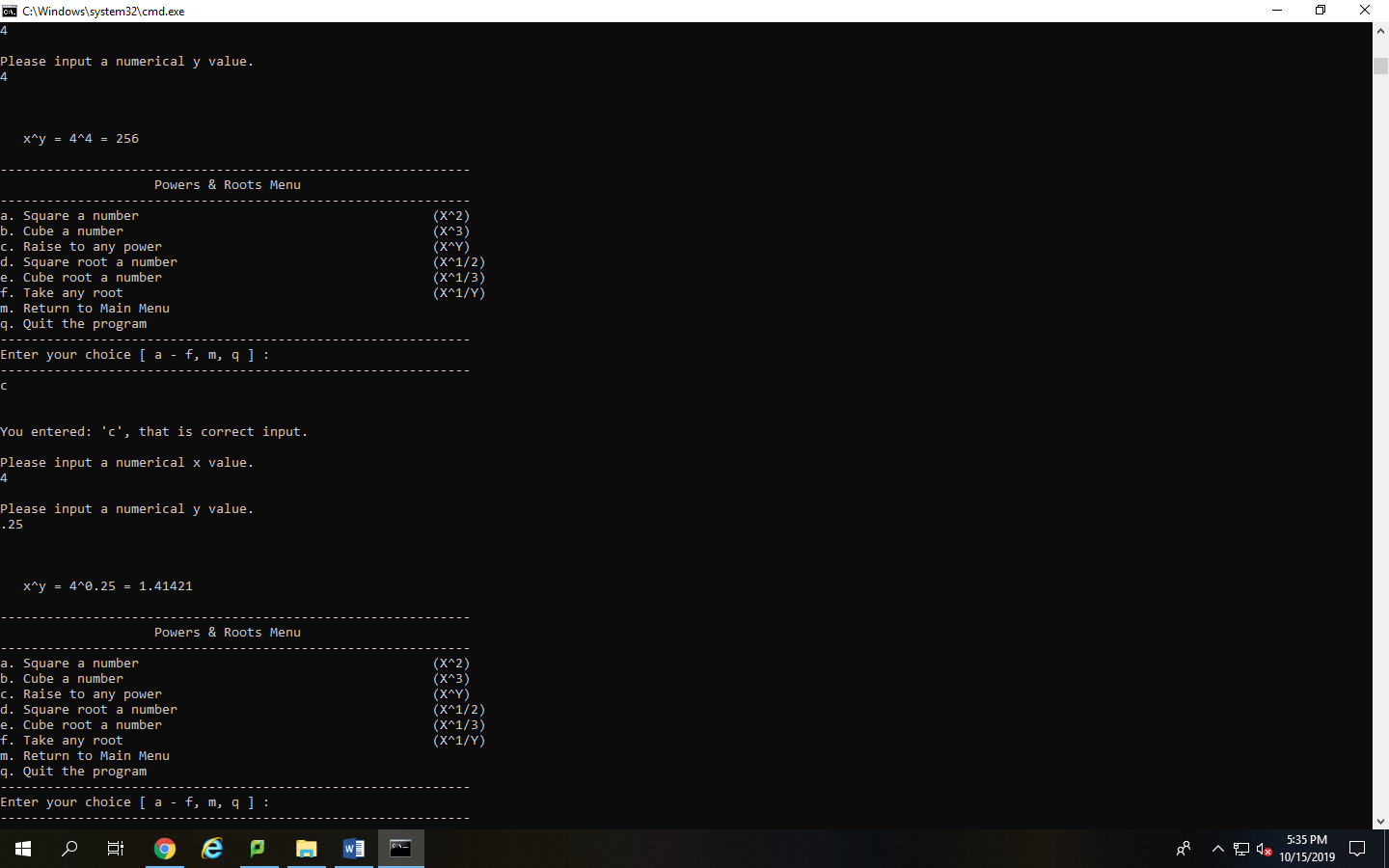
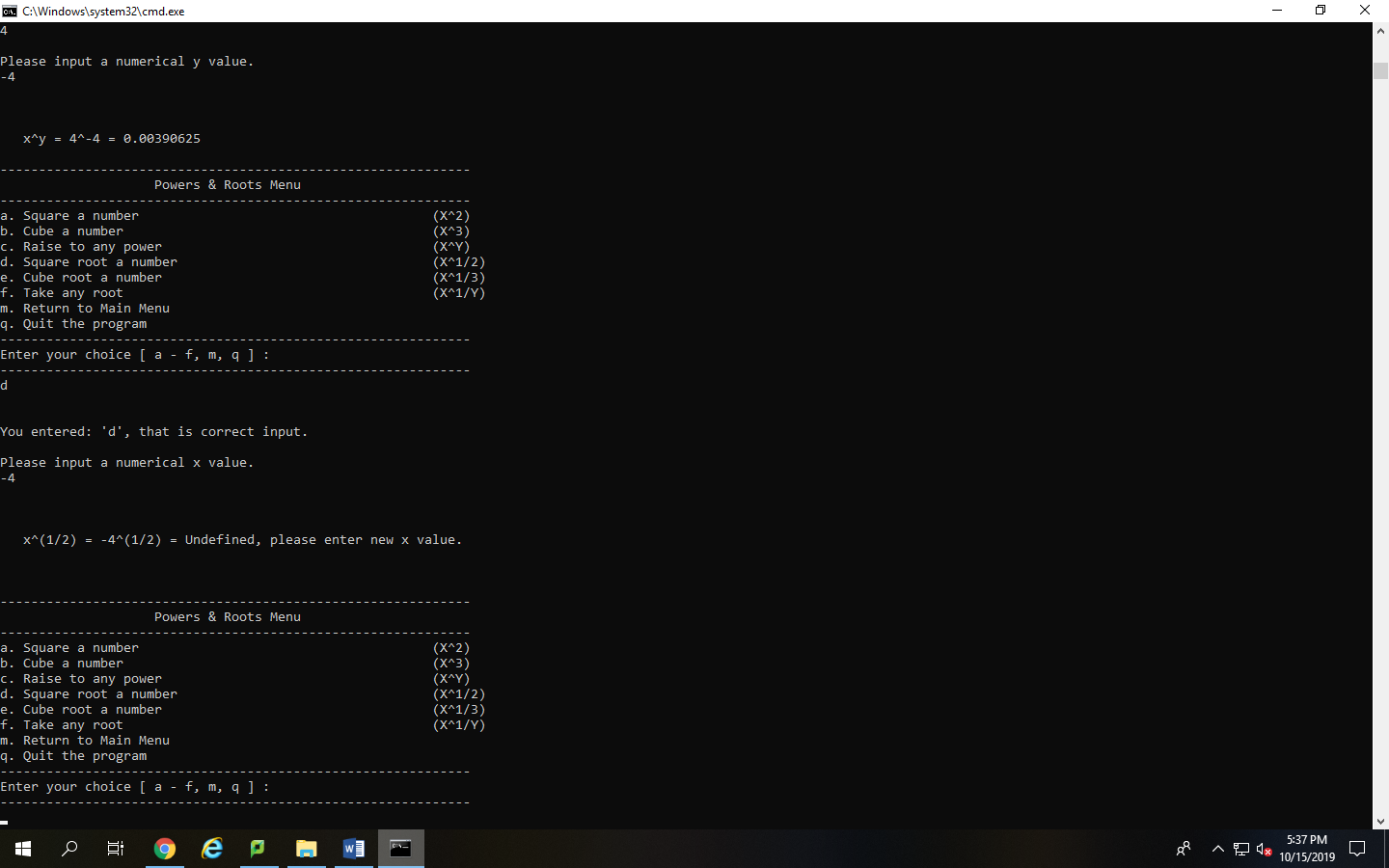
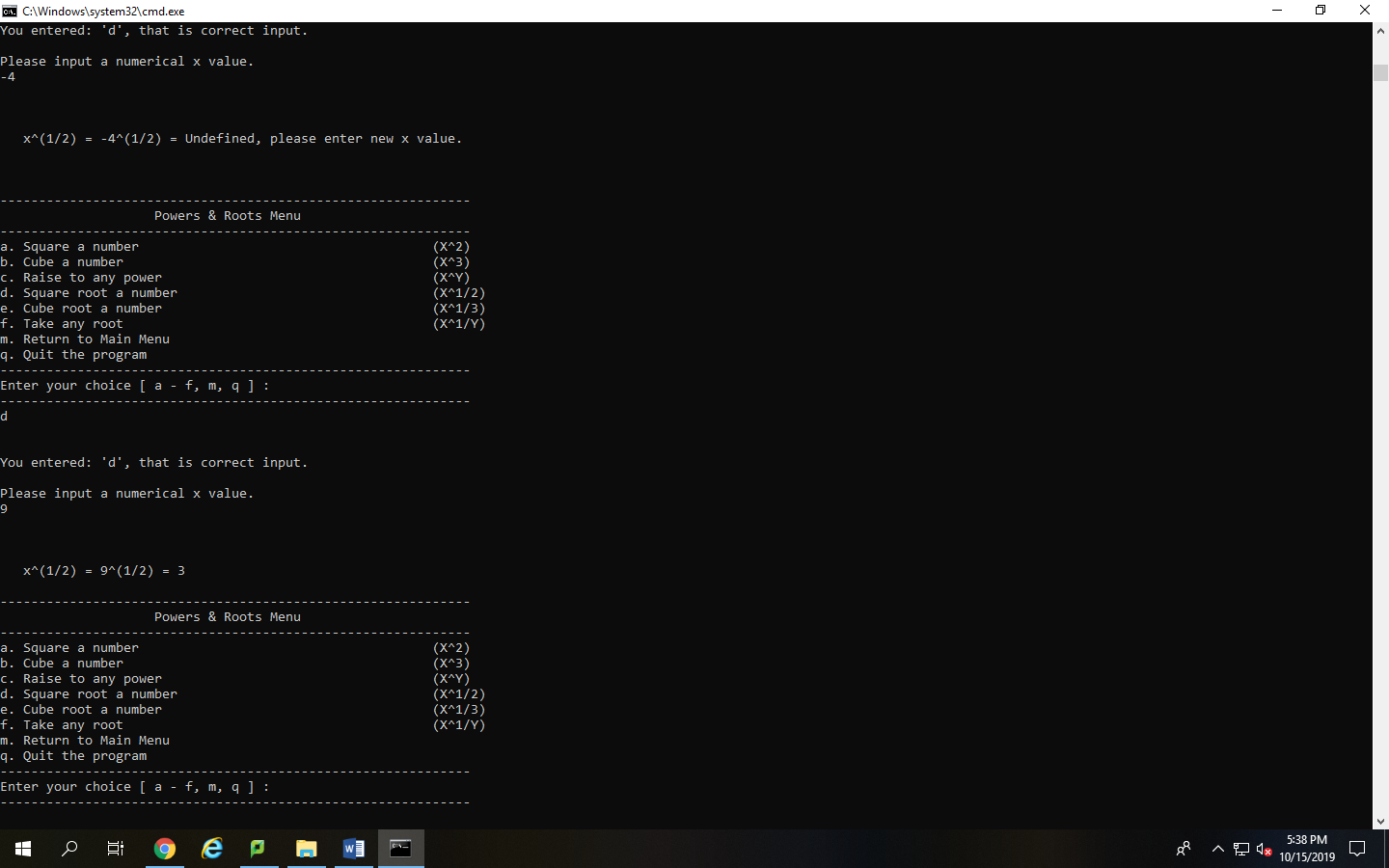
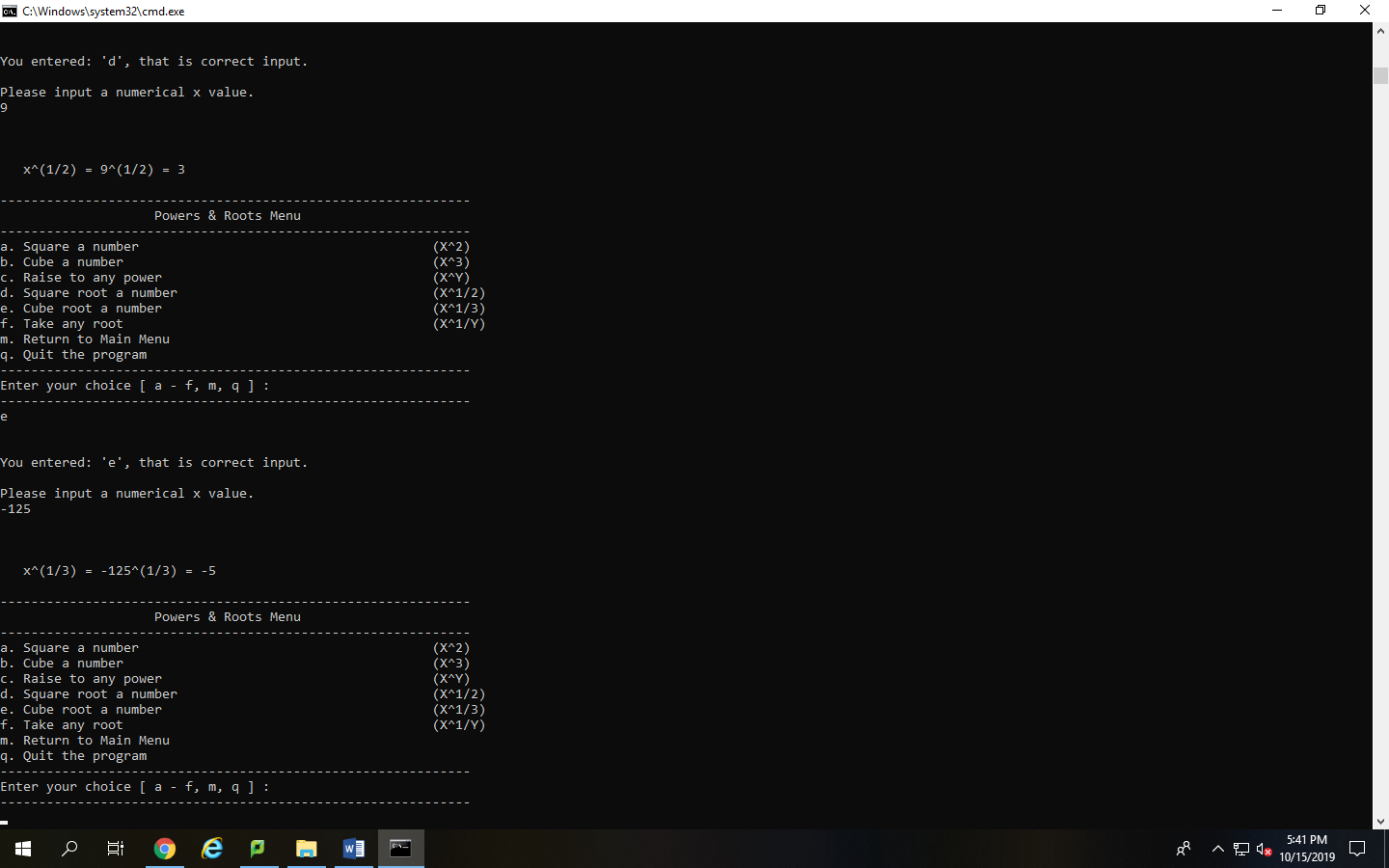
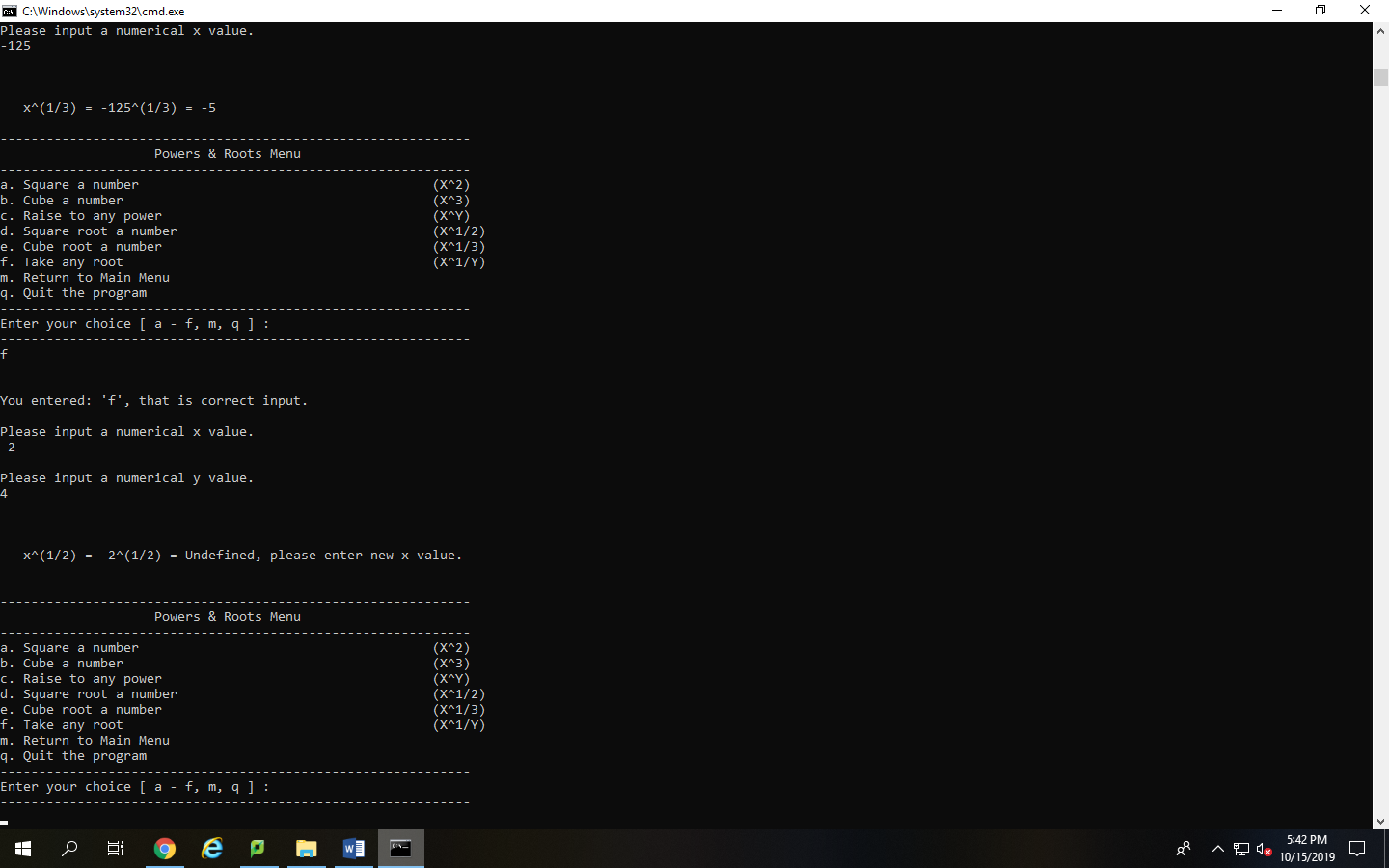
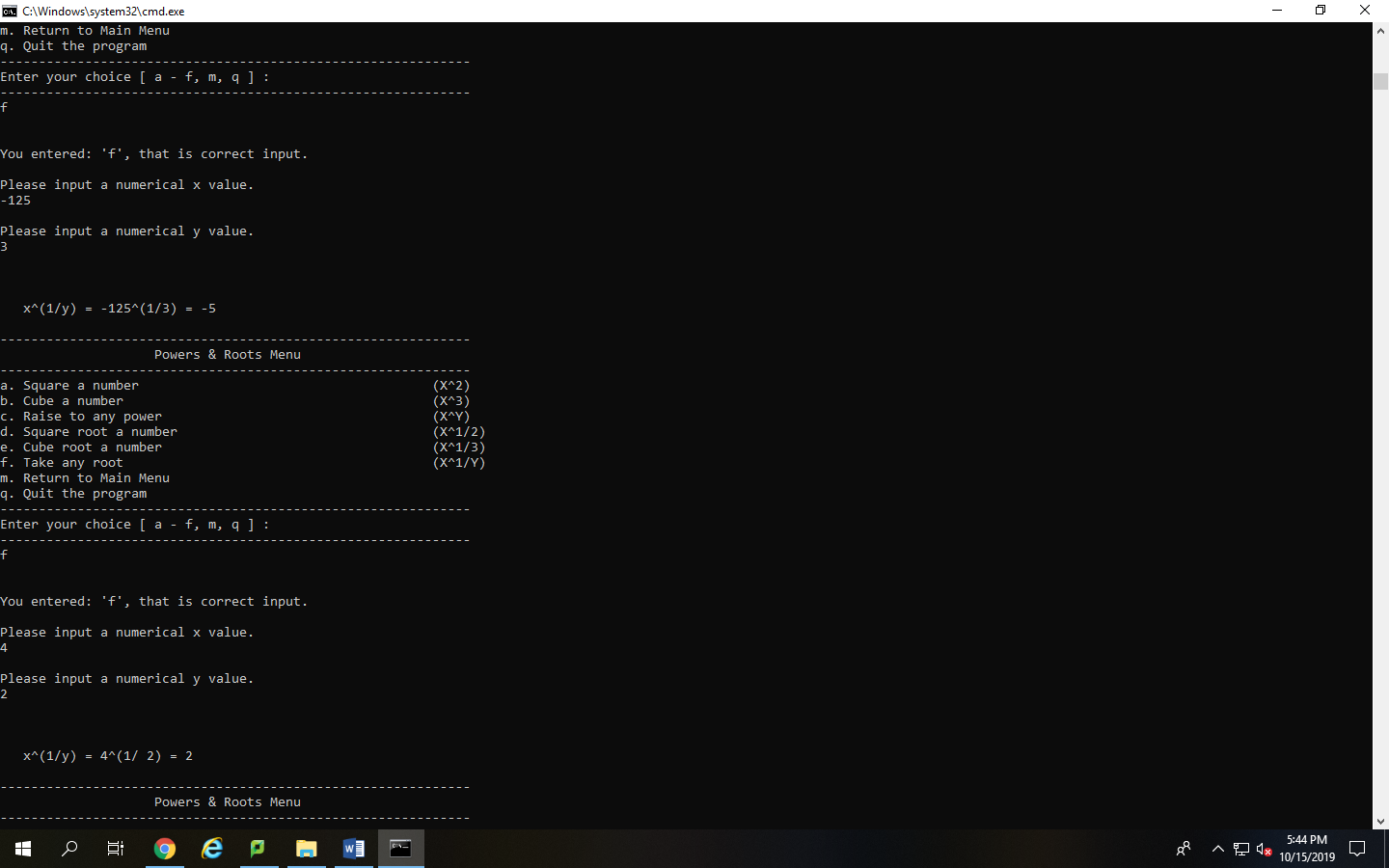
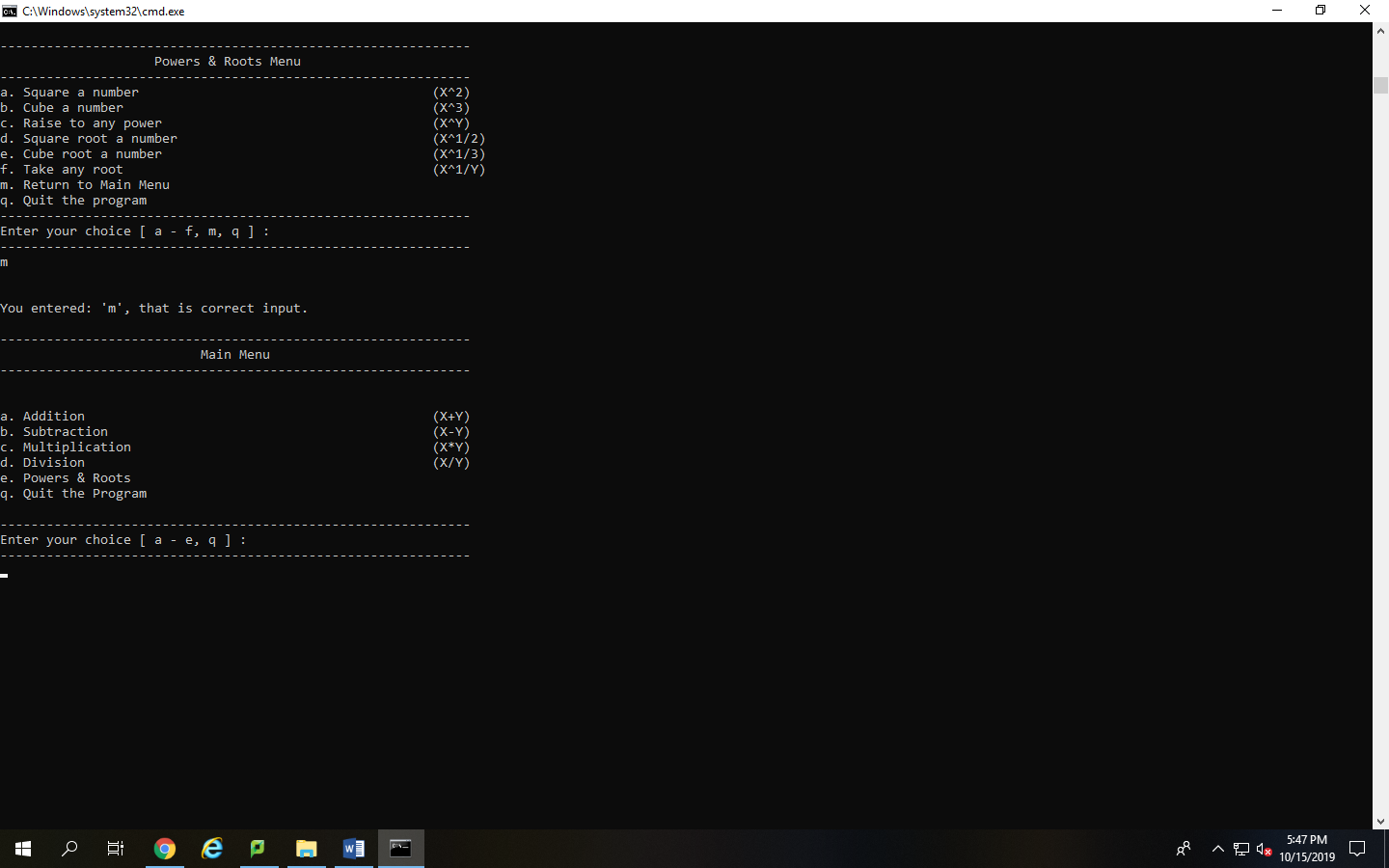
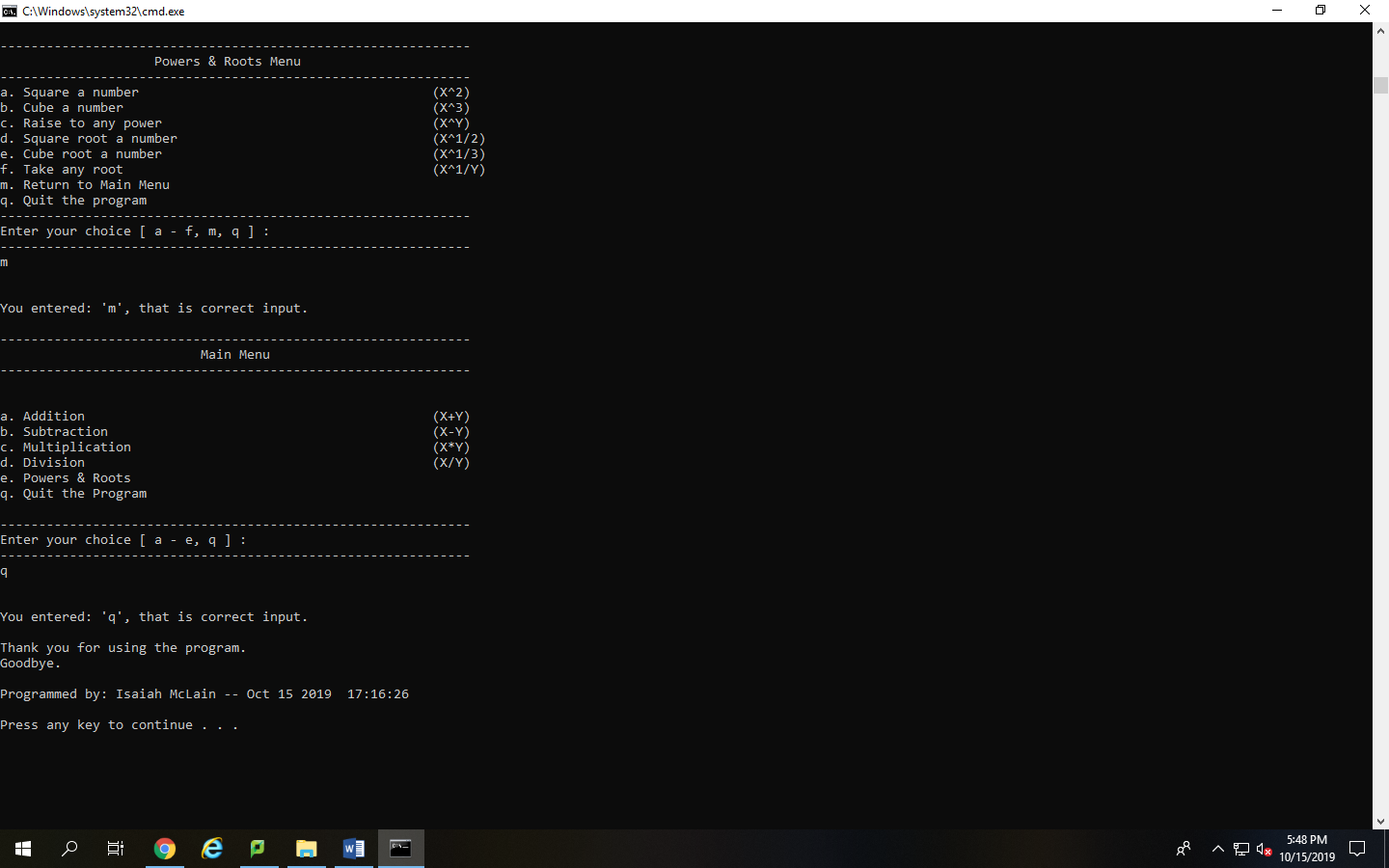
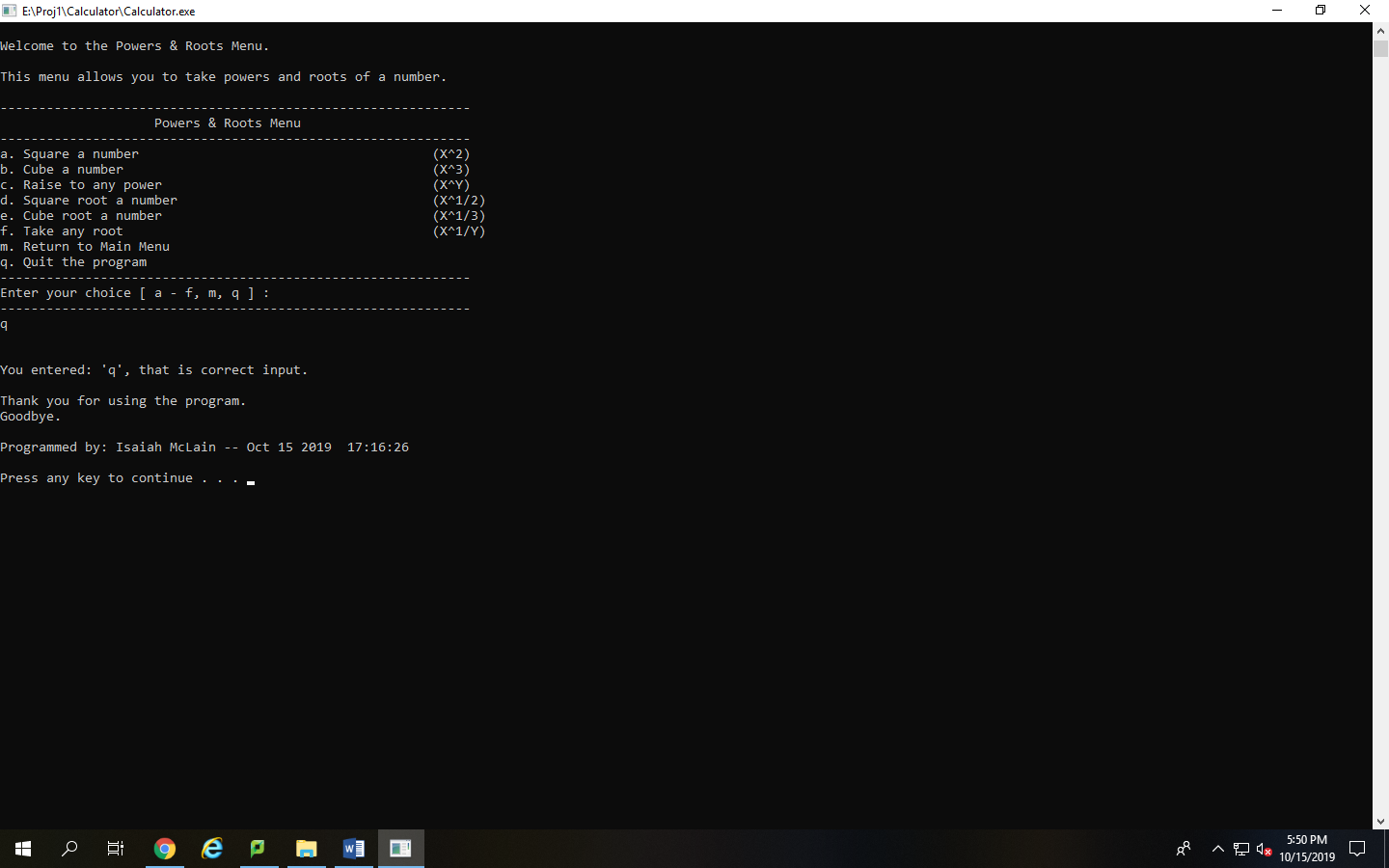
1. Displays that the program enables user to choose an item from the menu, in this case addition, and that they can choose floating point or integer form. I chose floating point form and added 24.5 and 21. The program is outputting 21.0, showing that I have show point manipulators in use. This helps show the user that even if they enter an integer value, they are still in float form.
2. Shows that if a user chooses an integer form for their answer, if they enter floating point values, then these values will be truncated. Also shows the other half of my project is working with the floating point and integer value choice.
3. This screenshot shows not only that my subtraction function is working but also that my program recognizes negative values and their properties as shown here. A user can enter -23 and the program understands that this value is negative and not as something else.
4. This screenshot shows that my multiplication function is working as it’s supposed to and that it handles numbers of different sizes and outputs the correct format. Showing that 24 times 1256 is equal to 30144
5. Screenshot shows that my program can handle the issue of division by zero but that it also will display a proper answer of Undefined and tells the user which value needs to be changed, in this case its y. It will not crash the program or output “nan.”
6. Screenshot shows that my division function works with normal numbers and that it can also handle division with negative numbers. Is another showcase of my show point manipulator at work letting the user know that they are in floating point form by attaching a “.0” to the end of the numbers.
7. Screenshot shows that the user can access the power and roots menu through the main menu and that they can choose a function off of the power and roots menu, in this case square a number, and perform a correct calculation with it.
8. Displays that the second menu loop function works and will leave the user in the power and roots menu unless they choose to leave it even after doing a calculation.
9. Screenshot displays that the cube any number function is working and also recognizes that a negative cubed is still a negative answer. This case -3 to the power of 3 is -27
10. Screenshot displays that the user can raise a number to any number they want. This includes decimals and normal integer values. In this case the user choose 4 to the power of 4 and 4 to the power of .25
11. Shows that the square root function understands that you cannot square root a negative value and gives the correct answer for a 7th grade math user, which is undefined.
12. Displays that the square root function works normally as well. Square rooting 9 to display the answer of 3.
13. Displays that the program can due a cube root and that the cube root function recognizes that the cube of a negative is a negative answer.
14. Shows that the root of any number function has the proper logic to recognize that a root of a negative number by an even root is undefined.
15. Displays that the take any root function can handle the issue of rooting a negative number by a cube and getting the correct answer of a negative number and that the function also recognizes that positive numbers rooted by an even number is a correct answer not undefined. This shows that the take any root function can handle the special cases that it may encounter.
16. Displays that the user can exit the power and roots menu/loop and return to the main menu properly.
17. Displays that the user can exit the program properly with the proper documentation at the end with programmer name and compile date.
18. Displays that the user can also exit the program through the power and roots menu as well without any program issues and with the proper exit documentation including timestamp and programmer name.