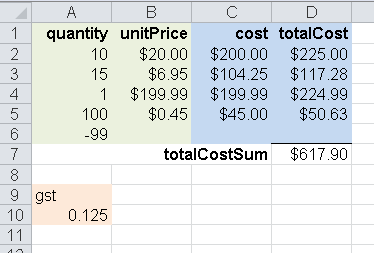
**Documentation of plan to code with testing and debugging**

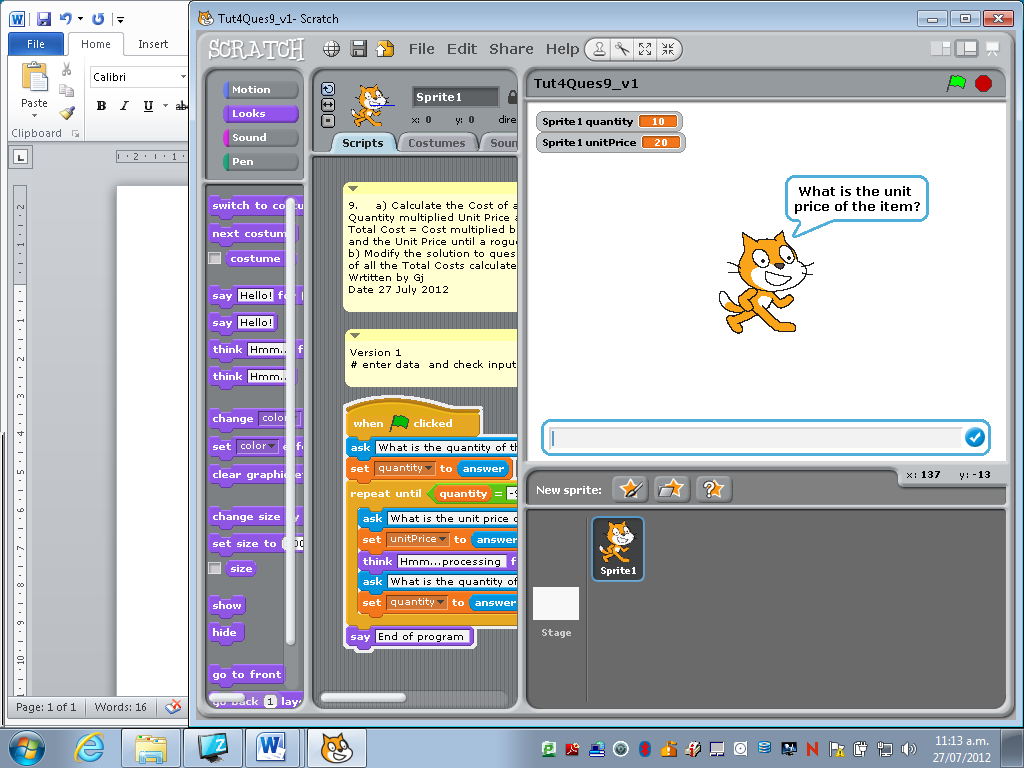
**Tutorial 4 Question 9**

**Test data**



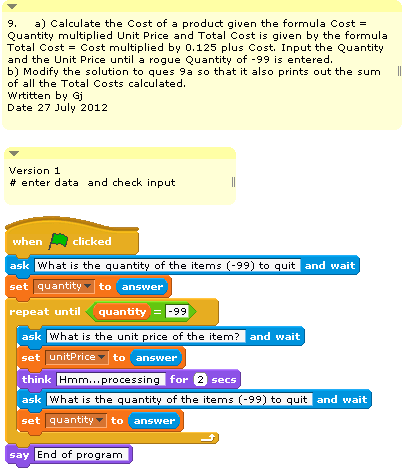
**Version 1**

Output:



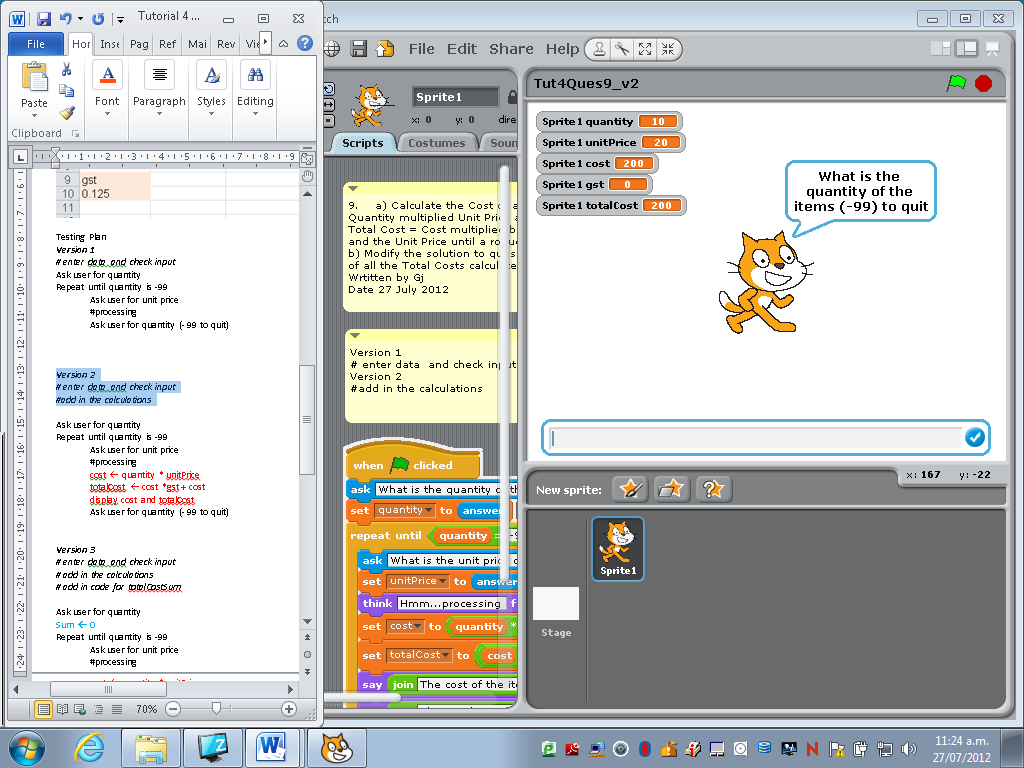
Program accepts input correctly and stops with loop variable of -99

Code for Version 1:



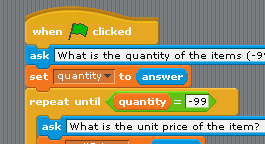
**Version 2**

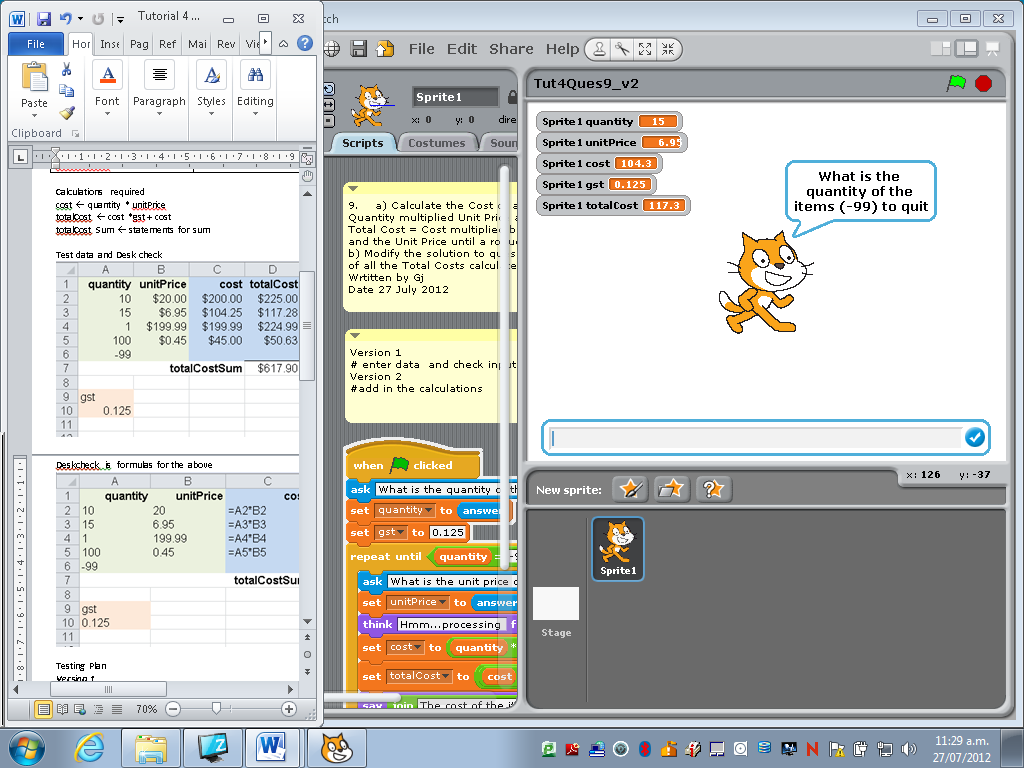
Output



Output for total cost is incorrect for first item

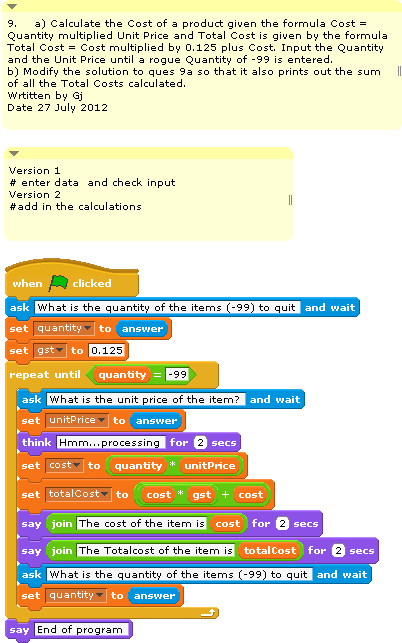
Gst value was not set at beginning of program





Output is now correct - values for variables show correct values for test data input

Code for version 2



**Version 3**

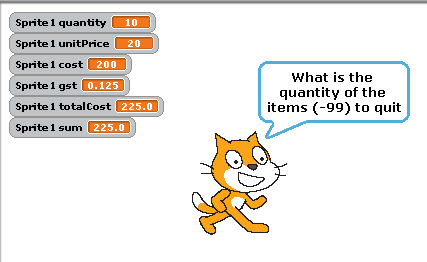
Output

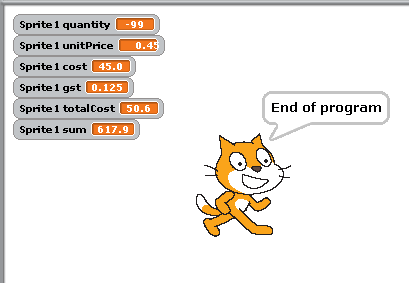


Output for sum is incorrect

Found wrong calculation for sum - should be add not multiply

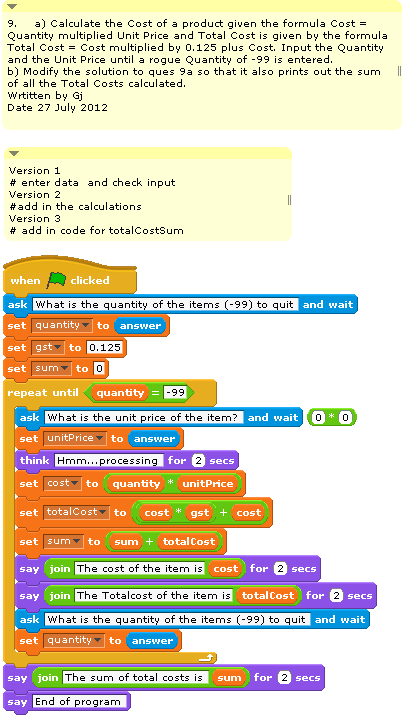






Program now gives correct output as shown by test data in above screen

**Code for version 3:**



Version 4 - add in code for data validation

**Test data for unexpected input**

|  |  |  |  |
| --- | --- | --- | --- |
| Quantity | Unit Price | Outcome | Testing for |
| -99 |  | No data entered | No data and therefore no runtime errors (division by zero for example) |
| 0 | 2.50 | No cost | Boundary values for quantity |
| 2 | 0 | No cost | Boundary values for unit price |
| 6.3 |  | error | Integer value for quantity |
| 2 | -6.5 | error | Out of range value for unit price - cost to be positive |
| -10 |  | error | Out of range value for unit price - will have to change loop variable as can’t then use -99 to stop |
| abc |  | error | Can’t have text for a number ( may code as zero in Scratch) |

Grey highlighted testing and coding would give **Merit** - have tested for Unexpected data of boundary values and no data at all

Rest of testing (at least two examples) with relevant coding would give **Excellence** as testing for unexpected data that is invalid data

For Excellence would also then expect to not have variable values displayed on the screen for the user