University of Stirling
Computing Science and Mathematics
CSCU9A2 Tutorial 3 – Sample solutions Spring 2017
(for week starting 13 February)

1. Here is a possible set of variables to hold the internal "state" of the document and editor: (by no means the only solution)

   - `text`: A string to hold the characters (no font, style or cursor – just plain text).
   - `cursorPos`: An int giving the index within text of the character preceding the cursor (so `-1` if cursor is at the start, and `length-1` if at end).

   Events that the program should respond to, and appropriate reactions: Just general ideas, since we do not know exactly what events our run time system might report, nor exactly what the text display widget can do (the point is to think about internal data representation, events, data updates, redisplay):

   - *Program launch/initial document load:*
     Read whole document into `text`, set `cursorPos` to `-1`, instruct text area display to show whole `text`, with cursor symbol before first character.
   - *Backspace key:*
     If `cursorPos` is `-1`: No action. Otherwise remove from `text` the character whose index is `cursorPos`, set `cursorPos--`, instruct text display to show whole `text`, with cursor symbol positioned after character at index `cursorPos` (at start if it's `-1`).
   - *Delete key:*
     If `cursorPos` is `length-1`: No action. Otherwise remove from `text` the character whose index is `cursorPos+1`, instruct text display to show whole `text`, with cursor symbol positioned after character at index `cursorPos` (at start if it's `-1`). [No need to adjust the value of `cursor` in this case.]
   - *Ordinary key press:*
     Insert new character into `text` after the one at position `cursorPos` (position 0 if `cursorPos` is `-1`), cursorPos++, instruct text display to show whole `text`, with cursor symbol positioned after character at index `cursorPos` (at start if it's `-1`).
   - *Mouse click:*
     Assume that the supplied event information (parameter of event handler), gives us the position at which the mouse was clicked: Set `cursorPos` to the new position. Instruct text display to show whole `text`, with cursor symbol positioned after character at index `cursorPos` (at start if it's `-1`).

2.
   (a) Black box tests are tests derived from the *specification*. For each test, the specification indicates what the actual result should be.

   A good set of tests would cover *typical* values from each of the age ranges, and values around the *boundaries* between the ranges – we get the ranges and boundary values from the specification. For example 3, 5, 6, 7, 10, 15, 16, 17, 18, 19, 30.

   (b) White box tests are tests derived from the *structure of the code itself* – for example to exercise each alternative in each **if** statement. Again, for each test, the specification indicates what the actual result should be.

   A good set of tests would exercise each **if** test in its true and false branches and also "boundary" values where the **if** test result switches from true to false. For example: 10, 15, 16, 17, 30 (the "marry" **if**), 10, 17, 18, 19, 30 (the "vote" **if**), and 3, 5, 6, 7, 10, 15, 16, 17, 30 (the "bus travel" **ifs**). (There are clearly overlaps between the sets, and each test would only be carried out once!)

   (c) The code gives the wrong bus rate travel for 16 year olds (should be adult rate from the spec, but the code gives junior rate) – so test this value would expose the fault. If the set of tests had not included this boundary value then the fault would not have been exposed.

SBJ February 2017