Five unusual things to improve feedback that you could try in your teaching

Steve Draper, David Nicol. 9 Dec. 2009
http://www.reap.ac.uk/
http://www.psy.gla.ac.uk/~steve/

This is the handout for a talk, see: http://www.psy.gla.ac.uk/~steve/talks/dundee2.html

1) Course team publishes in advance the year's timetable of feedback. See lower half of this page.

2) Comment bank plus magic rubric. See page 2 of this handout.
Magic rubrics: see http://www.psy.gla.ac.uk/~steve/localed/dweck.html

3) One minute papers. http://www.psy.gla.ac.uk/~steve/resources/tactics/minute.html

4) Student authored MCQs, complete with reasons for each response option.
This is discussed, with several references to published papers, in:
Draper,S.W. (2009) "Catalytic assessment: understanding how MCQs and EVS can foster deep learning" British Journal of Educational Technology vol.40 no.2 pp.285-293
Software (free) for administering student MCQ authoring in large classes: http://peerwise.cs.auckland.ac.nz/

5) Reciprocal Peer Critiquing.
http://www.psy.gla.ac.uk/~steve/localed/#mcr (material on Reciprocal Peer Critiquing)

6) Jaye Richard's use of EVS questions, not to test knowledge, but to motivate and interest in each new course topic; and to act as a specification for Enquiry Based Learning by pupils of the topic.
http://mimanifesto.wordpress.com/

Calendar of feedback for a course

Assignment	**Handin**	**Feedback ready**	**Feedback Type**
Essay	12 Jan	24 Jan	Written, personal
Essay 2	-	Weekly tutorial	Oral, feedforward advice
Essay 2	3 March	15 March	Written, mark, feedback
Term essay	1 April	Mid-June examiners' meeting	Mark only: summative
Lab report	14 Feb	28 Feb	Pro-forma mark and feedback sheet
Critical review	-	Weekly tutorial	Oral, feedforward advice
Critical review	1 March	3 March	Peer formative feedback
Critical review	7 March	15 March	Written and oral feedback, in group

Sample of a comment bank sheet plus magic rubric. This is by Quintin Cutts, and shows part of the first of 2 pages that made up the complete feedback sheet:

Computing Science 1P

Feedback sheet for Units 4/5

Look at items: 1 2 3 4 5 6 7 8 9 10

Remember, learning to program can take a surprising amount of time & effort – students may get there at different rates, but almost all students who put in the time & effort get there eventually. Making good use of the feedback on this sheet is an essential part of this process.

Acceptance test input

Unit 4: checker.py

input1.txt – 1 2 3 4 5 39 -1 *Output: 54 9 39 1 1*

input2.txt – 39 39 39 0 100 -1 *Output: 217 43 100 0 3*

Unit4: curvewithlines.py, ball.py: *no acceptance tests, tutors ran them*

Unit5: shapes.py – *acceptance test was a program calling fns in your program with calls shape(50,50,5,10) and shape(100,100,8,30) This was run by the tutor but mostly didn't work because the shapes.py file contained more than just the functions.*

Unit5: clock.py: - this was executed by the tutor to check on operation

Feedback points – mostly worth reading, even if not specifically identified for you

1. Expected only a couple of functions, pentagon and shape, in the file

Similarly to the last feedback sheet, functions only were expected in the shapes.py file. However, the Unit 5 sheet was perhaps unclear on this point again – so Quintin will endeavour to get this clear next time.

2. Checking for a condition that can only possibly be one of True or False

Sometimes, you can only be executing a block of code after satisfying some test – either in an *if* or a *while* condition. So take care not to test for the same condition again… e.g.

```
while a > 7:
    ...
    if b < 3 and a > 7:     # we can only be executing this if a > 7
        ...                 # hence no need for the test
```

3. Complete Canvas programs should always end with *complete()*

This is to ensure that the mouse and keyboard events are handled correctly – you'll see this more clearly next semester when we use the underlying Graphical User Interface library, tkinter.

4. Limited testing / experimentation of your program

It is easy to get your program working for one scenario, but no more. However, make sure it'll work in many different situations. An example was in the balls.py program. Many of you made the xInc value the same as the yInc value – so that the ball would travel diagonally across/down the screen, to the bottom right corner, and then back up again. Do you know if the program would

.... *I've truncated the 2 page feedback sheet here, to save space.*