CLASS TEACHING

Peter Jeavons

This session

- Why teach?
- How does it work?
- What does a marker do?
- What does a class tutor do?
- How do we do it well?

WHY TEACH?

Why teach?

- It can be very rewarding, seeing students learning under your guidance;
- It helps you to learn;
- It gives you good experience;
- It pays quite well;
- It helps the department.

HOW DOES IT WORK?

Definitions

In Oxford each “**course**” is organised by a lecturer who normally gives a series of “**lectures**” to everyone taking the course.

Alongside this, many courses have “**classes**”, which are interactive problem-solving sessions for groups of 6-12 students, that normally last about an hour.

Problems for the classes are set by the lecturer of each course.

Each class has a **marker**, and a **tutor**. It is possible to do just the marking, or just the tutoring, or both.

Definitions

For some courses, there are also “**tutorials**”, which are small-group teaching sessions normally for about two students, and usually arranged by colleges.

For some courses, there are also “**practicals**”, which are teaching sessions involving the use of computers or special equipment to undertake tasks set by the lecturer, managed by a **demonstrator**.

Administrative arrangements

In Computer Science classes are arranged for:

- **undergraduates:** Second year optional courses, plus all third and fourth year courses (but some colleges opt out of some classes).
- **postgraduates:** mainly MSc students, plus some Probationary DPhils.

Courses with practicals normally have 4 classes; courses without practicals normally have 6-7 classes.

Classes needing markers and tutors are advertised by the Academic Administrator Shoshannah.Holdom@cs.ox.ac.uk.

Administrative arrangements

Once you agree to tutor or mark for a particular class:

- Discuss the class timetable with Shoshannah or Jo and with the lecturer, before the term starts.
- Set a deadline for when you want the scripts handed in: normally lunchtime the day before; Jo tells the students.
- Book a room (with Jo.Leggett@cs.ox.ac.uk).
- Get question sheets from the lecturer or from the web (http://www.cs.ox.ac.uk/currentstudents/).
- Get model answers from the lecturer or /users/courses.

Payment – some approximate figures

For 7 sessions, each of 1 hour's duration:

- Tutors with a separate marker are paid £264.23 per term for a group of 1–10 students, £290.61 for a group of 11 students, and £317.23 for a group of 12+ students;
- Markers are paid £151.92 per term per group plus £15.97 per student;
- Tutors who do their own marking are paid at the Higher Tuition rates, a non-linear scale, but £556.50 for 10 students, and about £40 for each additional student.

If there are fewer than 7 classes, they get paid proportionately.

(Department intranet: Academic Admin, Info for lecturers etc.)

HOW TO BE A MARKER

Marking – overall strategy

- Do the questions yourself: do not just follow the model answer.
- Have a policy for dealing with late scripts.
- Marks do not count towards the students' final grades, but are an essential part of the feedback we give to them and to tutors.
- It is normally easiest to mark each question on all scripts rather than taking the scripts sequentially: it is easier to keep track of the details.

Marking – what to write

- Try to mark fairly: give credit to students for what they have achieved.
- Write comments on the work, explaining where the student has gone wrong, how he/she could have been done better, or how the presentation could be improved.
- Try to be encouraging: indicate when an argument is particularly elegant or impressive.
- Do not write out the model answer for the student!
- Correct the English, if appropriate.
- Do not nit-pick too much.

Marking - grades

- Give each question a grade on the following scale:
  - A: nearly all right, say more than 75%;
  - B: more than half right;
  - C: some progress, but less than half.

  with pluses and minuses as appropriate; the grade is intended to be indicative rather than 100% accurate.
- Give an overall grade for the piece of work, based on your grades for individual questions.
- Aim to spend 10–15 minutes per script.
- Enter the marks on the database https://www.cs.ox.ac.uk/minerva/, **after every class**.

Marking - feedback to tutors/supervisors

- It is essential that feedback is provided to the student's college tutor or supervisor.
- Enter the marks on the database https://www.cs.ox.ac.uk/minerva/, **after every class**; in addition to the overall grade, provide some details, at least the marks for each question, and preferably a few extra words.
- Flag particular problems to the tutor or supervisor (Jo will tell you who this is), e.g., unexplained absence, failure to submit work, unsatisfactory work.
- Colleges often complain about the quality of feedback, and can refuse to pay if they are not satisfied!
- Make sure the final report for each student is on Minerva by **9am on Monday of Week 8**.

Marker’s duties

- Discuss the work with the class tutor; highlight any difficulties.
- Attend the class, and give feedback to students.
- Take the class register, and enter it on the database.

HOW TO BE A TUTOR

Becoming a tutor

- Our policy is to ensure that class tutors have received sufficient training.
- Tutors should have previously acted as a Marker, or already be experienced teachers.
- The marker does the marking, attends the classes, presents some solutions to problems, and receives feedback from the class tutor.
- Subsequently, they can apply to be a tutor for classes.

Tutoring - before the class

- Make sure you can answer the questions on the sheet, and know the relevant background (e.g. material presented in the lectures).
- Decide which questions to talk about if you do not have time to cover them all.
- Think about what you want to say.
- Think about timing.
- Maybe prepare slides (see later).
- Prepare copies of model solutions for students.
- Check the room.
- Discuss students' scripts with the marker, so as to know what points to concentrate on.
- Have paperwork ready (e.g. the register).

Tutoring - during the class

- Remind the students of important concepts, where appropriate.
- Be friendly, smile, nod, give praise, etc.
- Encourage questions and interaction.
- Talk to the class (not to the board).
- Try to summarise important concepts at the end.

Tutoring - helping students learn

- Help the students to see how they could answer the questions; do not just present very polished answers.
- Try to lead them towards seeing the correct answer.
- Draw pictures.
- Give (small) examples.
- Maybe ask follow-up questions.
- Emphasise and repeat important points.

Tutoring - who answers the questions?

- Should you ask students to present their solutions?
  - It will keep them on their toes;
  - It will help you to identify misunderstandings;
  - Their answers might be less good than yours;
  - It will slow things down;
  - It will cause stress for some students.

Tutoring - slides or whiteboard?

- Slides work well for presenting factual information (e.g. bullet points);
- Slides discourage interactivity and discussion;
- Slides can be good for largish programs;
- Slides are bad for proofs;
- A whiteboard is good for non-linear writing.

Give out copies of any slides.

When using a whiteboard, write neatly, and in an organised way.

(For a tutorial, you might just write on paper.)

Example: binary search

We are given an integer array *a* of size *N*, sorted into increasing order. Write a program to find if the value *x* appears in the array. More precisely, the program should find the index of the first occurrence of *x* , if *x* does appear. The program should run in O(log *N*) time.

Example: binary search

- 2 perfect answers;
- 3 correct code but no comments;
- 5 small slips;
- 1 nowhere near

```
procedure Search(t: integer): integer;
(* pre: 0 ≤ n and a[0..n) is increasing
    post: a[0..Search(t)) < t and t ≤ a[Search(t)..n) *)
var l,r,m: integer;
begin l := 0; r := n;
(* inv: 0 ≤ l ≤ r ≤ n and a[0..l) < t and t ≤ a[r ..n) *)
   while l ≠ r do
      m := (l + r) div 2;
      if a[m] < t then l := m + 1 else r := m end;
   end;
   return l
end Search
```

Example: state space

Intelligent Systems
Problem Sheet 1

1. *The missionaries and cannibals problem.* Three missionaries and three cannibals are on the left bank of a river, with a boat that can hold one or two people. If on either side of the river or in the boat the number of cannibals exceeds the number of missionaries the cannibals will eat the missionaries. There are no other people involved, and any use of the boat must include at least one person. Finally, after each river crossing, all persons in the boat are required to disembark the boat and step onto the river bank; thus, the requirement on the numbers of missionaries and cannibals must be satisfied after each river crossing. The problem is to find a way for everyone to get safely from the left to the right river bank.

a. Formulate the problem precisely, and estimate the size of the state space.

b. Solve the problem optimally using an appropriate search algorithm. Is it a good idea to search for repeated states?

c. Why do you think people have a hard time solving this problem, given that the state space is so simple?

Example: state space

- 7 got full marks;
- 2 drew a picture
- 2 omitted last part

Intelligent Systems
Model Answers for Problem Sheet 1

1. This is just one out of many possible formalizations of the problem.

   a. *State space*: A state is a triple of the form $\langle m_L, c_L, b\rangle$, where m_L is the number of missionaries on the left river bank, c_L is the number of cannibals on the left river bank, and b represents the location of the boat and is either L or R. The number of missionaries and cannibals on the right river bank is implicitly determined by m_L and c_L. We assume that only valid states are considered—that is, that in all states the number of cannibals does not exceed the number of missionaries on either side of the river.

   Initial state: $\langle 3, 3, L\rangle$

   Actions: Actions are triples of the form $\langle m, c, dir\rangle$, where $m, c \in \{0, 1, 2\}$ determine the number of missionaries and cannibals, respectively, crossing the river, and *dir* is either $\rightarrow$ or $\leftarrow$.

   Transition model:

   $$\textsc{Result}(\langle m_L, c_L, L\rangle, \langle m, c, \rightarrow\rangle) = \langle m_L - m, c_L - c, R\rangle$$
   $$\textsc{Result}(\langle m_L, c_L, R\rangle, \langle m, c, \leftarrow\rangle) = \langle m_L + m, c_L + c, L\rangle$$

   We assume that the RESULT is defined only if the resulting state is valid; furthermore, ACTIONS(s) is defined as the set containing each action a such that RESULT(s, a) is well defined.

   Goal state: $\langle 0, 0, R\rangle$

   Step cost: 1

Tutoring - after the class

- Hand out copies of the lecturer's model answers;
- Be available to answer individual questions, within reasonable limits;
- Make sure everything is put in the database (at the end of term, include a fuller report);
- Flag particular difficulties to tutors/supervisors, e.g., unexplained absence, no work, very poor work;
- Give feedback to the lecturer, e.g., too many questions, theory not covered in lectures.

Getting feedback

- The course lecturer is supposed to attend (at least) one class and give you constructive feedback
  - Look on this as an opportunity, not an assessment;
  - Remind them when the classes are and invite them.
- Consider also asking a friend to attend, and attend his/her class in return.
- The lecturer will receive feedback from student questionnaires — make sure you receive it.