Introduction to Programming

Week 4 tutor’s kit

Note: You need copies of the class on page 4 for the tutorial and the mark sheet for Problem 1 - pp21-22 the COMP1001/1901 Resources book - these will be available from the department office already copied for you.

1. Notes about environment and situational issues for this class

Theme for the week Communicating objects - assisted by diagrams for visual thinking; Control flow, code tracing and generalisation, loops.

Important announcement

This is the deadline week for Problem 1. It will be foremost in their minds.

Due to the disruption caused by the strike in the first week of semester and the considerable number of people who missed the first week of classes, there will be no marks penalty for handling Problem 1 in and having it assessed in week 5. This means that we will do the marking of Problem 1 in weeks 4 and 5.

This will be announced in seminars.

About the CD

Some students will find that the CD is really easy to install and Blue runs as well as it does here. If this is not the case, it may be quite a lot of work to get it going. In that case, students can consult the Help (each other) Desk to see if their problem has already been answered. In addition, John Bignuccolo will be providing assistance.

However, the plan is that if students could not easily install the CD, they should ignore it till after they have done Problem 1. Certainly, do not let people use it as an excuse for not doing Problem 1.

You need to play two roles:

• calming those who cannot make the deadline and assure them they can do it for next week but they cannot start Problem 2 till they have been certified on Problem 1;
• helping students reflect on what they have learnt from Problem 1, especially about problem solving.

2. Tutorial script

<table>
<thead>
<tr>
<th>Activity 1</th>
<th>time for it minutes</th>
<th>total time minutes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduce session:</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Activity 2</th>
<th>time for it minutes</th>
<th>total time minutes</th>
</tr>
</thead>
</table>
Study problem solving model
This task is intended as an introduction to problem 2, based on reflection on Problem 1. It uses two models of problem solving: Polya and Boehm. Copies of these will be available with these notes - from your section leader. Also overhead slides will be available - helpful if you have a projector in your room. Also, each model is discussed below.

Note also that the code you bring to class should help the weakest students while still being useful as a problem solving reflection exercise for the stronger students.

2.1 Introduce task and ask students to form groups of about 5. Emphasise the role of this activity as learning from Problem 1 as a start for Problem 2. 5 10

2.2 Students use the handout sheet attached. It is a piece of code that a hypothetical student used to get started on Problem 1. They should critique it as they look at Polya’s model. 10 20

2.3 Class discussion 15 35

Understanding the problem - what sorts of activities did they do? Is Polya’s list of use? Polya was talking about maths and its not the same as computing so ask them to discuss this. (Note - most beginning students think that maths ability is important for CS - so differences as well as similarities should be emphasised, especially as much maths teaching involves repetition and practice at solving the same sorts of problems repeatedly where we aim to help students to solve new problems.)

Most important here is that a person’s understanding of a problem evolves as they work on it - this is possibly one of the defining features of a problem rather than an exercise - so try to get students to show they have worked out that you have to try to understand the problem but have to get on with other stages and then come back and review your understanding.

Divising a plan - how do Polya’s comments relate to the planning they are doing each week?

The most important thing to come up should be the idea of defining a simpler task - try to tease that out. Get people to volunteer simpler tasks they did first and to tell the class how that helped.

Carrying out the plan - how does this relate to what they have been doing, especially the weekly plan?

Looking back - again how does this relate? (That’s what we are doing in this activity.)

Activity 3
time for it total time

Visual thinking and the object model

5 40

3.1 Introduce the task below - it involves visualisation

3.2 Tracing some code - here the students work in groups and should draw a picture of the machine’s memory over time. This is a visualisation and involves tracing as follows.

Write line numbers against the code.

That way you can refer to each line.

Draw a box for each constant in the code.

Draw a table, with columns for each of the following:

-- line number (initially start, later each line a variable changes value)
-- each variable
-- remember to include all input parameters
-- and return values

Ask the class to volunteer good values to try.
This time, they need to trace execution over two classes. This refers to
the code attached to these notes plus the Account class on pages 41-42

3.2 Class review of the task - This time you need to draw the stages in
the execution of a Blue class:
• upon the creation of Test_out_things, there is a blob of memory
  reserved for it. There must be a small box for the creation class
  name. You need to emphasise that this blob of memory will hang
  around until Blue exits. Note that as shown in seminars, any instance
  variables would be shown here.
• Next, as part of the creation, a new blob of memory is created for the
  creation class. Emphasise that we draw it separately since it
  disappears once creation is complete. It should have two boxes, one
  each for the references to myaccount and youraccount
• then trace the call to creation for Account("Judy") - another blob of
  memory for the Account class, this time with two squares for the
  references to the instance variables _owner and _balance. There
  also need to be lists of the routine identifiers.
• then draw the memory blob for the creation routine - follow it
  through showing how it puts the parameters into the instance
  variables.
• then show that on completion of creation, its blob disappears (easy
  on a black/whiteboard).

You will have to use your judgement on how to run this part of the
tute. When you walked around as they tried this task beforehand, if
you saw them doing exactly what I have above and showing a really
good understanding of the object model, you could ask if people want
to go through the rest of the example. If you felt they did not have a
good understanding (and on last year’s experience, this will be the
case), keep going on through the example. If however, they are
looking very confused, leave it and move on. The object model is
important but difficult to really understand - we return to it later
in the semester. If students have problems now, don’t push it. Refer
them to Chapter 2 to read at their own rate.

If you are not confident you understand the object model make
sure you come to the section meeting so your section leader can go
through it.

Depending on how long you took to this point, you might take
requests for other things they would like to know about. EG. How you
will mark the Problem 1s (if the tute is before the lab), go through
syntax diagrams

3. 3-hr Workshop

Overview of the main activities. Today, you need to try to see each student - either to assess their Problem
1 or to discuss their planning for completing it by next week.

Starting this week, you will only attend the workshop for 1 hour each week. In today’s class you will use
that to get people started, mark off people’s plans received and collect something from each student,
hopefully a completed Problem 1 for many, but some code that was written in the last week for all the
others.
You also have about 15 minutes per student to mark Problem 1: 5 minutes for the demo and 10 to assess the code and report.

I would expect that many classes would work as follows:

- **Activity 1 - as below**
  - For students who are ready to hand in Problem 1, see that they get started on the self-assessment task, working with each other, preferably within their exiting groups.
  - Other students will want to work on Problem 1 - you will need to help them form groups of 2-3 so they can work on conceptual problems together. The workshop is not for private code development but you will have to convince students to use it as a chance to get over hurdles that are stopping them from completing problem 1 in their private time. You might help the groups find things that they can usefully do in the lab.
  - Start the marking. See the 5 minute demos for, say, 5 or 6 students. Then you could leave the lab and go to your desk to mark the code for these students and possibly a few more.
  - Come back to class for the last half hour to review the progress of people on the things they were working on. Collect plans for the coming week. Hand back any Problems 1s that you have completed the marking for. **Remember to record the marks BEFORE you return the work.**

### Activity 1

<table>
<thead>
<tr>
<th>time for it</th>
<th>total time</th>
</tr>
</thead>
<tbody>
<tr>
<td>minutes</td>
<td>minutes</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

**Class overview**

As usual introduce the activities:

- self-assessment web site
- marking problem 1
- progress of students

Collect Problem 1 submissions at this stage of class.

### Activity 2a

<table>
<thead>
<tr>
<th>time for it</th>
<th>total time</th>
</tr>
</thead>
<tbody>
<tr>
<td>minutes</td>
<td>minutes</td>
</tr>
<tr>
<td>60</td>
<td>65</td>
</tr>
</tbody>
</table>

Students should work from the self-assessment web site.

**A good student should be able to do Level 1 tasks by the completion of Problem 1 - ie this week.**

### Activity 2b

**You work around the class marking**

Get the single-face mark sheet from the office.

**Take a record of the marks for each student**

You will need to enter that using the inmarks program.

### Activity 3

<table>
<thead>
<tr>
<th>time for it</th>
<th>total time</th>
</tr>
</thead>
<tbody>
<tr>
<td>minutes</td>
<td>minutes</td>
</tr>
<tr>
<td>10</td>
<td>85</td>
</tr>
</tbody>
</table>

Pause gymnastics - as last week

### Activity 4

<table>
<thead>
<tr>
<th>time for it</th>
<th>total time</th>
</tr>
</thead>
<tbody>
<tr>
<td>minutes</td>
<td>minutes</td>
</tr>
</tbody>
</table>

Continue marking as they continue working

Finally, remember to mail feedback, comments and requests for seminar content to your section leader. **Make sure you mail a status report on the marks and the rate of marking.**
Judy Kay
1999 PBL Co-ordinator for Comp1001/1901
March 1999
class Test_out_things is

==================================================================================
== Author: Judy Kay
== Version: 1
== Date: March 1997
== Short: A class for testing out my understanding of the Account class
==
== This is simply a class which I will use for experimenting with
== the Account class, to make sure I know how it works
==
==================================================================================

uses Account

internal

interface

creation is
== Here is where I tinker with things - it is the only class I need
== It just creates some objects and runs them
var

myaccount : Account
youraccount : Account

end creation

end class