A-LEVEL COMPUTER SCIENCE • HOLIDAY HOMEWORK

12 Software Development – Part 2

Name: _____ Class: _____ Date: _____

Vocabulary

Write each word three times.

English	**Write it 3 times** (English)		
structure chart			
state-transition diagram			
dry run			
white-box testing			
black-box testing			
integration testing			
test plan			
extreme data			

Recall

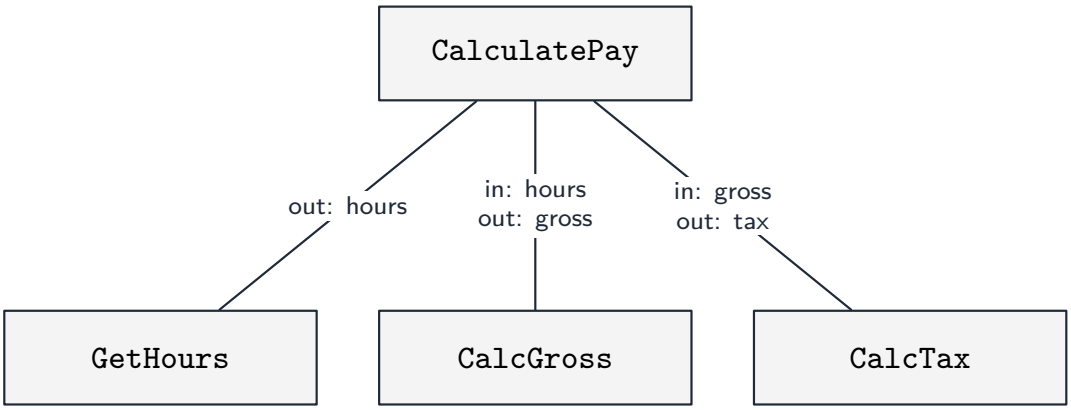
In **Part 1** you met the development life cycle and its models (waterfall, iterative, agile), the three error types (syntax, run-time, logic), test data (normal, abnormal, boundary) and the kinds of maintenance. Part 2 adds tools for *designing* a program, and methods for *testing* it.

New ideas

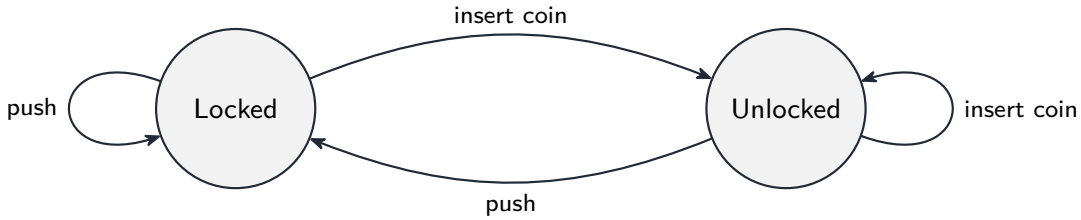
Read these first. Each idea has a short worked example. Then do the Practice.

1 Design tools

A **structure chart** shows a program broken into modules (subroutines), with the parameters passed between them – the caller above, the modules it calls below, and small arrows for the data going down and results coming back up.



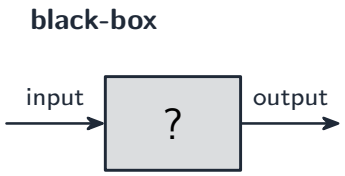
A **state-transition diagram** shows the states a system can be in and the events that move it between them – useful for a vending machine, a lock, or a traffic light.



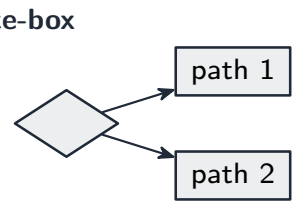
Worked example. A "make a drink" program's structure chart shows a top module calling `takeMoney`, `chooseDrink` and `dispense`, with the price passed down and an "ok?" result passed back up.

2 Testing methods

- a **dry run** – trace the code by hand, writing each variable in a table;
- **white-box testing** – designed from the *code's* structure, to cover every branch;
- **black-box testing** – designed from the *specification* only: give inputs, check the outputs;
- **integration testing** – combine the modules and test that they work together.



test from the spec



test every path in the code

Worked example. Black-box: a tester only knows "input two numbers, get their sum", so they try $2 + 3$ and expect 5. White-box: a tester who can see the code chooses inputs that make every `IF` branch run.

■ 3 Test strategy and test data

A test strategy is the overall plan; a **test plan** lists each test with its input, expected output and a column for the actual output. As well as normal, abnormal and boundary values, include **extreme data** – the largest and smallest values that are still accepted.

Watch out: black-box testing cannot tell you a hidden branch of code was never run – that is exactly what white-box testing checks.

Practice

12.1 State what a structure chart shows about a program. [2]

12.2 State what a state-transition diagram shows, and give one system it suits. [2]

12.3 State one difference between white-box and black-box testing. [2]

12.4 A field accepts marks from 0 to 100. Give one example each of (a) normal, (b) abnormal, (c) boundary data. [3]

12.5 State what integration testing checks. [1]

12.6 Explain *why* a project uses both black-box and white-box testing, rather than just one. [2]

12.7 Challenge. A turnstile is either `Locked` or `Unlocked`. Describe, in words, a state-transition for it: name two events and state what each one changes. [2]