TEMASEK
JUNIOR COLLEGE

Temasek Junior College
JC H2 Computing
Problem Solving & Algorithm Design 11
Testing

1. Testing

Testing the New System is the process of detecting errors in a piece of software. It can be carried out at all stages of the *software development cycle*.

Example 1 CW2011(A)

The examinations department of a school needs to store data on the examinations taken by its students. Examination subjects are classified as one of two types.

A subject can be classified as academic. The assessment process for an academic subject comprises two written papers. The duration of each of these two papers will be stored.

A subject can be classified as practical. For the practical assessment process there are no written papers but there is a single practical examination. A practical examination has a duration and a deadline date. The duration is the maximum time allowed for students to complete the work to be assessed. The deadline date is the date by which the practical assessment must be completed.

The examinations department decides to store the following data:

- **SubjectCode** is used to uniquely identify a particular subject and is four digits.
- **Name** is the name of the subject and is at most 30 characters.
- **SubType** is the type of subject and can have the values 'A' or 'P'.
- **P1Len** is the duration of the first written paper and can have the values 2.0, 2.5 and 3.0. For a practical subject this field would have the value 0.0.
- **P2Len** is the duration of the second written paper and can have the values 2.0, 2.5 and 3.0. For a practical subject this field would have the value 0.0.

EXAMS.DAT Sample data:

```
5566|Math|A 2.5 3.0 0.0 000000
6846|Physic|A 2.0 3.0 0.0 000000
5846|Computing|P 0.0 0.0 15.0 110430
0023|Biology|A 2.5 2.0 0.0 000000
1234|Chemistry|A 2.0 3.0 0.0 000000
3258|China Studies|A 3.0 3.0 0.0 000000
7789|Literature|A 2.5 3.5 0.0 000000
9999|Geography|A 2.5 2.0 0.0 000000
8463|Art|P 0.0 0.0 30.0 110829
2317|Chinese|A 2.0 2.0 0.0 000000
9517|Math|A 2.5 3.0 0.0 000000
```

- **PracLen** is the duration of the practical examination and can have values in the range 10.0 to 40.0. For an academic subject this field would contain the value 0.0.
- **CDate** is the final date by which the practical examination should be completed. It is six characters and is in the form 110504 (which represents 4th May 2011). For an academic subject this field would have the value '000000'.

A module, **ADDSUBJECT,** which when called will allow the user to enter data on a new subject. This data is then written to a text file named **EXAMS.DAT.**

EXAMS.DAT has the following structure:

<NoOfRecords><UpdateDate>
<SubjectCode><Name><SubType><P1Len><P2Len><PracLen><CDate>
<SubjectCode><Name><SubType><P1Len><P2Len><PracLen><CDate>
<SubjectCode><Name><SubType><P1Len><P2Len><PracLen><CDate>

NoOfRecords is the number of records stored in file
UpdateDate is the date that the file was last updated and is in the form DD-MM-YYYY.

Discuss with details the validations [**Testing**] need to apply for the data to be captured before they store into the file EXAM.DAT.

Draw flowcharts for the validations of DD-MM-YYYY, SubjectCode, P1Len and CDate.

1.1 Test plan

- For each stage a ***test plan*** will exist.
    - The plan will describe each item that needs to be tested.
- It will provide instructions or operations to be carried out and ***test data***, which are the inputs to be used.
- It will also provide the expected outcomes.
- Each set of test data and its expected outcomes forms a ***test case***.

The assumption of testing is that the algorithm designer/programmer will submit a set of test data which is sufficiently inclusive so as to fully test the algorithm/program but this may not be the case.

If the test data submitted by the algorithm designer/programmer shows a variance between the predicted and the actual output, this is a bug which will be addressed.

There are many algorithm/programs however which are expected to process an infinite number of acceptable inputs.

The algorithm designer/programmer can only supply representative samples of these inputs and therefore it is possible that undiscovered problems will remain.

In essence, testing can only show the presence of bugs, it cannot prove the absence of bugs.

Once the algorithms/programs have been written, they must be tested.

The testing phase can be a very tedious and time-consuming part of program development.

The algorithm designers/programmers are completely responsible for testing their algorithms/programs.

A **test** plan is usually written whilst the system is being developed (design phase). The test plan will contain details of every single thing that needs to be tested.

For example:

- Does the system **open and close** properly?
- Can data be **entered**?
- Can data be **saved**?
- Can reports be **printed**?
- When you do something **wrong**, does an **error message** appear?
- Is **invalid data rejected**? E.g. if you are not allowed to enter an amount above $1,000 on the system then a value of 1,001 should not be accepted (i.e. does the **validation** work?)

Test plans are very detailed, and contain many tests. Each test is specified very precisely.

A typical test would contain:

- Details of **what** is being tested
- The **test data** to use
- What is **expected to happen** when the test is performed

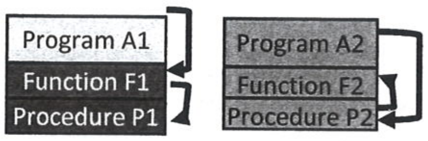
1.2 Designing test data/test cases

Algorithm/Program testing is best performed at three different levels:

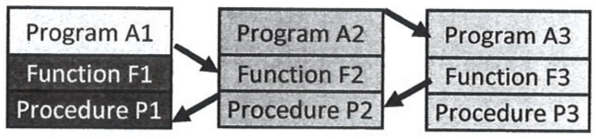
1. Components (subprograms) need to be individually tested to ensure that they conform to specifications.

   Function F1
   Procedure P1

2. When combined to form a program, the components must not act in unexpected ways, so each program module must be tested.

3. The suite of programs as a whole must be tested to ensure that all programs integrate correctly.

Although algorithm/program testing in theory should be exhaustive with every possible route being tested, with a large algorithm/program this is impractical as it would mean performing perhaps millions of tests. At a minimum, the test cases should be selected to ensure that:

1. Every statement in the program is executed at least once.
2. The effectiveness of all sections of code which detect invalid input is tested.
3. The accuracy of all processing is verified.
4. The program operates according to its original design specifications

1.3 Selecting test data

When choosing what data to use to test a system, you need to think about why we are testing the system to see if it works, and to check it doesn't break.

The most important area is to find the limits (or boundaries) of the data.

Tests should then be performed:

- just inside the boundary
- at the boundary
- just outside the boundary.

To do this, we use three types of test data:

10.3.1 Normal Data Values

This is data that would **normally** be entered into the system.
The system should accept it, process it, and we can then check the results that are output to make sure they are correct.

Example 2

> **PracLen** is the duration of the practical examination and can have values in the range 10.0 to 40.0.

Accepted normal values:
- 11
- 25
- 33
- 39

10.3.2 Extreme Data Values

Extreme value are still **normal** data.
However, the values are chosen to be at the absolute **limits of the normal range**. Extreme values are used in testing to make sure that **all normal values** will be accepted and processed correctly.

Example 3 In systems that deal with text, the extreme values are defined by how long the text can be.

> **Name** is the name of the subject and is at most 30 characters

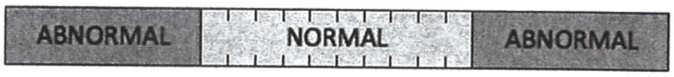
The limits would be:

"Art"	(3, min. length)
" Geography"	(30, max. length)

Example 4 In a system that was designed to accept and process test marks (percentages), then extreme test values would be:

- 0 (lowest possible value)
- 100 (highest possible value)

1.3.3 Abnormal Data Values



This is data that should **not normally be accepted** by the system - the values are **invalid**.

The system should **reject** any abnormal values.

Abnormal values are used in testing to make sure that invalid data does not **break** the system.

Example 5 In systems that deal with text, the values are defined by how long the text can be.

➢ **Name** is the name of the subject and is at most 30 characters

The abnormal test values would include:

- "" (0 in length, empty string)

Example 6 PracLen is the duration of the practical examination and can have values in the range 10.0 to 40.0.

Then abnormal test values would include:

- -10.0
- 8.0
- 42

1.3.4 Inspections and Walkthroughs

Test cases are usually applied to code that is executed. However, errors can also be examining the code manually, such as using inspections and walkthroughs.

- ***Inspection*** is a technique where a small team will read through the code during a group meeting, analysing it with a **checklist**. The checklist documents common problems, which the team will be explicitly looking for during the inspection.

- ***Walkthrough*** is a similar technique. In this case the team will manually execute the code using inputs from defined test cases. This is a formalised team version of a ***dry run***.

Example 7 (Class discussion) Name and describe three types of test data which are used when testing a system. IGCES 04172012/11 [3]

1.4 Testing strategies

Obviously, a system must be thoroughly tested before being installed to make sure that all errors are discovered and corrected before going 'live'. It is part of the designer's job to come up with a test strategy which will ensure that all parts of the system are properly tested.

Example 8 CW2011(B)

Given:

- SubjectCode is used to uniquely identify a particular subject and is four digits.
- Name is the name of the subject and is at most 30 characters.
- SubType is the type of subject and can have the values 'A' or 'P'.
- **PracLen** is the duration of the practical examination and can have values in the range 10.0 to 40.0. For an academic subject this field would contain the value 0.0.

Produce test data for **PracLen** explaining the purpose of each test. Include a screenshot of each test which clearly shows the test data and the outcome of the test. [4]

Requirement: Test data using

- normal data (valid, acceptable), purpose of test (range is stated to qualify example data)
- extreme data (boundary), purpose of test (range is stated to qualify example data)
- erroneous data (invalid), purpose of test (range or type is stated to qualify example data)
- example data (from test plan) and correct result shown in screenshots for any two different (types of) tests

Test plan: Test data for **PracLen**

PracLen	Validation check	Result	Reason	Error messages
	Presence	Invalid	Not present	"Duration of Practical exam cannot leave empty"
"twelve"	Data type	Invalid	Erroneous data [type]	"Invalid data! Only numeric value is accepted"
12.0	Range	Valid	Normal data	
10.0	Range	Valid	Extreme data	
40.0	Range	Valid	Extreme data	
-5.0	Range	Invalid	Out of range	"Duration is out of range (10.0 – 40.0)"
45.0	Range	Invalid	Out of range	"Duration is out of range (10.0 – 40.0)"

Tutorial 11 [Testing] Q1 and 2

Tutorial 11 [Testing]

1. Refer to **Example 8 CW2011(B),** copy and complete the following tables for produce test data for SubjectCode, Name, SubType and explaining the purpose of each test.

SubjectCode	**Validation check**	**Result**	**Reason**	**Error messages**

Name	**Validation check**	**Result**	**Reason**	**Error messages**

SubType	**Validation check**	**Result**	**Reason**	**Error messages**

2. A program has been written which inputs the marks gained by 1000 candidates in an examination. The top mark possible is 100. The program calculates the mean mark and outputs the highest mark, the lowest mark and the mean.

(a) It is decided to test the program by using sets of test data containing four marks each time. Explain why the testing was carried out using only four marks at a time. [2]

(b) Using the table below, give three separate test cases for testing the program. [6]

	Input data	Reason for test	Expected result
Test 1			
Test 2			
Test 3			