Teaching Black Box Testing to High School Students

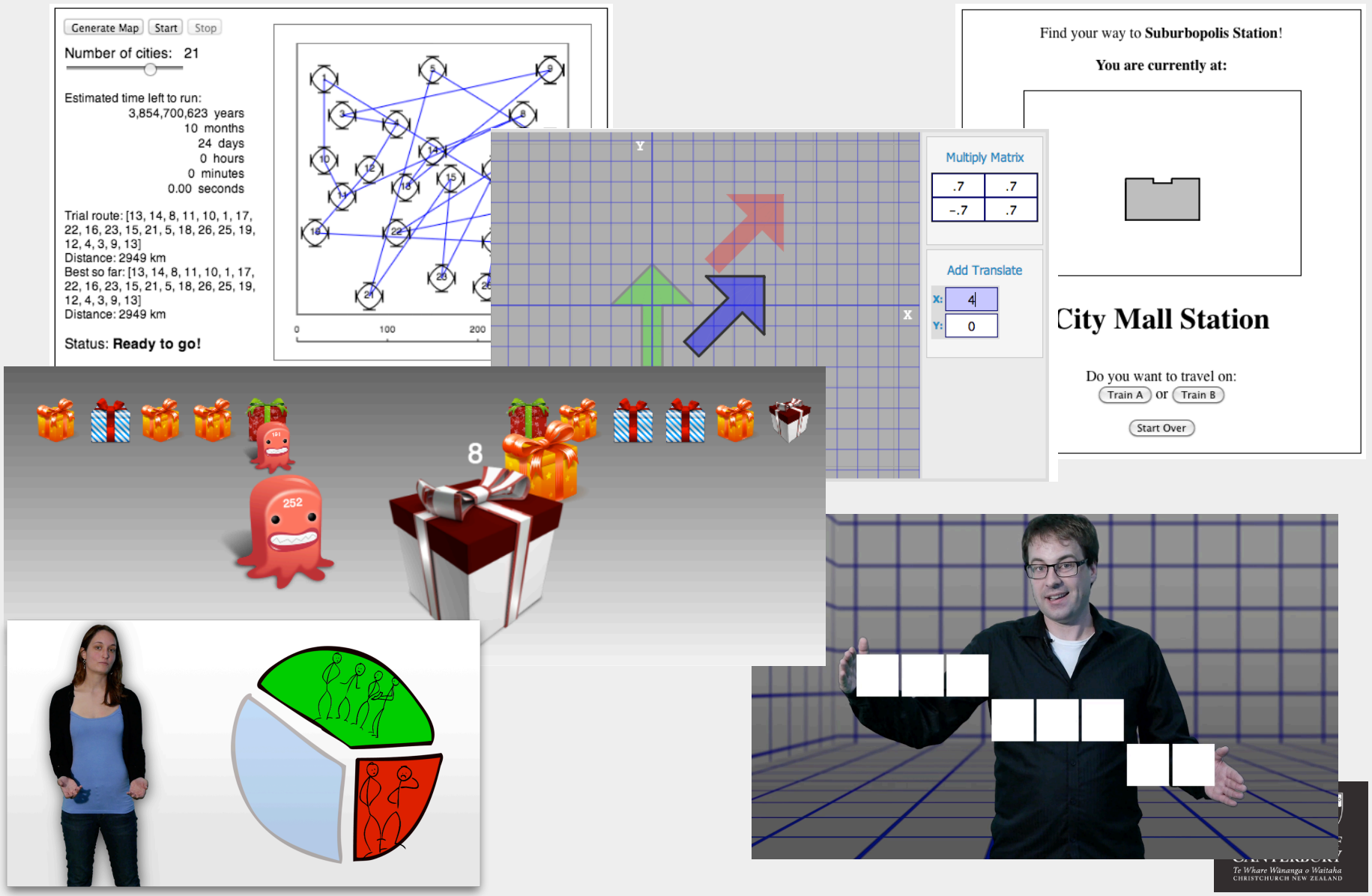
Amitrajit Sarkar
Christchurch Polytechnic Institute of Technology
New Zealand

Tim Bell
University of Canterbury
New Zealand

UC UNIVERSITY OF CANTERBURY
Te Whare Wānanga o Waitaha
CHRISTCHURCH NEW ZEALAND

CS Field guide

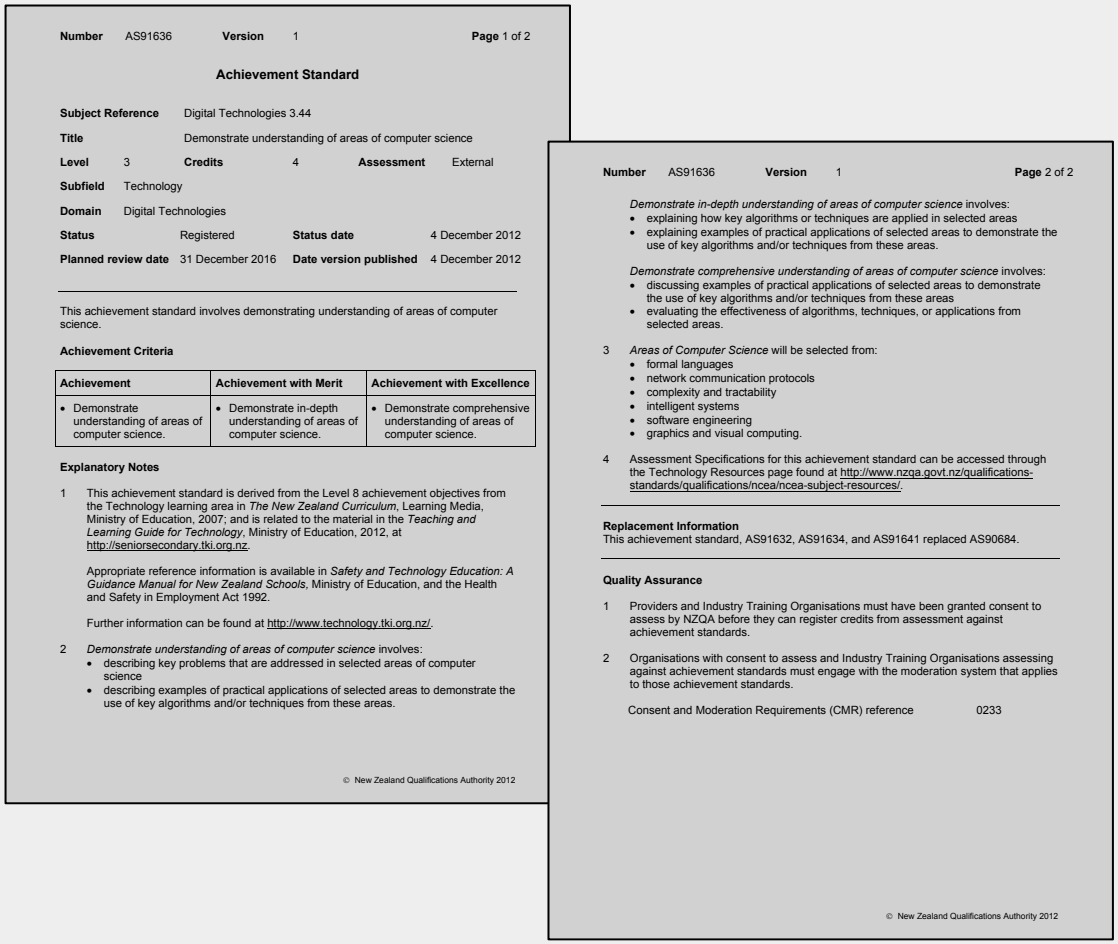
Black box testing



computerscienceclub.org

AS91636

- Demonstrate understanding of areas of computer science
- Level 3 (Year 13)



Number AS91636 **Version** 1 **Page** 1 of 2

Achievement Standard

Subject Reference Digital Technologies 3.44

Title Demonstrate understanding of areas of computer science

Level 3 **Credits** 4 **Assessment** External

Subfield Technology

Domain Digital Technologies

Status Registered **Status date** 4 December 2012

Planned review date 31 December 2016 **Date version published** 4 December 2012

This achievement standard involves demonstrating understanding of areas of computer science.

Achievement Criteria

Achievement	Achievement with Merit	Achievement with Excellence
• Demonstrate understanding of areas of computer science.	• Demonstrate in-depth understanding of areas of computer science.	• Demonstrate comprehensive understanding of areas of computer science.

Explanatory Notes

1 This achievement standard is derived from the Level 8 achievement objectives from the Technology learning area in *The New Zealand Curriculum*, Learning Media, Ministry of Education, 2007; and is related to the material in the *Teaching and Learning Guide for Technology*, Ministry of Education, 2012, at http://seniorsecondary.tki.org.nz.

Appropriate reference information is available in *Safety and Technology Education: A Guidance Manual for New Zealand Schools*, Ministry of Education, and the Health and Safety in Employment Act 1992.

Further information can be found at http://www.technology.tki.org.nz/.

2 *Demonstrate understanding of areas of computer science* involves:
- describing key problems that are addressed in selected areas of computer science
- describing examples of practical applications of selected areas to demonstrate the use of key algorithms and/or techniques from these areas.

Number AS91636 **Version** 1 **Page** 2 of 2

Demonstrate in-depth understanding of areas of computer science involves:
- explaining how key algorithms or techniques are applied in selected areas
- explaining examples of practical applications of selected areas to demonstrate the use of key algorithms and/or techniques from these areas.

Demonstrate comprehensive understanding of areas of computer science involves:
- discussing examples of practical applications of selected areas to demonstrate the use of key algorithms and/or techniques from these areas
- evaluating the effectiveness of algorithms, techniques, or applications from selected areas.

3 *Areas of Computer Science* will be selected from:
- formal languages
- network communication protocols
- complexity and tractability
- intelligent systems
- software engineering
- graphics and visual computing.

4 Assessment Specifications for this achievement standard can be accessed through the Technology Resources page found at http://www.nzqa.govt.nz/qualifications-standards/qualifications/ncea/ncea-subject-resources/.

Replacement Information
This achievement standard, AS91632, AS91634, and AS91641 replaced AS90684.

Quality Assurance

1 Providers and Industry Training Organisations must have been granted consent to assess by NZQA before they can register credits from assessment against achievement standards.

2 Organisations with consent to assess and Industry Training Organisations assessing against achievement standards must engage with the moderation system that applies to those achievement standards.

Consent and Moderation Requirements (CMR) reference 0233

Black box testing

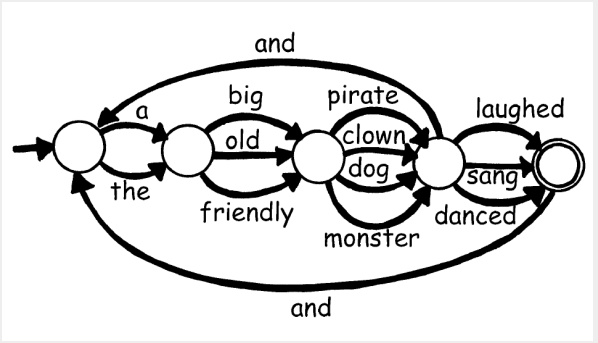
UC UNIVERSITY OF CANTERBURY
Te Whare Wānanga o Waitaha
CHRISTCHURCH NEW ZEALAND

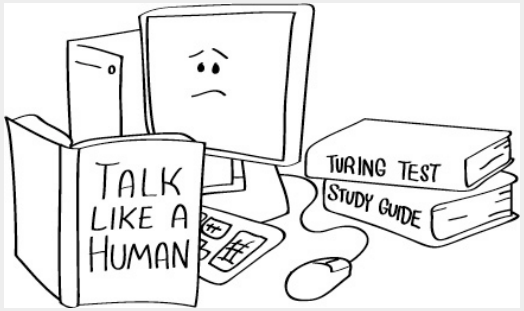
AS91636 (level 3)

- describing key problems that are addressed in selected areas of computer science
- discussing examples of practical applications of selected areas to demonstrate the use of key algorithms and/or techniques
- evaluating the effectiveness of algorithms, techniques, or applications

AS91636 (level 3)

- formal languages
- network communication protocols
- complexity and tractability
- intelligent systems
- software engineering
- graphics and visual computing





Achievement Standard

Subject Reference	Digital Technologies 3.44				
Title	Demonstrate understanding of areas of computer science				
Level	3	**Credits**	4	**Assessment**	External
Subfield	Technology				
Domain	Digital Technologies				
Status	Registered	**Status date**	4 December 2012		
Planned review date	31 December 2016	**Date version published**	4 December 2012		

Goals

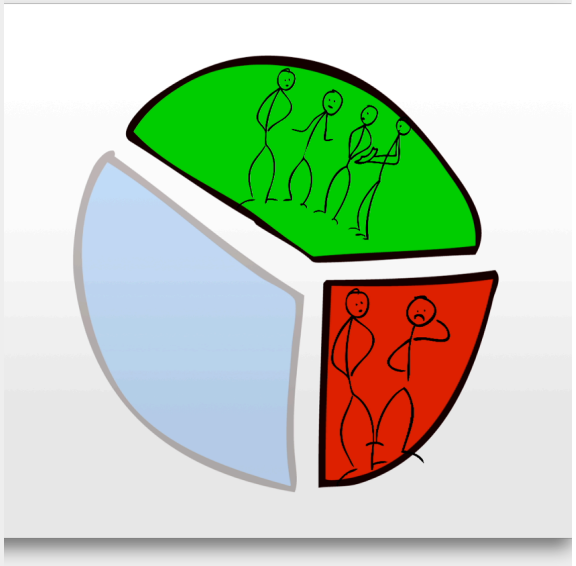
- Grounding in CS concepts
- Discover a passion for computing

Software testing

- Not debugging
- Black box testing
  - Functional testing
  - User acceptance testing

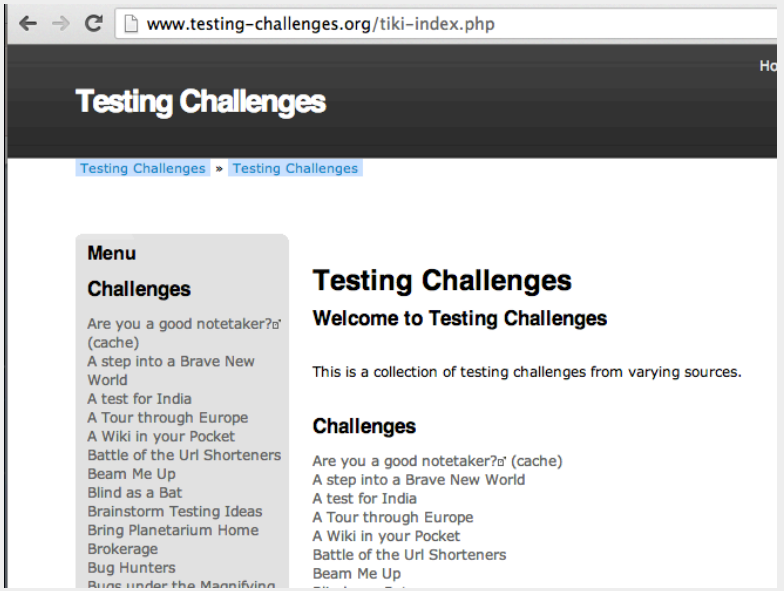
Importance of testing

- Programming 20% of projects
- About 25% (1999) to 33% (2009) of projects succeed
- ¼ failed outright or cancelled (2009)
- Testing 50% of cost
- Top-of-mind issue for managers
- MS: One tester per programmer
- Growing in ACM curriculum



Testing teaching

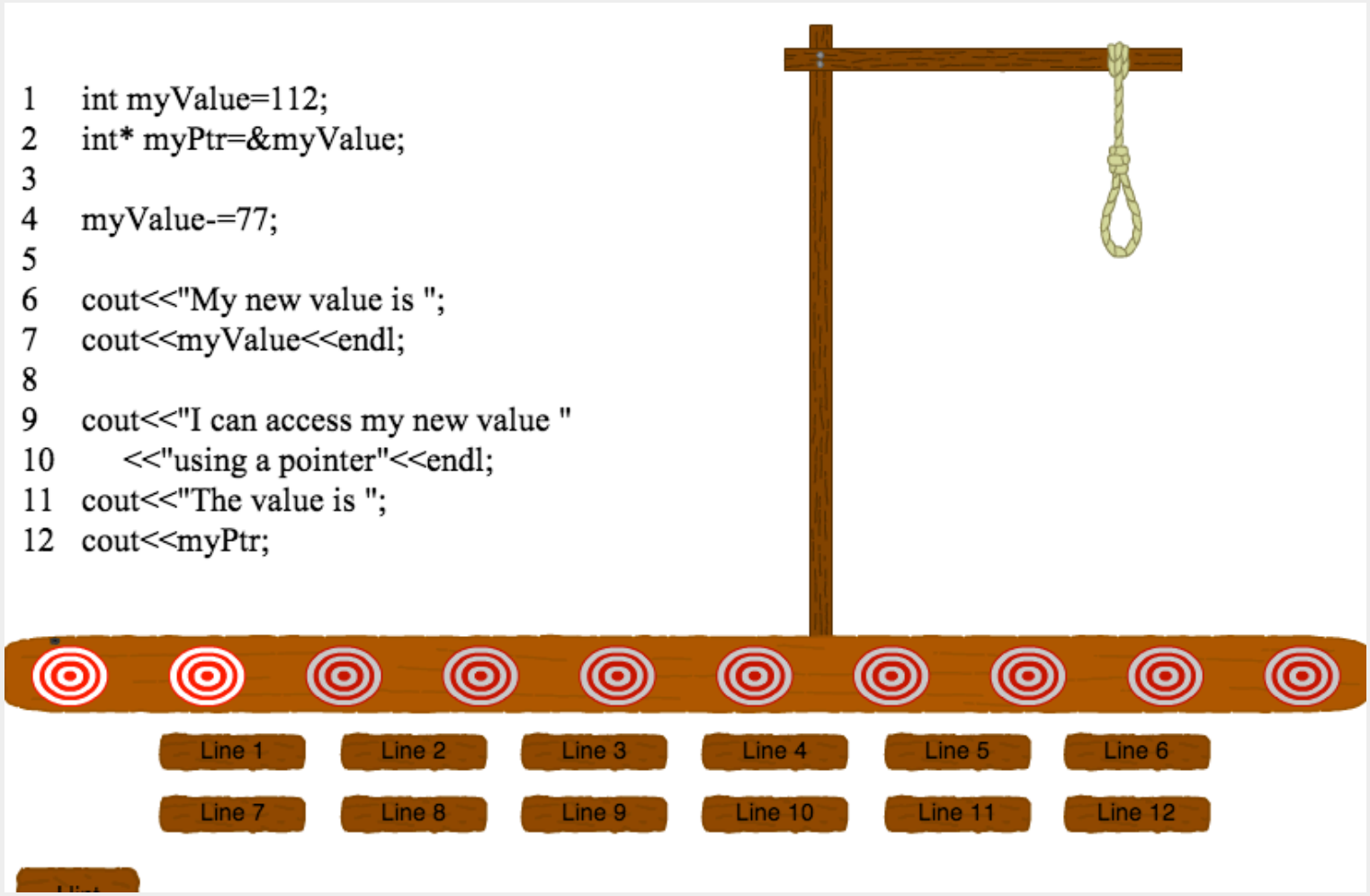
- Florida IT testing course
- Georgia tech
- NSF funded courses
- testing-challenges.org



UC UNIVERSITY OF CANTERBURY
Te Whare Wānanga o Waitaha
CHRISTCHURCH NEW ZEALAND

Hang bug


```
1    int myValue=112;
2    int* myPtr=&myValue;
3
4    myValue-=77;
5
6    cout<<"My new value is ";
7    cout<<myValue<<endl;
8
9    cout<<"I can access my new value "
10       <<"using a pointer"<<endl;
11   cout<<"The value is ";
12   cout<<myPtr;
```



www.cosc.canterbury.ac.nz/csfieldguide/student/Software%20engineering.html

Computer Science Field Guide

Student Version

Table Of Contents

16. Software engineering

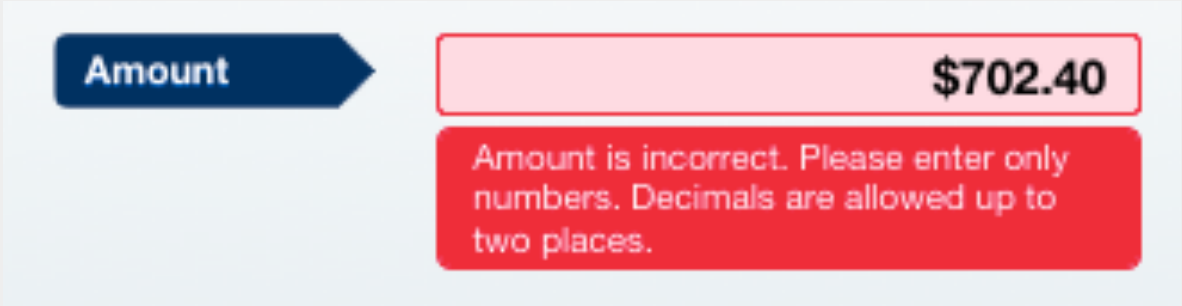
Warning: This chapter is still in development; the current version has been released for the 2013 school year with enough material to enable students to complete the NZ 3.44 achievement standard, but more material may be added or updated during the year to give more options for student projects.

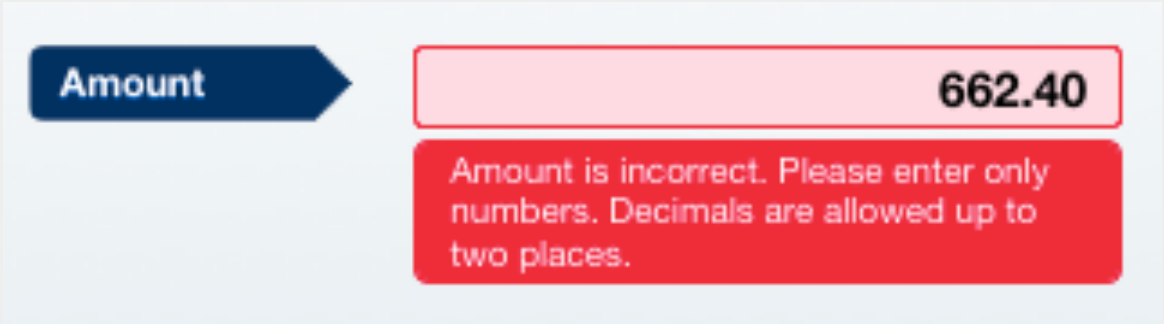
16.1. What's the big picture?

Software failures happen all the time. Sometimes it's a little bug that makes a program difficult to use; other times an error might crash your entire computer. Some software failures are more spectacular than others.

In 1996, The ARIANE 5 rocket of the European Space Agency was launched for its first test flight: Countdown, ignition, flame and smoke, soaring rocket... then BANG! Lots of little pieces scattered through the South American rainforest.

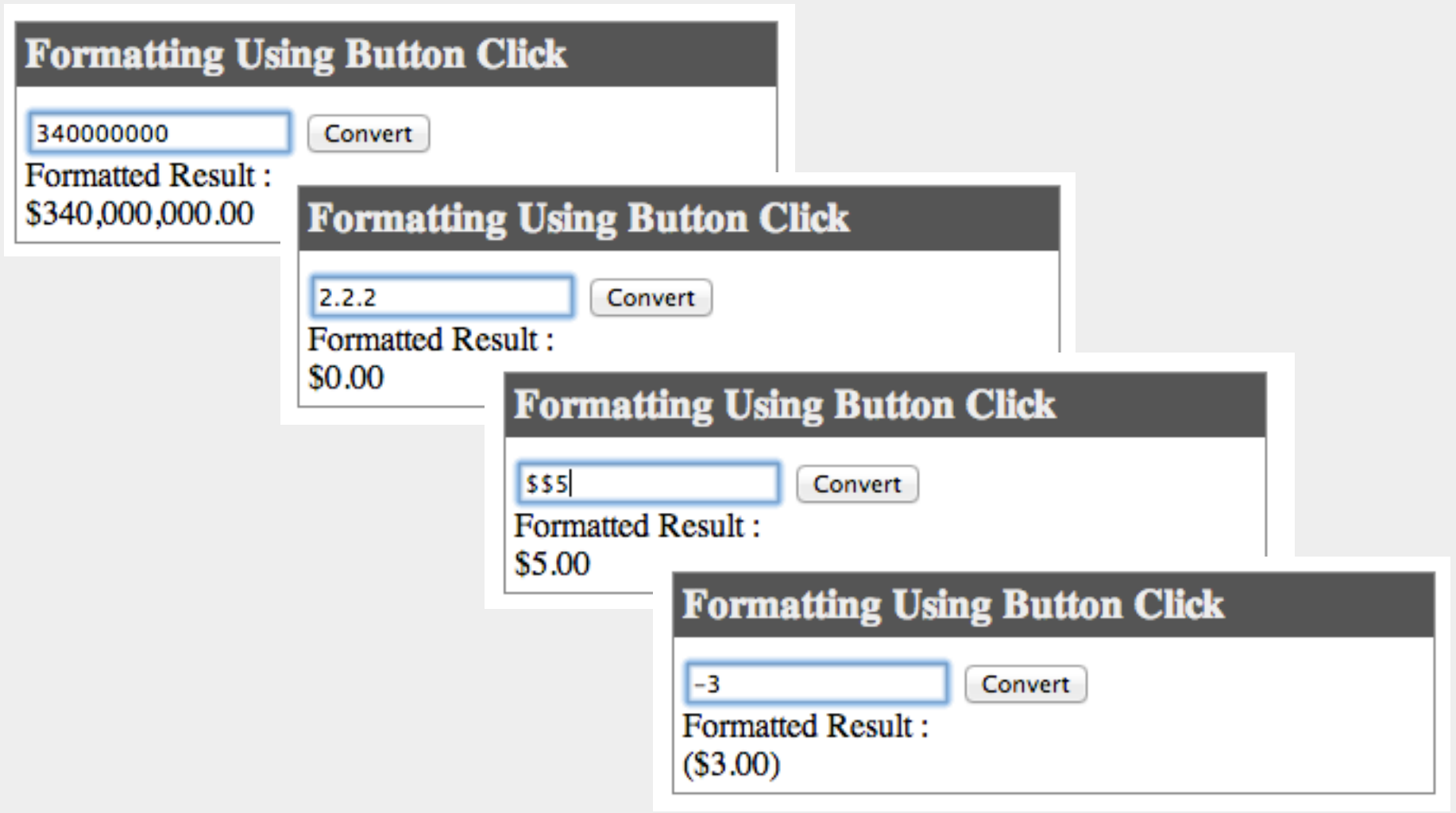
Initial testing experience



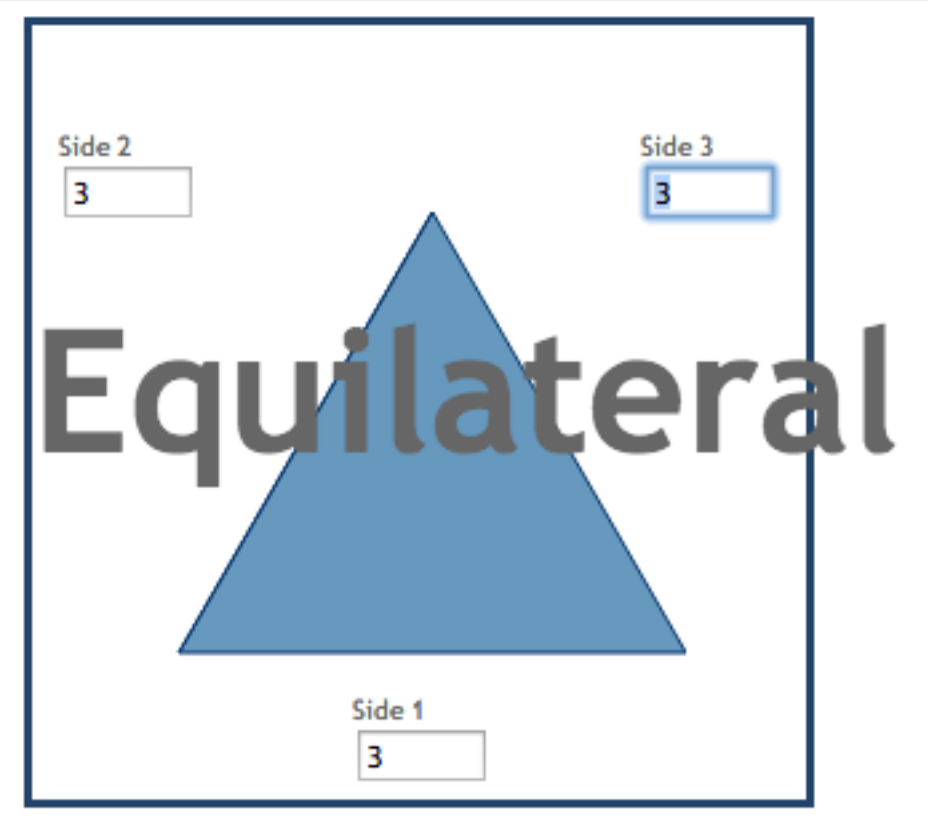


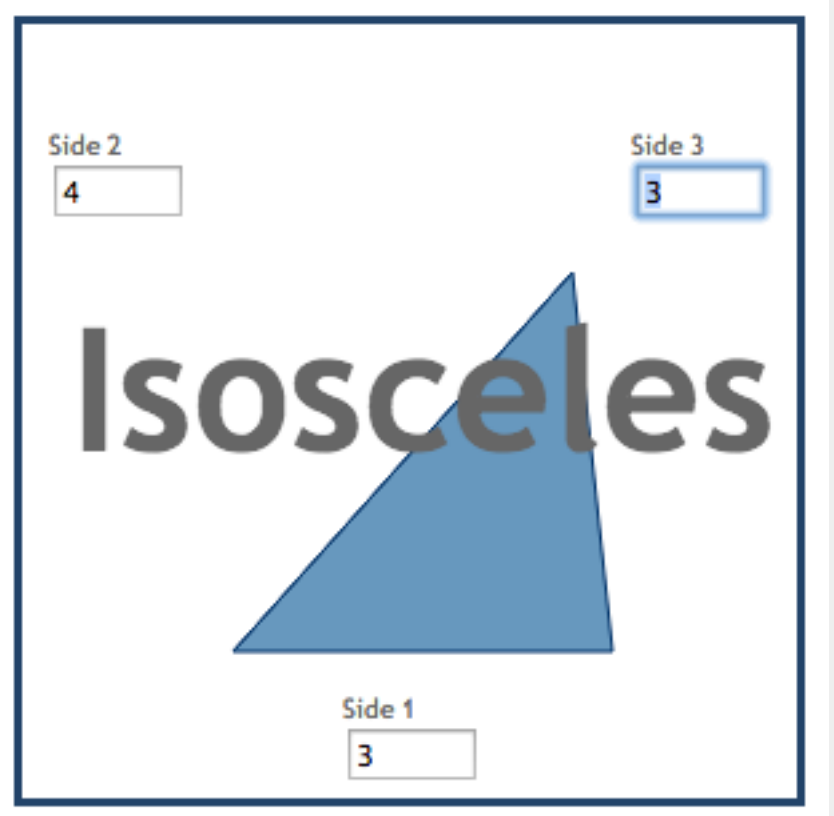
Initial testing experience

Black box testing



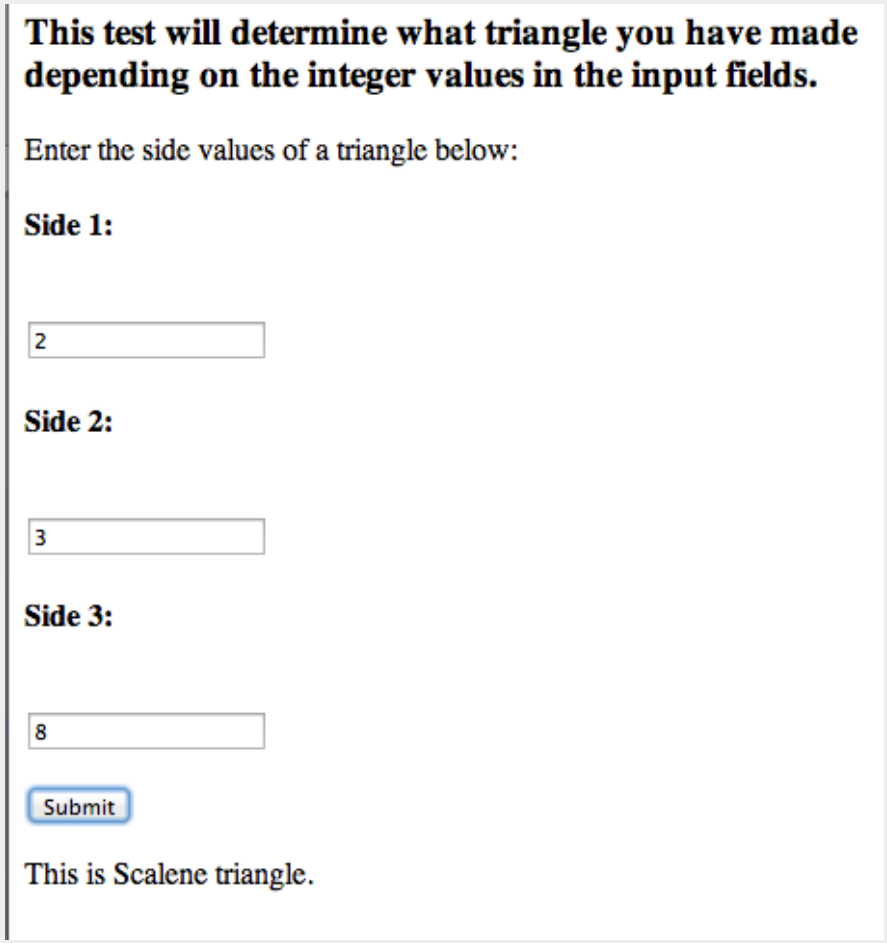
Meyer's triangle problem





http://practice.agilistry.com/triangle

System with errors



List of test cases

We want you to write a set of test cases—specific sets of data—to properly test a relatively simple program.Create a set of test data for the program—data the program must handle correctly to be considered a successful program.Here's a description of the program:

1. The Triangle program accepts three integer values as input. Each value represents a side of the triangle.
2. If the inputs are invalid (sides smaller than 0, or not integers) or if fewer than three values are provided the program outputs the message "Invalid input value(s)".
3. If the length of the largest side is greater or equal to the sum of the lengths of the two smaller sides the program will output the message "Not a Triangle".
4. If all three sides of the triangle are of equal length the program will output the message "Equilateral".
5. If exactly two sides of the triangle are of equal length the program will output the message "Isosceles".
6. If all three sides of the triangle are of different lengths the program will output the message "Scalene".

```
1,1,1,equi
3,4,5,sc
0,0,0,e
-3,3,4,s
```

check

You have met 6 out of 10 criteria.

1. must have one test case with all sides equal (and non zero)
2. must have one test case with all sides equal (and zero)
3. must have one test case with no sides equal (and no zero sides)
4. no side must be greater than the sum of the other two
5. no sides should be negative
6. Must have one right angled triangle

Development

- Iterative improvement
- Short explanations
- Warm-up examples
- Use program first
- Teacher guidance

Summative assessment

Ideas to test an alarm clock in 30 minutes

Assuming that this is the only question put in forth of us and we cannot get any more information on the mission like time required to run tests, how much money we could spend on testing, who the product end users are, is it a handheld alarm clock or a digital one, etc. This would be our generic approach considering that the alarm clock is a physical device with only 2 buttons one for the snooze and one for the stop as displayed in the picture below.

http://tuppad.com/blog/wp-content/uploads/2010/08/How.pdf

Personalised report

- Key problem: assure product (digital alarm clock) suitable for the end user.
- Key algorithms or techniques: black-box testing (and acceptance testing
- Practical application: software engineering projects
- Evaluation: critically evaluate an existing product (the alarm clock).

Lesson:

- ~~How to test~~
- Testing is hard
- Testing is important in SE

csfieldguide.org.nz

tim.bell@canterbury.ac.nz

UC
UNIVERSITY OF CANTERBURY
Te Whare Wānanga o Waitaha
CHRISTCHURCH NEW ZEALAND