Curtin University
Department of Computing
Software Engineering Testing

Semester 2, 2017

Assignment

Due date: Thursday 2nd November 2017 (Submission during first 10 mins of Week 12 lecture)

Assignment Description

This assignment is divided into four tasks. You will work on the assignment in **groups of three students**. All tasks are compulsory.

Task 1 (worth 35%): This task requires your team to research one category of testing tools and create a recommendation based on whether or not they should be purchased by a software development house specialling in ICT solutions aimed at the banking and financial industry (see the relevant **scenario** on the next page).

This task consists of two parts: a presentation and a written summary. It is up to the members of each team to decide on the distribution of work amongst members, the allocation of tasks, etc.
In the preparation of both the presentation and the summary you may consider the following general issues:

- The purpose and functionality of the given class of tools (what they are for, what they do)
- At what stage(s) of the software development lifecycle they can be used
- Inputs and outputs
- How do they work (a general description)
- Illustrative examples
- Merits and pitfalls
- Views of the Software Testing community on your class of tools
- Examples of commercially available tools and open source tools in your category, commenting on
    - their software and hardware platforms
    - their ease to learn
    - price (if commercial)
    - evaluation by the users
- Your evaluation:
    - How well this class of tools would meet the needs of the software house described in the "Scenario"?
    - What are the advantages and disadvantages of the tools regarding these specific needs?
    - If the tools were introduced, would the software house need to make additional investments (e.g. employ additional people, buy additional equipment, install additional networks, etc)?

This list is indicative only; you are free to present and discuss any tool related issues that you consider important.
The web page
(http://www.softwareqatest.com/qatweb1.html)

contains many useful links, some specifically to the sites concerned with Testing Tools. You may use these as your *starting point*, but do not limit your research to just these sources.

Scenario

You have recently joined a small software house specialising in the development of ICT solutions within the banking and financial industry. Technical aspects include data analytics, risk & compliance, quality control, complex data entry and retrieval, customer data encryption etc etc.

The software house has 10 technical staff with expertise in system design and production, database management, analytics, instrument control, decision support, knowledge of software development lifecycle and communications, project management, statistics and training.

The manager is considering the introduction of software testing tools to ensure that all of the software products produced are highly reliable and absolutely correct. You have been asked to evaluate one specific group of software tools and to provide a recommendation as to whether or not they should be purchased by the software house.

The current employees are not familiar with software testing tools, so you will need to provide a concise introduction outlining functionality, etc, of your selected group of tools.

Presentation

Your team will prepare and deliver a **6 minute presentation** on your chosen category of Software Testing Tools (Presentation date and time, During Week 12 lecture – 2nd November, 11 am – 1.00 pm). Schedule of the presentation will be chosen by a lucky draw – to provide equal opportunity to all teams. Every group will be given an opportunity to select one person to present on their behalf (presentation worth 5%).

Written summary

The submission should include a **short** summary and a discussion which addresses the points suggested above. The document length should be approximately **1,500 words** (about 2 pages of text) but it should not exceed 2,000 words. Please state the word count on the title page of your document. While preparing the document please keep in mind that it will be used by your fellow students for study and revision.

You must include as an appendix a copy of your presentation slides (6 slides per page format if possible). The slides do not count towards the word / page limit stated above.

It is likely that much of the material you use for the preparation of both your talk and the documentation is web based. It is necessary to appropriately acknowledge the use of all such material to avoid the allegation of plagiarism. It would be helpful for other students, who are going to use your document, if you could also include a list of the best web sites relevant to your tools.

Marking

A single mark will be given for both parts of this assessment. All members of the team will be given the same mark.

The following are the key factors for the assessment of the presentation:

- Topic introduced clearly and at the appropriate level;
- Key points covered clearly and logically;
- Effective use of appropriate visual materials;
- Clear recommendation.

The following are the key factors for the assessment of the written summary:

- Topic covered at the appropriate level;
- Factually correct;
- Excellent range of source materials;
- Discussion demonstrates critical analysis and the ability to evaluate evidence;
- Clear recommendation, well justified;
- Adequate acknowledgement of sources;

- Correct citation of references.

Task 2 (worth 15%): Construct two separate use cases and use case scenarios for interactions with a bank Automated Teller Machine. Do not try to capture all the functionality of the ATM into one graph; think about two different people using the ATM and what each one might do.

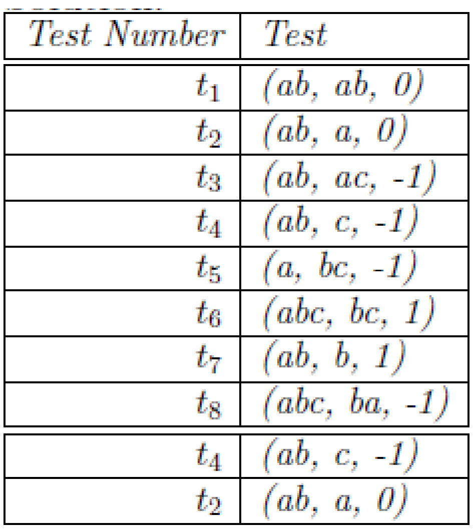
Design test cases for your scenarios.

(Hint: Follow steps as discussed in the lecture and you may also refer to:
https://www.ibm.com/developerworks/rational/library/04/r-3217/)

Task 3 (worth 15%): Consider the pattern matching example in Figure 2.21 from PO book (*TestPat* program available at https://cs.gmu.edu/~offutt/softwaretest/edition1/programs/). In particular, consider the final table of tests in Section 2.3 (also refer to PO book). Consider the variable *iSub*. Number the (unique) test cases, starting at 1, from the top of the *iSub* part of the table. For example, (*ab*, *c*,−*1*), which appears twice in the *iSub* portion of the table, should be labeled test t_4.

a) Give a minimal test set that satisfies *all defs* coverage. Use the test cases given.
b) Give a minimal test set that satisfies *all uses* coverage.
c) Give a minimal test set that satisfies *all du-paths* coverage.

(Hint: Detailed discussion leading to this task is provided in the PO book. You need to read it before attempting this task).



Test Number	Test
t_1	(ab, ab, 0)
t_2	(ab, a, 0)
t_3	(ab, ac, -1)
t_4	(ab, c, -1)
t_5	(a, bc, -1)
t_6	(abc, bc, 1)
t_7	(ab, b, 1)
t_8	(abc, ba, -1)
t_4	(ab, c, -1)
t_2	(ab, a, 0)

Task 4 (Total 35% - 15% on graph coverage plus 20% on logic coverage):
Use `greatestCommonDivisor( )` [Shown in **Figure 1**], to complete the tasks below.

(a) List the test requirements and a test set for:
  (i). All-du-path Coverage.
  (NOTE: This question may require further research – list all reasoning for the given answer and cite any relevant references).
  (ii). Prime Path Coverage.

(b) Undertake logic coverage of the given program in **Figure 1**:
  (i). List all predicates.
  (ii). Define all reachability conditions for the predicates defined in part (i).
  (iii). List test cases (and expected output) for:
    1) Predicate Coverage
    2) Clause Coverage

3) Correlated Active Clause Coverage (CACC).

(Hint: Follow the structure outlined in the PO textbook).

```java
public static int greatestCommonDivisor(int num1, int num2)
{
   List<Integer> factorsNum1 = new ArrayList<Integer>();
   List<Integer> factorsNum2 = new ArrayList<Integer>();

   for( int ii = 1; ii <= num1; ii++)
   {
      if( num1 % ii == 0 )
      {
         factorsNum1.add(ii);
      }
   }

   for( int jj = 1; jj <= num2; jj++)
   {
      if( num2 % jj == 0)
      {
         factorsNum2.add(jj);
      }
   }

   boolean found = false;
   int kk = factorsNum1.size()-1;
   int gsd = -1;

   while( !found && kk >= 0 )
   {
      int factor1 = factorsNum1.get(kk);
      int ll = factorsNum2.size() -1;
      while( !found && ll >= 0 )
      {
         if( factor1 == factorsNum2.get(ll) )
         {
            found = true;
            gsd = factor1;
         }
         ll--;
      }
      kk--;
   }

   return gsd;
}
```

Figure 1 – Method `greatestCommonDivisor( )`

NOTE:

(b) You are welcome to make assumption on any part of the Assignment (*but provide justification!*).

Deliverables

You may submit work in a group of three students. Each group need to submit ***one report*** in hardcopy on the due date ***before the commencement of the lecture*** (also include a CD/usb – submit any programs supporting your tasks).