Prof. Dr. A. Podelski,
Dr. B. Westphal
C. Schilling

Summer term 2018

Softwaretechnik/Software Engineering

http://swt.informatik.uni-freiburg.de/teaching/SS2018/swtvl

Exercise Sheet 1

Early submission: Wednesday, 2018-04-25, 12:00 Regular submission: Thursday, 2018-04-26, 12:00

Note: A good part of Software-Engineering is about communication in oral and written form. To practise this aspect, please prepare your submissions in the spirit of the discussion in Lecture 1 (each task is a tiny little scientific work; unless otherwise noted including: your understanding of the task in own words, your solution, your argument why your solution solves the task.)

Exercise 1 – Metrics (10)

1.1. Lines of Code Metrics

Consider the following lines of code (LOC) metrics:

- LOC_{tot} = Total number of lines of code
- LOC_{ne} = Number of non-empty lines of code
- LOC_{pars} = Number of lines of code that do not consist entirely of comments or non-printable characters.

(i) Calculate the value of the LOC metrics for the Java program in the file `MyQuickSort.java` accompanying this exercise sheet. (2)

(ii) The LOC metrics are often used as derived measure for the complexity or effort required to develop the code being measured.

In particular the family of LOC metrics is notorious for being subvertible. If a metric is subvertible, its value can be manipulated to increase it arbitrarily while preserving the same program semantics. I.e., for every program, there always exists a semantically equivalent program (that performs the same computation, and thus should have needed roughly similar effort to develop) that has substantially different metric values.
Convince yourself of this claim for the case of LOC_{pars}:

a) Give two semantically equivalent programs (in a high-level programming language of your choice, like Java, C++, C) with substantially (at least an order of magnitude) different metric values. (1)

b) Is your example a rare exception? If not, give a procedure to subvert given programs to a given metric value; if yes, argue why. (1)

(iii) What is the largest value of the LOC metrics for any of your contributions to a software project so far? What are the values of the LOC metrics for the whole project? (An assignment in a programming course also counts as a project.) (1)

Give just the order of magnitude; for your contribution, one of the intervals $[0, 30]$, $]30, 100]$, $]100, 300]$, $]300, 10^3]$, $]10^3, 3 \cdot 10^3]$, $]3 \cdot 10^3, 10^4]$, $]10^4, \infty[$*; and for the project* $[0, 10]$, $]10, 100]$, $]100, 1000]$, *...,* $]10^5, 10^6]$, $]10^6, \infty[$.

For example, Christian's values for LOC_{pars} *are* $(]10^3, 3 \cdot 10^3],]10^5, 10^6[)$ *(∼2,500 lines in a ∼500,000 lines framework).*

1.2. Cyclomatic Complexity

Consider the *cyclomatic complexity* or *McCabe* metric.

(i) Construct the control flow graph (CFG) for the method `quickSort` of the class `MyQuickSort` in the file accompanying this exercise sheet and calculate the value of its cyclomatic complexity as shown in the example of Fig. 1. (4)

(ii) In the example, we introduced additional auxiliary nodes to the control flow graph that serve as junction points for control paths; see, e.g., the circular node below node number 5. Another possibility of constructing the CFG would be to directly connect the nodes representing program locations. In our example, there would be a direct edge from node 5 to node 6 and from node 8 to node 6.

Does this choice of CFG construction alter the value of the cyclomatic complexity metric? Justify your answer. (1)

Program:

```
1  void insertionSort(int[] array) {
2    for (int i = 2; i < array.length; i++) {
3      tmp = array[i];
4      array[0] = tmp;
5      int j = i;
6      while (j > 0 && tmp < array[j-1]) {
7        array[j] = array[j-1];
8        j--;
9      }
10     array[j] = tmp;
11   }
12 }
```

The cyclomatic complexity is defined for graph G corresponding to the program as

$$v(G) = e - n + p.$$

Number of edges: $e = 11$
Number of nodes: $n = 6 + 2 + 2 = 10$ (nodes are marked with the corresponding line numbers)
External connections: $p = 2$

$v(G) = 11 - 10 + 2 = 3$

Corresponding graph G

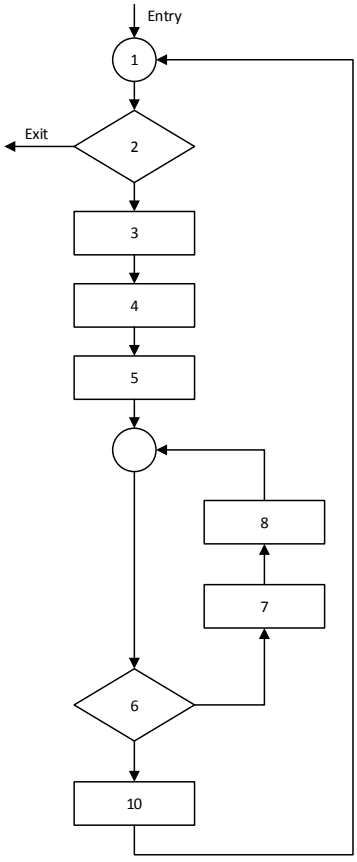

Figure 1: Example of the calculation of cyclomatic complexity