Technische Universität Wien
Programmiersprachen und Übersetzer
Prof. Dr. J. Knoop

A-1040 Wien
Argentinierstr. 8
Tel.: 01-58801-18510

"Analyse und Verifikation (185.276, VU 2.0, ECTS 3.0)" SS 2020

Übungsblatt 1 04.03.2020

Aufgabe 1 : (3∗4 Punkte)

Pentium-Bug, Marssonde Mars Climate Orbiter, Ariane-Absturz sind Beispiele auch einer breiten Öffentlichkeit bekannt gewordene durch Hard- und Software verursachtes Fehlverhalten von Systemen. Recherchieren Sie für diese oder andere (insgesamt drei) Beispiele Hintergründe und Ursachen. Geben Sie jeweils in Stichpunkten gesicherte, vermutliche oder vermutete Fehlerursache an, den verursachten Schaden, vorgenommene Abhilfen und die Fundstelle(n) ihrer Rechercheinformationen an.

Aufgabe 2 : (3∗2 Punkte)

Welche Dienstleistungen oder/und Werkzeuge bieten AbsInt, Coverity und Rapita ihren Kunden im Bereich von Analyse und Verifikation an? Nennen Sie jeweils einige Beispiele.

Aufgabe 3 : (5∗2 Punkte)

Im Artikel *"A Few Billion Lines of Code Later: Using Static Analysis to Find Bugs in the Real World"* beschreiben die Autoren ihre Erfahrungen, einen Forschungsprototypen zur Fehlersuche in C-Programmen zu einem industriellen Ansprüchen genügenden kommerziellen Produkt zu machen.

Lesen Sie den Artikel und beantworten Sie in knapper Form folgende Fragen:

1. "Ein Werkzeug, das Fehler in Produktionscode findet, ist ein gutes Werkzeug. Ein Werkzeug, das mehr Fehler findet, ist ein besseres Werkzeug."

   Wie wird diese Aussage aus einer wissenschaftlichen, wie aus einer kommerziellen Perspektive im Artikel beurteilt? Welche Gründe werden dafür, welche dagegen angeführt?

2. *Sound* bedeutet korrekt. In welchem Sinn wird der Begriff *sound* im Zusammenhang mit dem Fehlerfindewerkzeug der Fa. Coverity verwendet?

3. Was bedeuten die Begriffe *false positives* und *false negatives* im Zusammenhang mit dem Fehlerfindewerkzeug der Fa. Coverity?

4. Wie können sich *false positives* und *false negatives* auf die Akzeptanz eines Fehlerfindewerkzeugs auswirken und warum?

5. Im Artikel heißt es (s.S. 70 unten): "The C language does not exist; neither does Java, C++, and C#". Was ist damit gemeint? Welche Probleme ergeben sich daraus für die Kommerzialisierung eines Forschungswerkzeugs wie im Fall der Fa. Coverity?

Aufgabe 4 : (5∗2 Punkte)

Im Artikel *"Lessons from Building Static Analysis Tools at Google"* beschreiben die Autoren ihre Erfahrungen mit der Entwicklung und dem Einsatz statischer Code-Analysewerkzeuge bei Google.

Lesen Sie den Artikel und beantworten Sie in knapper Form folgende Fragen:

1. Welche Gründe können in der Praxis dazu führen, dass Analysewerkzeugen nicht vertraut wird oder sie nicht verwendet werden? Decken sich die angeführten Gründe mit denen im Artikel von Al Bessey et al. aus Aufgabe 3?

2. Was sind typische Analysen, die bei Google durchgeführt werden, bevor Code produktiv gestellt wird?

3. Wie wird die Spannung zwischen Analysekomplexität und -stärke auf der einen Seite und Skalierbarkeit auf der anderen Seite aufgelöst?

4. Welche Bedeutungsvarianten von "false positive" werden unterschieden? Gibt es hierbei Unterschiede im Gebrauch zum Artikel von Al Bessey et al. aus Aufgabe 3?

5. Konkrete Erfahrungen werden zu Einsatz und Integration des Werkzeugs *FindBugs* berichtet (s.S. 61, linke Spalte). Gibt es hier Gemeinsamkeiten zu den Erfahrungen, die im Artikel von Al Bessey et al. aus Aufgabe 3 berichtet werden?

Hinweis: Sie können die Artikel

- Al Bessey, Ken Block, Ben Chelf, Andy Chou, Bryan Fulton, Seth Hallem, Charles Henri-Gros, Asya Kamsky, Scott McPeak, Dawson Engler. *A Few Billion Lines of Code Later: Using Static Analysis to Find Bugs in the Real World.* Communications of the ACM 53(2):66-75, 2010.

- Caitlin Sadowski, Edward Aftandilian, Alex Eagle, Liam Miller-Cushon, Ciera Jaspan. *Lessons from Building Static Analysis Tools at Google.* Communications of the ACM 61(4):58-66, 2018.

aus dem TU-Netz heraus in der ACM Digital Library (`www.acm.org/dl`) herunterladen.

Abgabe: Mittwoch, den 11.03.2020, vor der Vorlesung.