

Software-Qualität: Übung 2

Debugging

Program Slicing
Hypothesen
Reproduzierbarkeit von Fehlern



University of Zurich
Department of Informatics



Inhalt

Ziele

- § Übung 2 erklären

Agenda

- § Zielsetzung und Ablauf der Übung
- § Aufgaben

Zielsetzung

- § Wichtige Debugging-Ansätze anwenden können
 - § Program Slicing
 - § Erstellen von Fehlerhypothesen
 - § Umgang mit Hypothesen über Programmzustände
- § Einflussfaktoren auf Reproduzierbarkeit von Fehlern verstehen

Administratives

§ Ablauf

§ 7. Mai: Ausgabe der Übung und Einführung

§ 14. Mai: Fragemöglichkeiten (Voranmeldung)

§ 18. Mai, 18:00 CET: Deadline für Abgabe

§ Form

§ Individuell, paarweise, oder zu dritt (Ausnahme)

§ Infrastruktur

§ Werkzeug: Büroanwendungen und Eclipse

§ Fragen: Uniboard Diskussionsforum und
cramer@ifi.unizh.ch

§ Abgabe: cramer@ifi.unizh.ch und fricker@ifi.unizh.ch

Aufgaben

- § Aufgabe A:
Program-Slicing
- § Aufgabe B:
Fehlerhypothesen
- § Aufgabe C:
Reproduktion von Fehlern
- § Aufgabe D:
Hypothesen über Programmzustände

Teil A: Program Slicing

§ Gegeben kurzes C-Programm

§ Aufgabe

§ a) Set-Use, Use-Set, Set-Set Beziehungen

§ b) Program Dependency Graph (PDG)

§ c) Backward- und Forward-Slice

§ d) Probleme, welche mit PDG entdeckt werden können

Teil A: Set-Use, Use-Set, Set-Set Beziehungen

Programm

```
§ 06  inword = NO;  
    . . .  
17    if ( inword == NO )
```

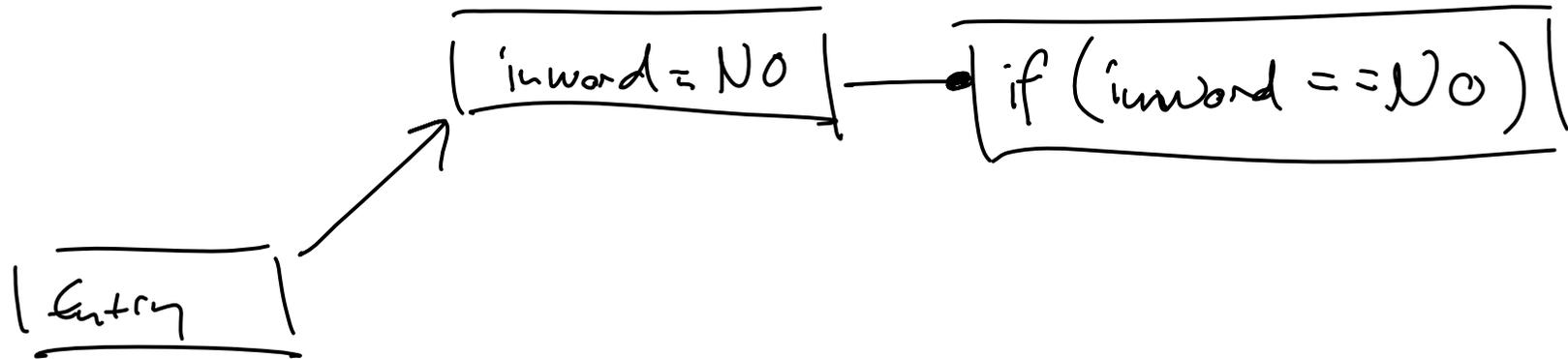
Set-Use Beziehung

§ 6 – 17: (inword=NO;) – (if (inword==NO))

Ø Alle möglichen Kombinationen von Beziehungen erarbeiten

Ø jedoch nicht Set-Set-Use, etc.

Teil A: Program Dependency Graph



→ Kontrollabhängigkeit (Steuerfluss)
—• Datenabhängigkeit (Datenfluss)

Wegweisen Sie alle Beziehungen in einem einzigen Graphen.

Teil A: Statische Slices

Slices

§ Programmcode kopieren

§ Erreichte Zeilen: stehen lassen

§ Nicht erreichte Zeilen: durchstreichen

∅ Verwenden Sie den PDG

Teil B: Fehlerhypothesen

§ Gegeben NumberTranslator.jar

§ Input: 21 → Output „twenty one“

*Achtung: Integer-Explos
kann *.jar zu *.zip
ändern.
→ einfach umbenennen.*

§ Aufgabe

§ Fehler finden und beschreiben, welche sind in NumberTranslator.jar eingeschlichen hat

§ Komplette Historie der Diagnosen abgeben

∅ Diagnose

§ Hypothese:
Zehnerziffer wird dekrementiert

§ Testfälle (Input → Output):
12 → 2, 23 → 13, 45 → 35

§ Testresultate (Input → Output):
12 → 12, 23 → 23, 45 → 45

∅ Hypothese soll verworfen werden

Teil C: Reproduktion von Fehlern

§ Was sind Einflussfaktoren auf die Reproduzierbarkeit von Fehlern?

§ 2 Beispiele von Faktoren

Beispiel eines vom Faktor abhängigen Fehlers
Reproduktion des Fehlers

§ Bug Story „Program only works on Wednesday“

§ Faktor: Wochentage (Zeit) und die Länge deren Namen

§ Fehler: Benutzereingabe wird an allen Wochentagen ausser am Mittwoch falsch interpretiert

§ Reproduktion: Systemzeit auf einen nicht-Mittwoch setzen

Teil D: Hypothesen über Programmzustände

§ Gegeben

- § Implementation einer Variante von Quicksort
- § Input, welcher einen Fehler verursacht

§ Aufgabe

- § a) Einfachsten Input finden, der den Fehler proviziert
- § b) Hypothese über korrekten Verlauf des Programmzustandes definieren
- § c) Vergleichen des Verlaufs des Ist-Programmzustandes mit der Hypothese
- § d) Defektlokalisierung und Korrekturvorschlag

Teil D: Hypothesen über Programmzustände

Programmzustandsbeschreibung

§ Debug-Punkt: [Zeilennummer]

§ Programmzustand: Werte der Variablen
start, end, untergrenze,
obergrenze, vergleiche mit,
elemente

Ø Zeigen Sie tabellarisch, wie sich der
Programmzustand verändern soll, bzw.
dies auch tut → Unterschiede!

Teil D: Debugging mit Eclipse

1) Rechte Maustaste
→ Kontextmenü

2) "Toggle Breakpoint"

```
1 public class HelloWorld {
2
3
4     private String message;
5     public HelloWorld(String msg) {
6         message=msg;
7     }
8     public void sayHello() {
9         System.out.println(getMessage());
10    }
11    private String getMessage() {
12        return message;
13    }
14    public static void main(String[] args) {
15        HelloWorld h = new HelloWorld("Hello World");
16        h.sayHello();
17    }
18 }
```

Teil D: Debugging mit Eclipse

1) Rechte Maustaste
→ Kontextmenü

2) Anwendung in Debug-Modus
starten

```
public class HelloWorld {  
    private String message;  
    public HelloWorld(String msg) {  
        message=msg;  
    }  
    public void sayHello() {  
        System.out.println(getMessage());  
    }  
    public String getMessage() {  
        return message;  
    }  
    public static void main(String[] args) {  
        HelloWorld hw=new HelloWorld("Hello World");  
        hw.sayHello();  
    }  
}
```

report, please visit:
report/crash.jsp

Teil D: Debugging mit Eclipse

The screenshot shows the Eclipse IDE in a debug state. The main editor displays the source code of `HelloWorld.java` with a breakpoint set at line 9, `System.out.println(getMessage());`. The `Variables` window is open, showing the current state of the program. The variable `this` is listed with the value `HelloWorld (id=10)`. Handwritten red annotations highlight the breakpoint and the variable information.

Name	Value
this	HelloWorld (id=10)

erreichtes Breakpoint

Variablen

Variabel-Werte

Teil D: Debugging mit Eclipse

The screenshot shows the Eclipse IDE in a debug state. The main window displays the 'HelloWorld' application, which is suspended at line 21 of the main method. The 'Variables' view on the right shows the following data:

Name	Value
this	HelloWorld (id=10)
message	"Hello World"
count	11
hash	0
offset	0
value	char[11] (id=322)
[0]	H
[1]	e
[2]	l
[3]	l
[4]	o
[5]	
[6]	W
[7]	o
[8]	r
[9]	l
[10]	d

Handwritten red annotations are present:

- An arrow points from the text "einfache Variable" to the "message" field in the Variables view.
- A bracket groups the "value" field and its elements, with the text "Array" written below it.

Viel Glück!

§ Ablauf

§ 7. Mai: Ausgabe der Übung und Einführung

§ 14. Mai: Fragemöglichkeiten (Voranmeldung)

§ 18. Mai, 18:00 CET: Deadline für Abgabe

§ Form

§ Individuell, paarweise, oder zu dritt (Ausnahme)

§ Infrastruktur

§ Fragen: Uniboard Diskussionsforum und cramer@ifi.unizh.ch

§ Abgabe: cramer@ifi.unizh.ch und fricker@ifi.unizh.ch

§ Gesamtaufwand

§ ca. 10h

§ Zeitaufschreibung machen und Probleme **frühzeitig** melden