WS 2016/17

Simon Kaiser, 354692
Philipp Hochmann, 356148
Felix Kiunke, 357322
Giacomo Klingen, 356778
Daniel Schleiz, 356092

Softwaretechnik
Aufgabenblatt 7

Aufgabe 7.1

a)

Ohne Betrachtung des Programmcodes wäre das Black-Box Verfahren eine geeignete Weise zum Testen. In dem Rahmen möchte man sich möglichst geschickt Eingabedaten überlegen, womit man die Methode testen könnte. Dazu ist es sinnvoll, sich Äquivalenzklassen für die Eingabedaten zu entwickeln, um mit so wenig wie möglich Eingaben die Methode so umfangreich wie es geht zu testen. Die Äquivalenzklassen von den Eingabedaten für diese Methode könnten wie folgt aussehen:

- $x < 0$
- $x > 0$
- $x = 0$

Insbesondere sollte die 0 geprüft werden, da unter Umständen in der Methode Divisionen vorkommen könnten, wobei 0 als Nenner problematisch wäre. Außerdem sollte man noch explizit Integer.MAX_VALUE und Integer.MIN_VALUE testen, da die Methode arithmetische Operationen verwenden könnte, welche Überläufe ausgelöst werden könnten, was das Ergebnis verfälschen würde. Da außerdem auch null in die Methode übergeben werden könnte, sollte geprüft werden, wie die Methode damit umgeht.

b)

- $x < 0$: Eingabemenge: { Integer.MIN_VALUE, -49783, -14, -1}. Neben dem oben genannten Randfall Integer.MIN_VALUE sollte auch noch -1 als Randfall betrachtet werden, da eventuell arithmetische Operationen durchgeführt werden, wodurch die Eingabe in den positiven Bereich rutscht. Außerdem sollen mit -14 und -49783 noch ein zufälliger “kleiner” und “großer” Wert getestet werden, da nur Randfallbetrachtungen nicht ausreichen.

- x > 0: Eingabemenge: { 1, 15, 59285, Integer.MAX_VALUE}. Neben dem oben genannten Randfall Integer.MAX_VALUE sollte auch noch 1 als Randfall betrachtet werden, da eventuell arithmetische Operationen durchgeführt werden, wodurch die Eingabe in den negativen Bereich rutscht. Außerdem sollen mit 15 und 59285 noch ein zufälliger "kleiner" und "großer" Wert getestet werden, da nur Randfallbetrachtungen nicht ausreichen.
- x = 0: Eingabemenge: { 0 }.

Zudem sollte auch noch null als Eingabe getestet werden.

Aufgabe 7.2

Aufgabe 7.3

a)

Der Zweigüberdeckungstest ist stärker, da nicht nur alle Anweisungen ausgeführt werden, sondern auch alle möglichen Entscheidungen (false & true) durchlaufen werden. Hier kann sich herausstellen, dass manche Verzweigungen nie benutzt werden und somit eine unnötige Abfrage darstellen.

b)

Die folgende Funktion soll das Vorzeichen des übergebenen Parameters berechnen. -1 für negativ, +1 für positiv, 0 für 0.

```
1:    public static int sign(int x){
2:        if(x<0){
3:            return -1;
4:        } else {
5:            return 1;
6:        }
7:    }
```

c)