















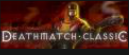




Anwendung: Äquivalenzklassen- bildung

← → STORE LIBRARY		STEAMOS + LINUX		VIEW	
GAMES		★	☁	STATUS	METASCORE
					LAST PLAYED
	Amnesia: The Dark Descent			Not installed	85
	Antichamber			Not installed	82
	Aquaria			Not installed	82
	Audiosurf 2		☁	Not installed	76
	The Banner Saga		☁	Not installed	80
	Bastion		☁	Not installed	86
	BioShock Infinite		☁	Not installed	94
	Borderlands 2		☁	Not installed	89
	Braid		☁	Not installed	90
	Broforce		☁	Not installed	83
	Chaos on Deponia		☁	Not installed	78
	Counter-Strike			Not installed	88
	Counter-Strike: Global Offensive		☁	Not installed	83
	Counter-Strike: Source		☁	Not installed	88
	Darkest Dungeon		☁	Not installed	84
	Day of Defeat			Not installed	79
	Dead Island		☁	Not installed	80
	Deathmatch Classic			Not installed	New
	Deponia		☁	Not installed	74

Ausgangslage

Aufbau und Funktion eines Lastenhefts – Beispiel 9(10)

6. Qualitätsanforderungen:

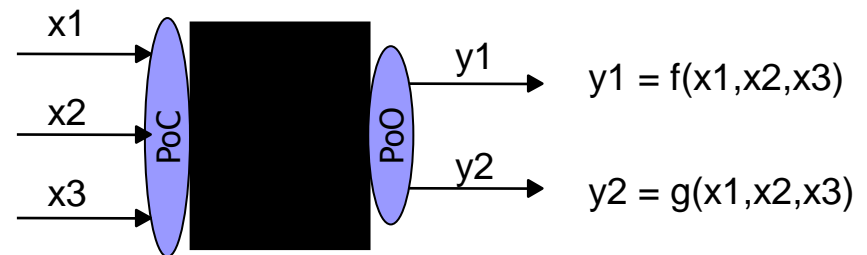
Produktqualität	sehr gut	gut	normal	irrelevant
Funktionalität		x		
Zuverlässigkeit	x			
Benutzbarkeit	x	x		
Effizienz			x	
Änderbarkeit			x	
Portierbarkeit				x

Die Benutzbarkeit der Funktionen mit dem die Benutzer arbeiten müssen sehr gut sein, da ein breites Spektrum an Nutzern angesprochen wird. Die Benutzbarkeit aller übrigen Funktionen muss gut sein, da sie von einem kleineren Klientel bedient wird (Spielehersteller und VAULT-Mitarbeiter).

7. Ergänzungen (wie z.B. Abgrenzungskriterien):
Buchhaltungsfunktionen (z.B. Erstellung von Abrechnungen zwischen VAULT und Spieleherstellern) gehören nicht zum Leistungsumfang.

13

Requirements
(Spezifikation)



Black-Box-Testing

Spezifikation

- DAMPF besitzt auch die Möglichkeit für den Spielehersteller seine Spiele hochzuladen.
- Dafür gibt es eine Funktion im GUI, welche folgende Eingabeparameter erwartet:
 - Titel des Spiels
 - Mindestens 2, maximal 20 Zeichen und nur Großbuchstaben und Leerzeichen
 - Altersbeschränkung (ab welchem Alter freigegeben)
 - Eine Zahl zwischen (inklusive) 0 und 18

Wiederholung: Äquivalenzklassenbildung

- Schritt 1: Aufstellung von Eingabebedingungen
- Schritt 2: Bildung von Äquivalenzklassen
- Schritt 3: Äquivalenzklassen identifizieren
- Schritt 4: Testfälle definieren (gültige Ä-Klassen)
- Schritt 5: Testfälle definieren (ungültige Ä-Klassen)

Äquivalenzklassenbildung

Eingabebedingungen definieren & Ä-Klassen finden

Schritt 1: Aufstellung von Eingabebedingungen

Schritt 2: Bildung von Äquivalenzklassen

Schritt 3: Äquivalenzklassen identifizieren

Eingabebedingungen	Gültige Ä-Klasse	Ungültige Ä-Klasse
Anzahl der Parameter	2 (v1)	Keiner (i1), Einer (i2), >2 (i3)
Titel (Bezeichner)	A-Z, Leerzeichen(v2)	Sonstige Zeichen (i4)
Titel (Länge)	$2 \leq x \leq 20$ (v3)	<2 (i5), >20 (i6)
Alter (Bezeichner)	Ziffern (v4)	Sonstige Zeichen (i7)
Alter (Wert)	$0 \leq x \leq 18$ (v5)	<0 (i8), >18 (i9)

Äquivalenzklassenbildung

Testfälle bestimmen

Schritt 4: Testfälle definieren (gültige Ä-Klassen)

Schritt 5: Testfälle definieren (ungültige Ä-Klassen)

- Gültige Ä-Klassen

- „SUPER MARIO“, 3 (v1,v2,v3,v4,v5)

- Ungültige Ä-Klassen

- (i1)
 - „VALID“, (i2)
 - „VALID“, 3, „VALID“ (i3)
 - „#lame“, 3 (i4)
 - „B“, 3 (i5)
 - „SUPER LONG MEGA COOL TITLE“, 3 (i6)
 - „VALID“, „INVALID“ (i7)
 - „VALID“, -2 (i8)
 - „VALID“, 19 (i9)

Äquivalenzklassenbildung

Testfälle bestimmen

Schritt 4: Testfälle definieren (gültige Ä-Klassen)

Schritt 5: Testfälle definieren (ungültige Ä-Klassen)

■ Gültige Ä-Klassen

□ „SUPER MARIO“, 3 (v1,v2,v3,v4,v5)

Eingabebedingungen	Gültige Ä-Klasse	Ungültige Ä-Klasse
Anzahl der Parameter	2 (v1)	Keiner (i1), Einer (i2), >2 (i3)
Titel (Bezeichner)	A Z, Leerzeichen(v2)	Sonstige Zeichen (i4)
Titel (Länge)	$2 \leq x \leq 20$ (v3)	<2 (i5), >20 (i6)
Alter (Bezeichner)	Ziffern (v4)	Sonstige Zeichen (i7)
Alter (Wert)	$0 \leq x \leq 18$ (v5)	<0 (i8), >18 (i9)

Äquivalenzklassenbildung

Testfälle bestimmen

Eingabebedingungen	Gültige Ä-Klasse	Ungültige Ä-Klasse
Anzahl der Parameter	2 (v1)	Keiner (i1), Einer (i2), >2 (i3)
Titel (Bezeichner)	A Z, Leerzeichen (v2)	Sonstige Zeichen (i4)
Titel (Länge)	$2 \leq x \leq 20$ (v3)	<2 (i5), >20 (i6)
Alter (Bezeichner)	Ziffern (v4)	Sonstige Zeichen (i7)
Alter (Wert)	$0 \leq x \leq 18$ (v5)	<0 (i8), >18 (i9)

■ Ungültige Ä-Klassen

- (i1)
- „VALID“, (i2)
- „VALID“, 3, „VALID“ (i3)
- „#lame“, 3 (i4)

- „B“, 3 (i5)
- „SUPER LONG MEGA COOL TITLE“, 3 (i6)
- „VALID“, „INVALID“ (i7)
- „VALID“, -2 (i8)
- „VALID“, 19 (i9)

Äquivalenzklassenbildung

Codierung der Testfälle

```
public class SpielTest {  
  
    @Test  
    public void testValidInput() {  
        Spiel spiel = new Spiel( titel: "SUPER MARIO", alter: 3);  
        assertEquals( expected: "SUPER MARIO", spiel.getTitel());  
        assertEquals( expected: 3, spiel.getAlter());  
    }  
  
    @Test(expected = InvalidTitel.class)  
    public void testInvalidTitel() {  
        Spiel spiel = new Spiel( titel: "b", alter: 3);  
    }  
  
    @Test(expected = InvalidAlter.class)  
    public void testInvalidAlter() {  
        Spiel spiel = new Spiel( titel: "VALID", alter: -2);  
    }  
}
```