University of Basel

# Software Engineering

Marcel Lüthi, Universität Basel

# Whitebox testing

# Whitebox testing

- Ableiten von Testfällen anhand interner Struktur
  - Struktur vom Code definiert Äquivalenzklassen.

- Spezifikation nicht im Vordergrund um Testfälle zu finden
  - Testfälle hängen von Implementation ab

**Überdeckungskriterien**

- Anweisungsüberdeckung
- Zweig/Bedingungsüberdeckung
- Pfadüberdeckung

```
int gcd(int x, int y) {

  while (x != y) {
      if (x > y) {
          x = x – y;
      } else {
          y = y – x;
      }
  }
  return x
}
```

# Anweisungsüberdeckung

Wähle Testmenge $T$ so, dass jedes Statement in Programm $P$ mindestens einmal ausgeführt wird (für beliebiges $d \in T$)

```
int example(int x, int y) {
    int res = 0;

    if (x > 0) {
        res = 1;
    } else {
        res = 2;
    }

    if (y > 0) {
        res = 3;
    } else {
        res = 4;
    }
    return res;
}
```

Vollständige Abdeckung
{
  (x=2, y=3),
  (x=-13, y =51),
  (x=97, y =17),
  (x = -1, y = -1)
}

Minimale Testmenge mit vollständiger Abdeckung
{
  (x = -13, y =51),
  (x=2, y = -3)
}

# Beispiel: Euklid's Algorithmus

```
int gcd(int x, int y) {

  while (x != y) {
      if (x > y) {
          x = x – y;
      } else {
          y = y – x;
      }
  }
  return x
}
```

Minimale Testmenge

{
  (x=3, y=2)
}

# Bedingungsüberdeckung

Die Testmenge $T$ wird so gewählt, dass jeder Zweig des Kontrollflusses mindestens einmal durchlaufen wird und alle möglichen Elemente von zusammengesetzten Bedingungen mindestens einmal aktiv sind.

```
int example(int x, int y) {
  if (x < 0 && y < 0) {
    return 0
  } else {
    return 1;
  }
}
```

Minimale Testmenge

{
  x=2, y=2;
  x=-2, y=2;
  x=2, y = -3
}

# Pfadüberdeckung

Die Testmenge $T$ wird so gewählt, dass jeder Pfad im Programm einmal durchlaufen wird

```
int example(int x, int y) {
    int res = 0;

    if (x > 0) {
        res = 1;
    } else {
        res = 2;
    }

    if (y > 0) {
        res = 3;
    } else {
        res = 4;
    }
    return res;
}
```

Minimale Testmenge
{
(x= 1, y= 1),
(x= -1, y = 1),
(x= 1, y = -1),
(x= -1, y = -1)
}

- Bestes Testkriterium, aber nicht praktikabel
  - Explosion der Anzahl Pfade bei langen Programmen oder Loops

*Auch perfekte Überdeckung kann nicht verhindern, dass Teile der Spezifikation nicht implementiert sind*