Programm- & Systemverifikation

Assignment 2

Georg Weissenbacher

184.741

Part 1 of Assignment 2

- Derive test cases for the insertion function of a **balanced (AVL) binary search tree**.

Part 1 of Assignment 2

- Derive test cases for the insertion function of a **balanced (AVL) binary search tree**.
- Use the following techniques:
  a) Equivalence class partitioning
  b) Boundary value testing

Signature

```
/* recursive tree structure                    */
typedef struct _tree
{
  struct _tree * left;
  struct _tree * right;
  int element;
  int height;
} Tree;
```

`insert(int e, Tree *t)`: Insert element `e` into the tree `t`

Note:

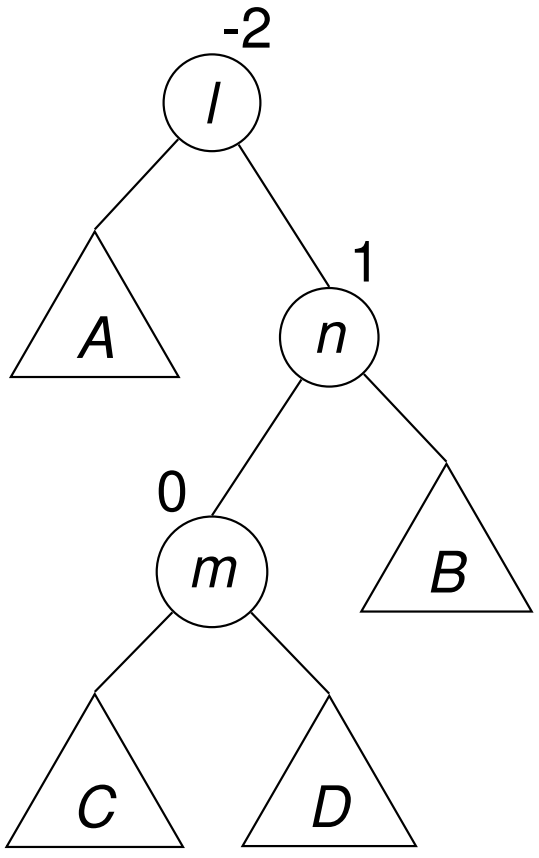
- You don't know the concrete implementation
- But you know how an AVL is supposed to work

Equivalence Classes for Inputs

Remember: Tree `t` is an input, too!

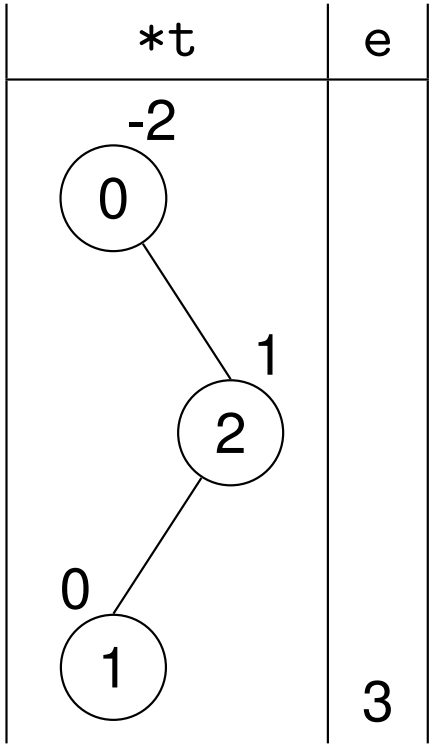
- Balanced: $|\text{left height} - \text{right height}| \leq 1$
- Elements in left sub-tree are smaller than elements in right sub-tree

1. Derive equivalence classes:
   - based on balance
   - number of elements
   - content
   - ...
2. Justify *and* illustrate your equivalence classes (see right).
3. Use table (as in lecture) to list your equivalence classes



Boundary Value Testing

1. Derive test cases using boundary value testing:
   - cover all equivalence classes (valid, invalid)
   - take outputs into account
2. Illustrate your test cases (see right)
3. Use table (as in lecture) to list your test cases



Submitting your solution

- Part 2 is a “pencil and paper” assignment, available separately
- Your solution must be handed in via TUWEL on April 30, 4pm
  - Use the file names `part1.pdf` and `part2.pdf`
    - There’s a scanner in the main library
  - Late submissions will not be accepted
- Make sure the solution shows your student ID and your name.