Security Engineering
Problem Session 2
WS2018/19

Prof. Stefan Lucks, Nathalie Jolanthe Dittrich
<first>.<middle>.<lastname>(at)uni-weimar.de

Bauhaus-Universität Weimar

Nov 09, 2018

No Plagiarism!

	Total
Students	24
Submissions	12
Plagiarism (Groups)	4
Plagiarism (Students)	6
Not working code (Groups)	3
Not working code (Students)	6

Consequences

- To gain admission to do the exam up from the third Problem Set you need to get at least 25 % of the points.
  - If you gain less than 25% in one Problem Set you need to have at least 50% in another one.

Submission Guidelines (I)

- If your code does not compile, it will be graded with 0 points
  - The proper file formats for ada code are: *.adb and *.ads. Keep in mind that you cannot use them equivalently! **Other file formats cannot be compiled by a standard ada compiler hence they will be graded with 0 points!**
- If you have to answer a theoretical question (like Task 4 from Problem Set 1), submit your answer either as *.pdf or as comments in your code. Other file formats will not be accepted which will lead to 0 points (This holds especially for *.docx!).
- **Plagiarism will not be tolerated!**
  - If you are caught cheating once, the corresponding tasks will be graded with 0 points. If you are caught cheating twice, you will be excluded from the course!
  - You can ask other groups for help or discuss the assignment with then, **but** you have to mention that in your submission, otherwise this is plagiarism and will have consequences.

Submission Guidelines (II)

- I will not only grade the functionality of your code but also the style. (`http://en.wikibooks.org/wiki/Ada_Style_Guide`)
- I don't want you to just produce code. I want you to produce **good code**.
  - Not just errors but also warnings will lead to a deduction of points.

Section 1

Miniprojects

Mini Projects

Section 2

Testgen

testgen

- Small testing framework

testgen

- Small testing framework
- Write *test driver* for package:
  `http://www.uni-weimar.de/cms/fileadmin/medien/medsicherheit/Teaching/SS07/SEfSVS07/programs/testgen.zip`

testgen

- Small testing framework
- Write *test driver* for package:
  `http://www.uni-weimar.de/cms/fileadmin/medien/medsicherheit/Teaching/SS07/SEfSVS07/programs/testgen.zip`
- Requires `tg` by André Spiegel:
  `http://www.iste.uni-stuttgart.de/se-old/links/links-se/test/tg.html`

testgen (cont'd)

```
Context
   with Ada.Numerics.Elementary_Functions;
   with Ada.Text_IO;
   with Vectors;
   use Ada.Numerics.Elementary_Functions;
   use Ada.Text_IO;
   use Vectors;

***** Test Add
Define  L: constant Vector := (2, 3, 4);
       R: constant Vector := (5, 6, 7);
       Actual: Vector;
       Expected: constant Vector := (L.X + R.X, L.Y + R.Y, L.Z + R.Z);
Test    Actual := L + R;
Pass    Actual = Expected
```

- Write test driver: `vectors_tests.ts`
- Compile to Ada test code: `vectors_tests.adb`
- Compile and run Ada code

Why Two Testing Frameworks?

	`testgen`	`AUnit`
Ease-of-use	✓	–
Little Overhead	✓	–
Learnability	✓	(✓)
Exceptions	✓	✓
Documentation	(✓)	✓
Scalability	–	✓

- Decide yourself what is optimal for your case

Why Two Testing Frameworks?

	`testgen`	`AUnit`
Ease-of-use	✓	–
Little Overhead	✓	–
Learnability	✓	(✓)
Exceptions	✓	✓
Documentation	(✓)	✓
Scalability	–	✓

- Decide yourself what is optimal for your case
- Note: Separate your source code from your tests
  - Make `src` and `tests` directories

Questions?