

Summary

Unitgrade: Autograding solution used in 02465:

- Shallow extension of unittests
- Student experience
- Requires less work to set up than codejudge
 - Vedranas exam from introduction to programming
 - Alcestes exam from the CPP course and exercise 6

CMU Autolab: Platform used at CMU and many other universities

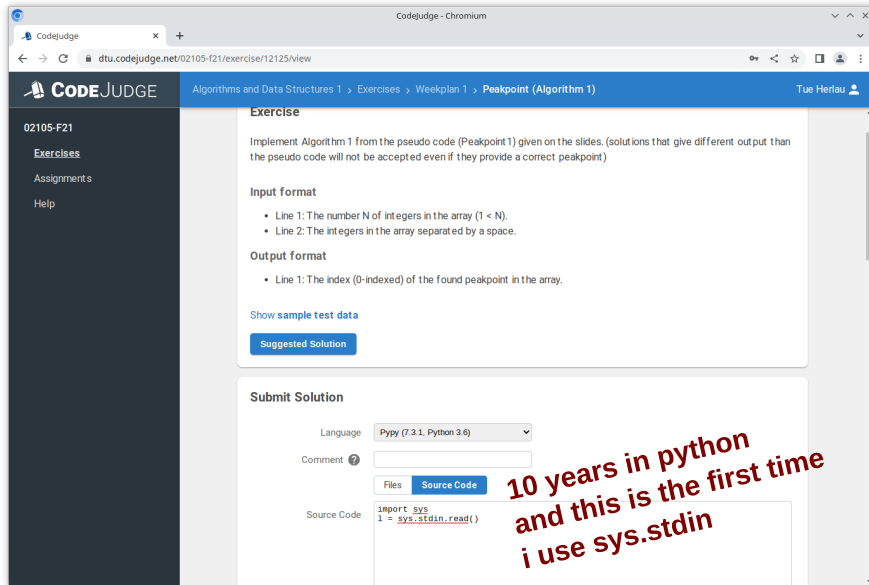
- Functions like a container for tests
- Fully text-configured

• <https://gitlab.compute.dtu.dk/tuhe/unitgrade>

• Videos: <https://youtu.be/jC9AzZA5FcQ>

• <https://autolabproject.com/>

Unitgrade: what got me started



CodeJudge - Chromium

dtu.codejudge.net/02105-f21/exercise/12125/view

CODEJUDGE Algorithms and Data Structures 1 > Exercises > Weekplan 1 > Peakpoint (Algorithm 1) Tue Herlau

Exercise

Implement Algorithm 1 from the pseudo code (Peakpoint1) given on the slides. (solutions that give different output than the pseudo code will not be accepted even if they provide a correct peakpoint)

Input format

- Line 1: The number N of integers in the array ($1 < N$).
- Line 2: The integers in the array separated by a space.

Output format

- Line 1: The index (0-indexed) of the found peakpoint in the array.

Show [sample test data](#)

[Suggested Solution](#)

Submit Solution

Language: Pypy (7.3.1, Python 3.6)

Comment:

Files [Source Code](#)

Source Code

```
import sys
l = sys.stdin.read()
```

**10 years in python
and this is the first time
i use sys.stdin**

Online autograders

- They do many things
 - Exercise descriptions
 - Administrative tasks (student onboarding, grading, exports)
 - Calendar-tasks (due-date, handling delays)
 - Assignment handin
 - Plagiarism checking
 - An IDE
 - A CI/CD system (upload code and run)
 - Various forms of automation/ideas about organizing exercises
 - A test system

My goals

User experience: **Best** allow students to solve their programming problems

Administrative ease: Make the course **software**, **tests**, and **written material** easy to write and maintain

User experience

- Which test system best allows **students** to fix problems in **their** homework?

User experience

- Which test system best allows **students** to fix problems in **their** homework?
- Which test system best allow **me** to fix problems in **my code**?

User experience

- Which test system best allows **students** to fix problems in **their** homework?
- Which test system best allow **me** to fix problems in **my code**?
- **Web-based tests**
 - No debugger
 - Local/remote code (or use the bad online IDE)
 - Increased run-time (can tests be run in isolation?)
 - Blackboxing (environment, packages, file locations, how code is called)
 - Drivers
 - Preprocessors
 - Postprocessors
 - Hacks (cat, fix-floats, etc.)
 - Adopt conventions from the tool (is reading from stdin optimal?)

User experience

- Which test system best allows **students** to fix problems in **their** homework?
- Which test system best allow **me** to fix problems in **my code**?
- **Web-based tests**
 - No debugger
 - Local/remote code (or use the bad online IDE)
 - Increased run-time (can tests be run in isolation?)
 - Blackboxing (environment, packages, file locations, how code is called)
 - Drivers
 - Preprocessors
 - Postprocessors
 - Hacks (cat, fix-floats, etc.)
 - Adopt conventions from the tool (is reading from stdin optimal?)
- **Unittests:**
 - Best-practice for 25 years
 - Use your favorite IDE: Debugger + autocomplete + inline documentation
 - Test are runnable in isolation
 - Transparency; everything is python
 - Everything can be unittested
 - Student learns relevant skills (unittesting)

Example: Intro to python exam

Setup: pip install unittest

Assignment 1 Water height

The height of the water in a pond changes daily governed by two contributions: the constant decrease of height due to the water outflow and the variable increase of height due to the rain. Given the water height for one day and the rain value r , the water height for the next day can be computed as

$$h_{\text{today}} = h_{\text{yesterday}} + r - 2.$$

If the computed value is negative it should be replaced by 0, indicating that the pond is empty. The computation can be repeated for any number of days, as long as we have rain values.

■ Problem definition

Write a function `water_height` which takes an initial water height h_0 and a vector \mathbf{r} with rain values for a number of days as input. The function should return a water height after the days have passed. The function should also work if \mathbf{r} contains only one or no days. (In case of no days, h_0 should be returned.)

■ Solution template

```
def water_height(h0, r):  
    # insert your code  
    return h
```

Input

h_0 A non-negative number with initial water height.
 \mathbf{r} A vector with rain values (non-negative numbers) for a number of days. A vector may contain multiple, only one, or no elements.

Output

h The water height after the number of days has passed.

Example: Writing your own test

Assignment 1 Water height

The height of the water in a pond changes daily governed by two contributions: the constant decrease of height due to the water outflow and the variable increase of height due to the rain. Given the water height for one day and the rain value r , the water height for the next day can be computed as

$$h_{\text{today}} = h_{\text{yesterday}} + r - 2.$$

If the computed value is negative it should be replaced by 0, indicating that the pond is empty. The computation can be repeated for any number of days, as long as we have rain values.

■ Problem definition

Write a function `water_height` which takes an initial water height h_0 and a vector \mathbf{r} with rain values for a number of days as input. The function should return a water height after the days have passed. The function should also work if \mathbf{r} contains only one or no days. (In case of no days, h_0 should be returned.)

■ Solution template

```
def water_height(h0, r):  
    # insert your code  
    return h
```

Input

h_0 A non-negative number with initial water height.
 \mathbf{r} A vector with rain values (non-negative numbers) for a number of days. A vector may contain multiple, only one, or no elements.

Output

h The water height after the number of days has passed.

Casestudy: Introduction to python (Vedrana)

Codejudge

- 5 problems
- 11 tests per problem
- 60 files
- 282 lines of code

Unitgrade

- 5 problems
- 11 tests per problem
- 3 files
- 127 lines of code

Bonus: My version contains a handout stub for students to work with.

Casestudy: Introduction to python (grading)

My Course ▾ Announcements Course Content ▾ Help ▾ Piazza online QA

[Assignments](#) > [Project part 1: Dynamical Programming](#) > [Submissions](#)

Project part 1: Dynamical Programming - Submissions

Publish All

Edit Assignment

Email Groups Without Submissions

Add Feedback Files

Submission Log

Users

Submissions

Search For...



[Show Search Options](#)

Download

Email

Mark as Read

Mark as Unread

Delete

Publish Feedback

<input type="checkbox"/>		Submission Date	Last Name ▲, First Name	Delete
<input type="checkbox"/>	Group 1		Published: 14 March, 2022 10:18 PM Feedback Read: 18 March, 2022 10:52 AM	

Casestudy: Problem set 6 (Alceste)

02393 C++ Programming Exercises

Assignment 6

To be handed in via CodeJudge — <https://dtu.codejudge.net/02393-e21/assignments>

1 A class for fractions of integers

The goal of this assignment is to implement a class of fractions of integers supporting some basic operations like addition and multiplication. You can use the following interface for the class as a starting point and modify it at will:

```
class fraction {
private:
    // Internal representation of a fraction as two integers
    int numerator;
    int denominator;

public:
    // Class constructor
    fraction(int n, int d);

    // Methods to update the fraction
    void add(fraction f);
    void mult(fraction f);
    void div(fraction f);

    // Display method
    void display(void);
};
```

The main idea for the class above is that a fraction $\frac{a}{b}$ is represented by two integers (the numerator a and the denominator b) and several methods are provided to support arithmetic operations on fractions. For example:

- `fraction(int n, int m)` constructs the fraction $\frac{n}{m}$;
- `void add(fraction f)` updates a fraction by adding fraction f to it.
- `void mult(fraction f)` updates a fraction by multiplying it by fraction f .
- `void div(fraction f)` updates a fraction by dividing it by fraction f .

Your task. Write a program that reads sequences of simple expressions from `cin`, each having one of the following forms:

- `a / b + c / d`
- `a / b * c / d`

Class representing fractions

Casestudy: Problem set 6 (Alceste)

- Problem set 6, create a Fraction class with addition

Codejudge

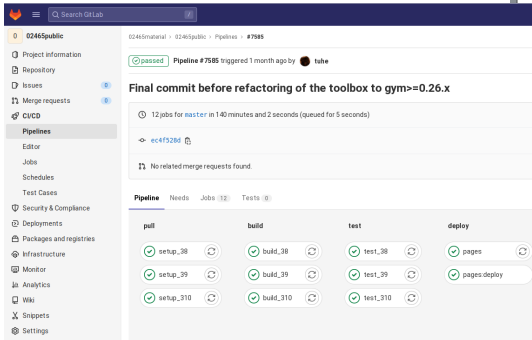
- 1 problem
- 8 files (.in, .ans)
- 2 tests
- 12 lines of code (only specifies input & output; contains more code in LaTeX)

Unitgrade

- 1 problems
- 2 files
- 6 tests
- 86 lines of code

Bonus: Contains more specific tests, generates handout files, contains a solution, and allows automatic checks of implementation for later refactoring.

Features: Automation



The screenshot shows the GitLab CI/CD interface for a project named '02465public'. The left sidebar contains navigation links for Project information, Repository, Issues, Merge requests, CI/CD, Pipelines, Editor, Jobs, Schedules, Test Cases, Security & Compliance, Deployments, Packages and registries, Infrastructure, Monitor, Analytics, Wiki, Snippets, and Settings. The main content area shows a pipeline named 'Pipeline #7585' triggered 1 month ago by 'tuh'. The pipeline status is 'passed'. Below this, it says 'Final commit before refactoring of the toolbox to gym>=0.26.x'. It indicates '12 jobs for master in 140 minutes and 2 seconds (queued for 5 seconds)'. A link to 'ec4f528d' is provided. Below this, it says 'No related merge requests found.' The pipeline details section shows a table with columns: Pipeline, Needs, Jobs (32), Tests (8). The jobs are grouped into four categories: pull, build, test, and deploy. Each category has three jobs listed with their status (passed) and a refresh icon.

Pipeline	Needs	Jobs (32)	Tests (8)
pull		build	test
✓ setup_38		✓ build_38	✓ test_38
✓ setup_39		✓ build_39	✓ test_39
✓ setup_310		✓ build_310	✓ test_310
			deploy
			pages
			pages.deploy

1 Working with fractions (fractions.py)

In this problem, you have to implement the methods in the following class:

```
# cpp_course/fractions.py
class Fraction:
    def __init__(self, n, m):
        self.n = n
        self.m = m

    def __add__(self, other):
        # Computes f = f1 + f2 (where f1 and f2 are both Fraction-objects and
        # f1=self, f2=other)
        return f

    def __mul__(self, other):
        # Overwrite to implement f = f1 * f2
        return f

    def __truediv__(self, other):
        # Overwrite to implement f = f1/f2, or more specifically self/other.
        return f

    def __str__(self):
        """ Creates a string representation. You can use it as
        """
        return str(Fraction(1,2)) to output 1/2"""
        return f"({self.n} / {self.m})"
```

When done, you should be able to create two fractions objects and add them using the `+`-operator, which will be transformed to a call of the form `f1.__add__(f2)` as so:

```
# cpp_course/fractions.py
f1 = Fraction(1, 2) # Represents 1/2
```


Casestudy: Exam in C++ Course 2021

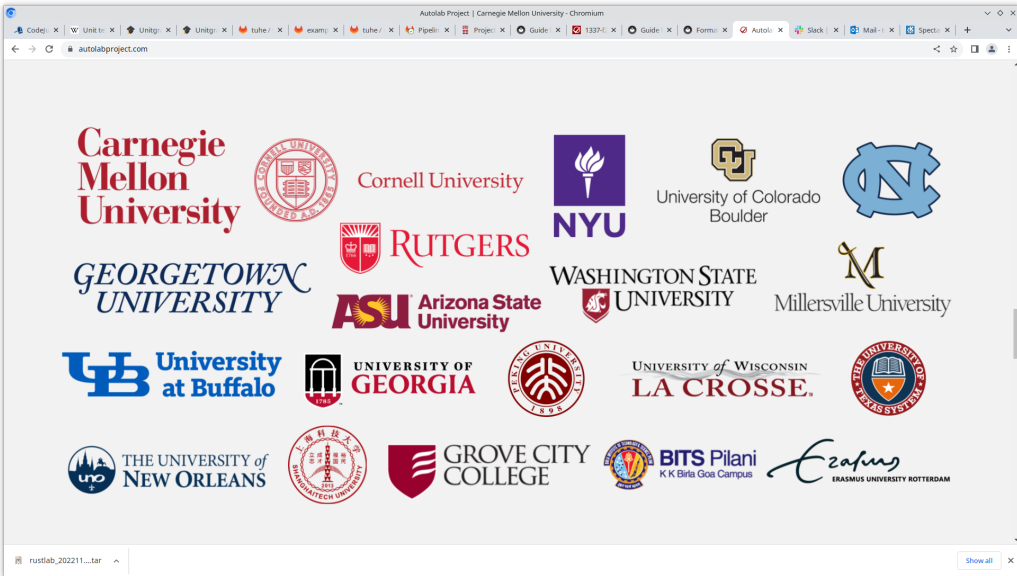
- Exam set for 2021
- 4 problems

Codejudge

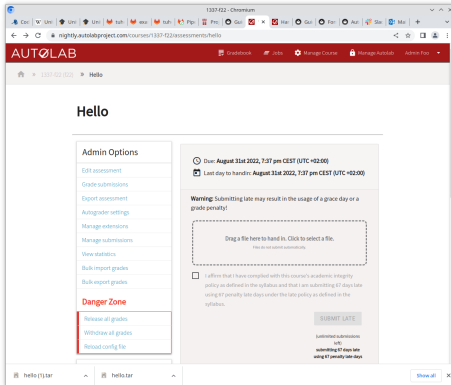
- 72 files (!)
- 16 .yaml files
- 18 .ans files
- 552 lines of code (excluding .ans files)

Unitgrade

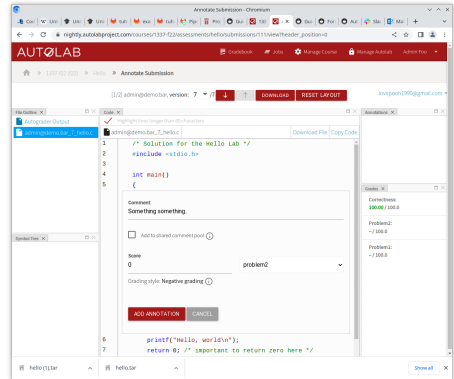
- 6 files (4 problems + 1 test + 1 deploy)
- 116 lines of code
- More tests



- Classic autograder: Students, classes, assignments.
- Sign-up can be handled using student email (confirmation emails)



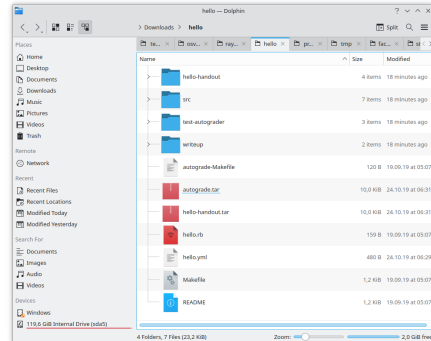
16 DTU Compute



Two alternative takes on autograding

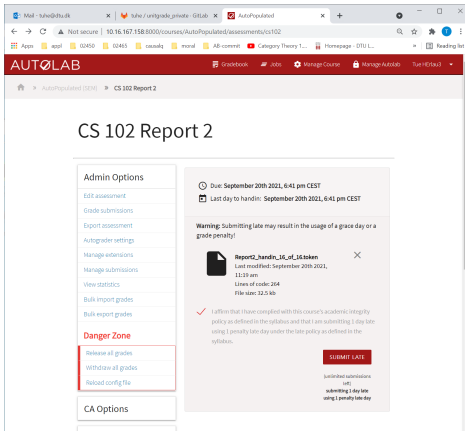
7.11.2022

- An assignment is a single .tar file:
- tl;dr:
 - The container will copy your autograder files the students into the same directory
 - Then it calls autograde_Makefile
 - You return json with the scores to stdout
- Easy to call e.g. a unittest script
- Tests can be run locally
- Many examples online



- Unitgrade allows one-line deployment to an autolab .tar file for upload

```
output_tar = deploy_assignment(name, INSTRUCTOR_BASE=instructor_base, INSTRUCTOR_GRADE_FILE=f"{instructor_base}/{name}/{instructor}",  
STUDENT_BASE=student_base, STUDENT_GRADE_FILE=f"{student_base}/{name}/{student}",  
autograde_image_tag=autograde_image)
```



CS 102 Report 2

Admin Options

- Edit assessment
- Grade submissions
- Export assessment
- Autograder settings
- Manage extensions
- Manage submissions
- View statistics
- Bulk import grades
- Bulk export grades

Danger Zone

- Release all grades
- Withdraw all grades
- Reload config file

CA Options

Warning: Submitting late may result in the usage of a grace day or a grade penalty!

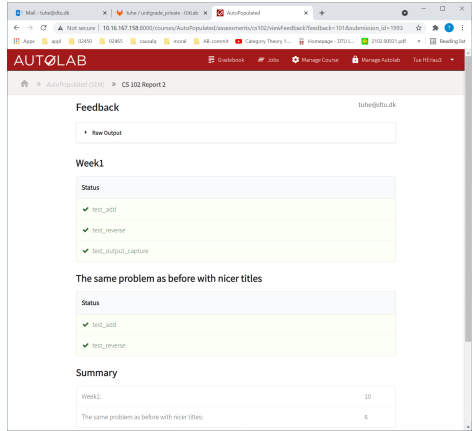
Report_handin_id_of_idtoken

Last modified: September 20th 2021, 11:19 am
Lines of code: 264
File size: 32.5 kb

I affirm that I have complied with this course's academic integrity policy as defined in the syllabus and that I am submitting 1 day late using 1 penalty late day under the late policy as defined in the syllabus.

SUBMIT LATE

(unlimited submissions left)
submitting 1 day late using 1 penalty late day



Feedback

tue@dtu.dk

New Output

Week1

Status

- ✓ test_add
- ✓ test_reverse
- ✓ test_output_capture

The same problem as before with nicer titles

Status

- ✓ test_add
- ✓ test_reverse

Summary

Week1:	10
The same problem as before with nicer titles:	6