You need access to a computer with Internet access and Python with all libraries used in the course exercises installed. Join the course Slack channel.

You are allowed to use Internet and course materials. You are not allowed to discuss with anyone in any form during the exam. You are allowed to discuss with the course instructor in Slack.

All code and text must be written by yourself. Copying substantial parts of code from any source is plagiarism.

1. Basics (5pts) **[Return before 10am15]**

   Take a screenshot of your computer desktop and save it (see Figure 1). Open a Web browser and go to the course Moodle page, log in, open the grades view that shows your points for each exercise, move the browser window to the top left so that the exercise grades and desktop are both visible and take another screenshot. Submit screenshots to the exam Moodle.

   Submitted items:

   - Screenshot 1 as PNG file: `Surname_Firstname_desktop1.png`
   - Screenshot 2 as PNG file: `Surname_Firstname_desktop2.png`

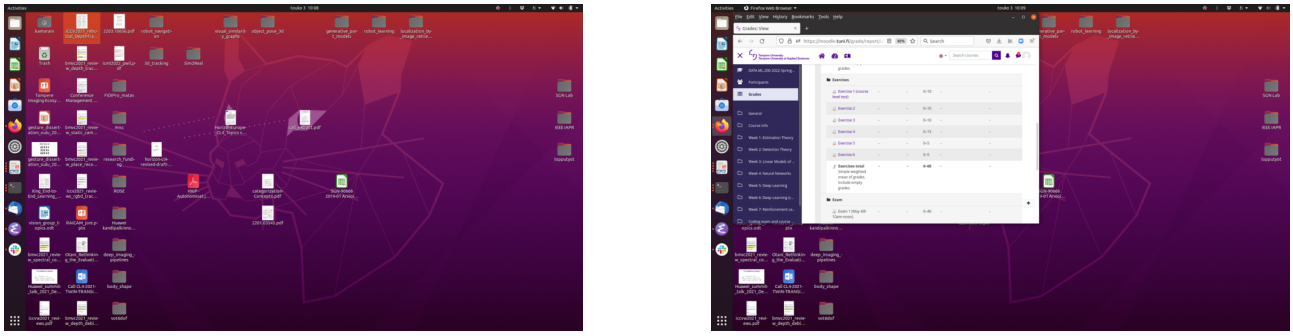

Figure 1: Empty desktop (left) and desktop including your exercise grades Moodle view (right)

   Note: Do not update any items after their deadlines!

2. Scikit-Learn classifier comparison (35pts) **[Multiple deadlines]**

   You are allowed to use available implementations of the below classifiers in Sklearn and also other libraries used in the course exercises.

   Download the training and test sample data files (`X_train.dat` and `X_test.dat`), and the correct class labels for the samples (`y_train.dat` and `y_test.dat`).

   (a) Random classifier (5pts) **[Return before 10am45]**
   Implement a random classifier that randomly assigns one of the class labels to all test samples. Compute classification accuracy and print it. Everything should be in a single Python program and run as shown below:

```
$ python kamarainen_joni_classifier1.py
Random accuracy: x.xxxx
$
```

   Submitted items:

   - Screenshot of running the code in your desktop: `Surname_Firstname_classifier1.png`
   - Python code: `Surname_Firstname_classifier1.py`

(b) k-NN classifier (10pts) **[Return before 11am15]**
Extend the previous program by adding there a k-NN classifier. Run k-NN (Nearest Neighbor) for k=1, 2, 3, 4, 5 and output its test set accuracies:

```
$ python kamarainen_joni_classifier2.py
Random accuracy: x.xxxx
k-NN accuracy x.xxxx for k=1
k-NN accuracy x.xxxx for k=2
k-NN accuracy x.xxxx for k=3
k-NN accuracy x.xxxx for k=4
k-NN accuracy x.xxxx for k=5
$
```

Submitted items:

- Screenshot of running the code: `Surname_Firstname_classifier2.png`
- Python code: `Surname_Firstname_classifier2.py`

(c) Decision tree classifier (15pts) **[Return before 11am45]**
Further extend your code and add a decision tree classifier using the maximum tree depths of 1, 2, 3, 4 and 5, and output the test set accuracies:

```
$ python kamarainen_joni_classifier3.py
Random accuracy: x.xxxx
k-NN accuracy x.xxxx for k=1
...
k-NN accuracy x.xxxx for k=5
Decision tree accuracy: x.xxxx for max depth 1
Decision tree accuracy: x.xxxx for max depth 2
Decision tree accuracy: x.xxxx for max depth 3
Decision tree accuracy: x.xxxx for max depth 4
Decision tree accuracy: x.xxxx for max depth 5
$
```

Submitted items:

- Screenshot of running the code: `Surname_Firstname_classifier3.png`
- Python code: `Surname_Firstname_classifier3.py`

(d) Decision tree plot (5pts) **[Return before 12pm00 (noon)]**

Submitted items:

- Decision tree graphical plot (max depth achieving the best accuracy): `Surname_Firstname_classifier3_tree.png`