



SINTEF

SINTEF Virtual Lab (vLab)

User meeting 2023



SINTEF

What is the vLab?



docker



kubernetes



Azure

Server Options

SHOP stable (15.4.0.1)

Latest stable version of SINTEF Energy's Short-term Hybrid Optimization Program. Python 3.11. Julia and R kernels. Based on jupyter/datascience-notebook. Visit <https://shop.sintef.energy> for more information, and to connect your vLab to our interactive documentation and examples.

SHOP latest (ba12dfd2)

Latest build of SHOP SINTEF Energy's Short-term Hybrid Optimization Program. Pre-release. Python 3.11. Julia and R kernels. Based on jupyter/datascience-notebook. Visit <https://shop.sintef.energy> for more information, and to connect your vLab to our interactive documentation and examples.

Temporarily static SHOP 15.3.4.0 on Jupyterlab v3 and Python 3.10

Version 15.3.4.0 of SHOP using Python 3.10 and Jupyterlab v3. Use this image if you need the git GUI, as it is not yet available in Jupyterlab v4.

ProdRisk 10.5.1 + SHOP 15.4.0.1

Python 3.11. Julia and R kernels. Based on jupyter/datascience-notebook. ProdRisk examples pre-deployed.



Latest files



SHOP 15.4.0.1

31.22 MB 9 file(s)



SHOP 15.4.0.0

17.01 MB 10 file(s)



SHOP 15.3.5.0

33.12 MB 12 file(s)



ProdRisk



Latest files



Release 10.5.1

115.57 MB 2 file(s)



ProdRisk 10.5.1 release candidate 2

113.67 MB 2 file(s)



ProdRisk 10.5.1 release candidate

117.37 MB 2 file(s)

OPEN TABS Close All

- basic_vlab.ipynb
- battery_wind_solar.md
- Launcher
- cuts.md
- shop_hydrosun_test_cases_6.0_World_test_1.md
- shop_hydrosun_test_cases_v6.0_World_preparation.md
- Terminal 1
- Terminal 2
- Launcher

KERNELS Shut Down All

- Python 3 (ipykernel)
- shop_hydrosun_test_cases_6.0_World_test_1.ipynb
- Python 3 (ipykernel)
- shop_hydrosun_test_cases_6.0_World_test_1.md
- Python 3 (ipykernel)
- basic_vlab.ipynb
- Python 3 (ipykernel)
- battery_wind_solar.md
- Python 3 (ipykernel)
- cuts.md
- Python 3 (ipykernel)
- shop_hydrosun_test_cases_v6.0_World_preparation.md

LANGUAGE SERVERS Shut Down All

- pylsip (python)

TERMINALS Shut Down All

- terminals/1
- terminals/2

Results

Finally we plot and print and plot some results.

Expected objective value

```
[13]: my_area = prodrisk.model.area["my_area"]
expected_objective_val_kkr = my_area.expected_objective_value.get()
print(f"Expected objective value: {expected_objective_val_kkr} kkr")
Expected objective value: 21227.695 kkr
```

Stochastic time series of reservoir volumes for ModuleA

```
[14]: rsv_vols = mod.reservoirVolume.get()
ltm_res.plot_percentiles(rsv_vols, "Volume [Mm3]", "", percentiles_limits=[0, 25, 50, 75, 100])
```

Stochastic time series of local inflow to ModuleA

```
[15]: inflow = mod.localInflow.get()
ltm_res.plot_percentiles(inflow, "Reservoir inflow [m3/s]", "", percentiles_limits=[0, 25, 50, 75, 100])
```

Visual topology

```
[5]: # Print out the topology
shop.model.build_connection_tree()

[5]:
```

Key topology data

```
[5]: input_plot_functions.plot_key_topology_data(shop, currency)
```

Reservoir Reservoir_A LRL: 479.0 meter above sea level
 Reservoir Reservoir_A HRL: 490.0 meter above sea level
 Reservoir Reservoir_A max volume: 1168.625 Million m3
 Reservoir Reservoir_A max surface area: 162.0 km2
 Reservoir Reservoir_B LRL: 462.0 meter above sea level
 Reservoir Reservoir_B HRL: 464.0 meter above sea level
 Reservoir Reservoir_B max volume: 12.3 Million m3
 Reservoir Reservoir_B max surface area: 7.7 km2

```
fig.update_layout(title=plot_figure.title)
fig.update_layout(legend=dict(orientation="h", y_pos=0))
fig.update_xaxes(title_text="<b>Time</b> (Hour)")
fig.update_yaxes(title_text="<b>Hourly inflow</b> (m³/s)")
fig.update_xaxes(title_text="<b>Hourly inflow</b> (m³/s)")
fig.update_yaxes(title_text="<b>Yearly inflow</b> (m³/s)")
fig.update_xaxes(title_text="<b>Yearly inflow</b> (m³/s)")
fig.update_yaxes(title_text="<b>Yearly inflow</b> (m³/s)")
fig.update_layout(showlegend=False)

fig.add_vrect(x0="2003-12-25", x1="2005-01-05", fillcolor="red", fillwidth=0.5)

fig.show()
```

Yearly average inflow: 2802.3411529333084

48-year original reservoir inflow in K...

```
[4]: # Read inflow in the reference year
Reservoir_A_inflow = round(pd.read_csv("auxiliary/Reservoir_A_inflow.csv"), 2)
Reservoir_B_inflow = round(pd.read_csv("auxiliary/Reservoir_B_inflow.csv"), 2)

# Combine two series of reservoir inflow
inflow_hourly = pd.DataFrame()
inflow_hourly = pd.concat([Reservoir_A_inflow, Reservoir_B_inflow])

# Write to CSV file
inflow_hourly.to_csv("auxiliary/Data_inflow_6.csv", index=False)
```

Battery, solar and wind

This example shows how batteries can be modelled in SHOP to smooth variations for solar or wind to meet a firm demand. We define the optimization horizon over 24 hours with 15 minutes time resolution:

```
[1]: from pyshop import ShopSession
import pandas as pd
import numpy as np
import plotly.graph_objs as go

shop = ShopSession()
starttime = pd.Timestamp('2019-05-16T00:00:00')
endtime = pd.Timestamp('2019-05-17T00:00:00')
shop.set_time_resolution(starttime=starttime, endtime=endtime, time_resolution=15)
```

We create a battery with 90% charge and discharge efficiency. The battery size is 4 MWh with 0.5 MW as the maximum charge and discharge capacity. The initial state-of-charge is 2 MWh and end state-of-charge must be greater than or equal to the initial.

```
[2]: battery = shop.model.battery.add_object("Battery")
battery.charge_efficiency.set(0.9)
battery.discharge_efficiency.set(0.9)
battery.max_charge_power.set(0.5)
battery.max_discharge_power.set(0.5)
battery.max_energy.set(4.0)
battery.initial_energy.set(2)
```

```
init_value = battery.initial_energy.get()
battery.max_energy_constraint.set(pd.Series(index=[starttime, endtime], data=[init_value, init_value]))
```

We also create a solar power source with a typical profile acquired from renewables.ninja:

```
[3]: solar_profile = pd.read_csv('solar_profile.csv', header=3).set_index('datetime')
solar_profile.index = pd.to_datetime(solar_profile.index)
solar_profile = solar_profile[solar_profile['electricity'] > 0]
solar = shop.model.solar.add_object("Solar")
solar.power_forecast.set(solar_profile)
```

Now we can run the optimization and look at the cut results. The final value in the TXY attribute `end_value` on the `cut_group` object holds the optimized cut value (future expected (negative) income). The `binding_cut_up` attribute holds the cut constraint that was the most constraining cut in this optimization.

```
[5]: #Optimize model by calling "run_model"
run_model(shop)

#Get the final end value of the cut group, which is the future expected income
cut_val = -my_cuts.end_value.get().iloc[-1]

#Get the binding cut constraint and print the cut information
cut_index = int(cut_index)

rhs_out = my_cuts.rhs.get()[0]
active_rhs = rhs_out.values[cut_index]

print(f"The future expected income is {cut_val:2f} € and the binding constraint is {active_rhs:2f} €/h")
for rsv in shop.model.reservoir:
    name = rsv.get_name()
    cut_coeffs = rsv.water_value_input.get()[0]
    v_ref = cut_coeffs.index[cut_index]
    ww = cut_coeffs.values[cut_index]

end_val = rsv.storage.get().iloc[-1]
```

```
thon3.11/site-packages/~rodrisk_shop_simulator-0.1.0.dist-info/.
You can safely remove it manually.
Successfully installed prodrisk-shop-simulator-0.1.0
jovyan@jupyter-auth-7cportal-7c1:~/prodrisk-shop-simulator-new$ python examples/basecase/dashboards/prodrisk/script.py simulator_checks
onfig.yaml
.ipynb checkpoints/ results/ shop_model.yaml
jovyan@jupyter-auth-7cportal-7c1:~/prodrisk-shop-simulator-new$ python examples/basecase/script.py
Init simulator
Reading Prodrisk cuts: 100%
Saving cuts to file: 100%
2/2 [00:02-00:00, 1.89s/it]

Init SHOP simulations
Scenario 0: 100%
53/53 [01:11-00:00, 1.35s/it]
Simulator time: 74.21814632415771 seconds.
jovyan@jupyter-auth-7cportal-7c1:~/prodrisk-shop-simulator-new$
```

```
top - 12:55:22 up 6 days, 6:47, 0 users, load average: 1.38, 1.34, 1.40
Tasks: 12 total, 2 running, 10 sleeping, 0 stopped, 0 zombie
%Cpu(s): 29.1 us, 3.1 sy, 0.0 ni, 66.1 id, 0.1 wa, 0.0 hi, 1.6 si, 0.0 st
MiB Mem : 15990.8 total, 1599.2 free, 9518.1 used, 4692.5 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used, 6121.9 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
7	jovyan	20	0	4897324	1.7g	65624	R	100.7	10.6	0:16.35	jupyter
189	jovyan	20	0	1791892	392796	72236	S	0.3	2.4	0:40.02	python
1	jovyan	20	0	2784	948	868	S	0.0	0.0	0:00.13	tmux
152	jovyan	20	0	756812	78284	16988	S	0.0	0.0	0:00.75	python
205	jovyan	20	0	1022612	242904	68852	S	0.0	1.5	0:03.13	python3+
222	jovyan	20	0	757028	67708	16724	S	0.0	0.4	0:00.79	python
339	jovyan	20	0	1829312	196892	68272	S	0.0	1.2	0:02.17	python
442	jovyan	20	0	7632	4108	3472	S	0.0	0.0	0:00.66	bash
606	jovyan	20	0	7632	4192	3572	S	0.0	0.0	0:00.72	bash
654	jovyan	20	0	10352	3432	3068	R	0.0	0.0	0:00.22	top
1693	jovyan	20	0	1640264	244168	74668	S	0.0	1.5	0:04.27	python
2039	jovyan	20	0	1497164	187308	64648	S	0.0	1.1	0:01.86	python

RTC:hydrosun/test_cases_version_6.0/World

Notebook

- Python 3 (ipykernel)
- Julia 1.9.3
- Pluto Notebook
- R

Console

- Python 3 (ipykernel)
- Julia 1.9.3
- R

Other

- Terminal
- Text File
- Markdown File
- Julia File
- Python File
- R File
- Show Contextual Help



SINTEF

GitLab + vLab == True

GitLab interface for the HydroSun project. The page shows the repository details, including the project ID (3645), 12 commits, 2 branches, and 0 tags. A commit by Jiehong Kong is highlighted with the message "Update latest test cases and clean the old code". Below this, there are options to add various files like README, LICENSE, and CONTRIBUTING. A table lists the repository's contents, including folders like 'document', 'paper', 'presentation', 'report', and 'test_cases_version_3.3', along with their last commit dates. At the bottom, there is a section for the "HydroSun latest stable test cases version" with a brief description.

Name	Last commit	Last update
document	upload all the necessary documents	6 months ago
paper	upload all the necessary files for the pr...	6 months ago
presentation	upload all the necessary files for the pr...	6 months ago
report	upload all the necessary files for the pr...	6 months ago
summer student version	add summer student version	2 weeks ago
test_cases_version_3.3	upload files for v3.3	6 months ago
test_cases_version_5.0	delete unrelated cases in Imola in versi...	2 weeks ago
test_cases_version_6.0	Update latest test cases and clean the ...	1 day ago
README.md	update README.md	6 months ago



vLab interface showing the HydroSun test cases (v6.0) - Test 1 for World. The interface includes a file explorer on the left, a central panel with a commit message "shop_hydrosun_test_cases_6.0_World_test_1.md", and a right-hand panel displaying the test case details. The details include the responsible parties (Jiehong Kong and Bjørnar Fjeldal), the latest update date (30.09.2023), and an introduction section. The introduction describes the test's purpose: to check the impact of hydro inflow and solar availability on production schedules of hybrid plants in Western Africa, Eastern Asia, and Northern Europe. It also lists specific regions: Guinea (Kogbedou), Philippines (Magat), and Norway (Trondheim). Below the introduction, there is a section for checking the watercourse and input data, which includes a code snippet for setting up the test functions and libraries.

```

[1]: # Add functions developed for this example. They are saved in the same folder.
import set_test_functions
import execute_test_functions
import input_plot_functions
import result_plot_functions

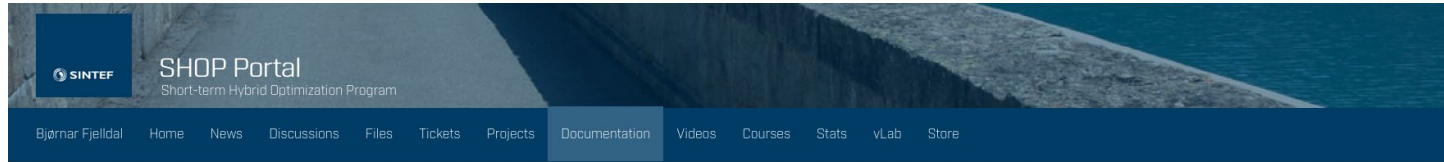
# Add basic libraries and packages for running pySHOP
from pyshop import ShopSession
import pandas as pd
import plotly.express as px
from plotly.subplots import make_subplots

```



SINTEF

Integrated with documentation and courses



Getting started

Introduction

Interacting with SHOP

Attribute datatypes

Commands, objects and attributes

Commands

Objects

Attributes

Tutorials

Water value descriptions

Reserves

Ramping

Best Profit

Simulation

Examples

Individual water values in SHOP

Cut descriptions in SHOP

Basic pump example

Reserve capacity allocation

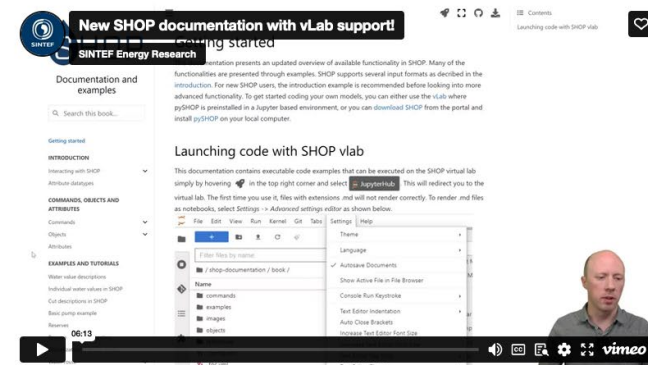
Discretization of droop results

Water route



Best profit basin example

Getting started

This documentation presents an updated overview of available functionality in SHOP. Many of the functionalities are presented through examples. SHOP supports several input formats as described in the [introduction](#). For new SHOP users, the introduction example is recommended before looking into more advanced functionality. To get started coding your own models, you can either use the [vLab](#) where pySHOP is preinstalled in a Jupyter based environment, or you can [download SHOP](#) from the portal and install [pySHOP](#) on your local computer.



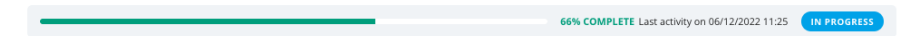
Launching code with SHOP vlab

This documentation contains executable code examples that can be executed on the SHOP virtual lab simply by hovering  in the top right corner and select . This will redirect you to the virtual lab. The first time you use it, files with extensions `.md` will not render correctly. To render `.md` files as



Introduction to pyShop

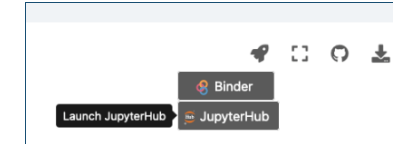
17/02/2022 Bjørnar Fjellidal



This course gives an introduction to the basic building blocks for pySHOP and will show you some of the powerful features in scripting SHOP from Python.

The course will cover how to configure time intervals and time resolutions, how to add objects such as reservoirs and plants to a pySHOP instance, how to apply attributes to these objects, and how to build relations between them. We will also have a look at how to apply commands defining how SHOP and its solver should operate.

In order to access the vLab inside of the course module for lessons with exercises, you can do so by hovering over the rocket logo and selecting "JupyterHub":



Selecting this in lessons with exercises opens the vLab inside of the course module. Make sure to be logged in prior to pressing this

Course Content

- Introduction
- Time
- Objects
- Attributes
- Relations
- Commands





SINTEF

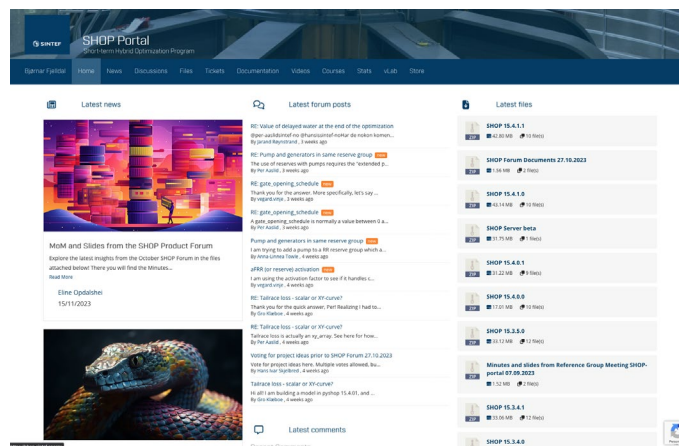
Single-sign-on (SSO)



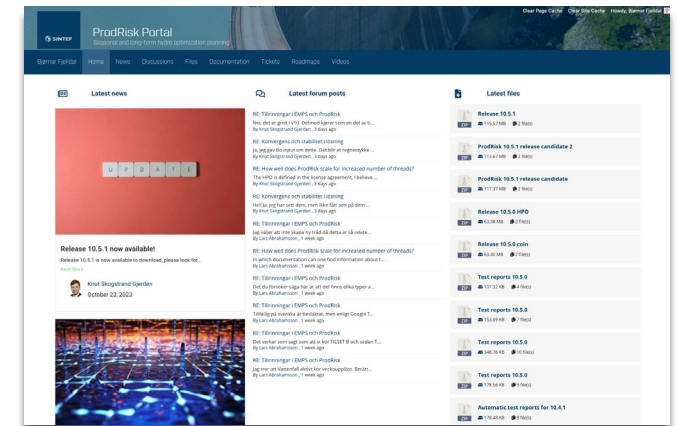
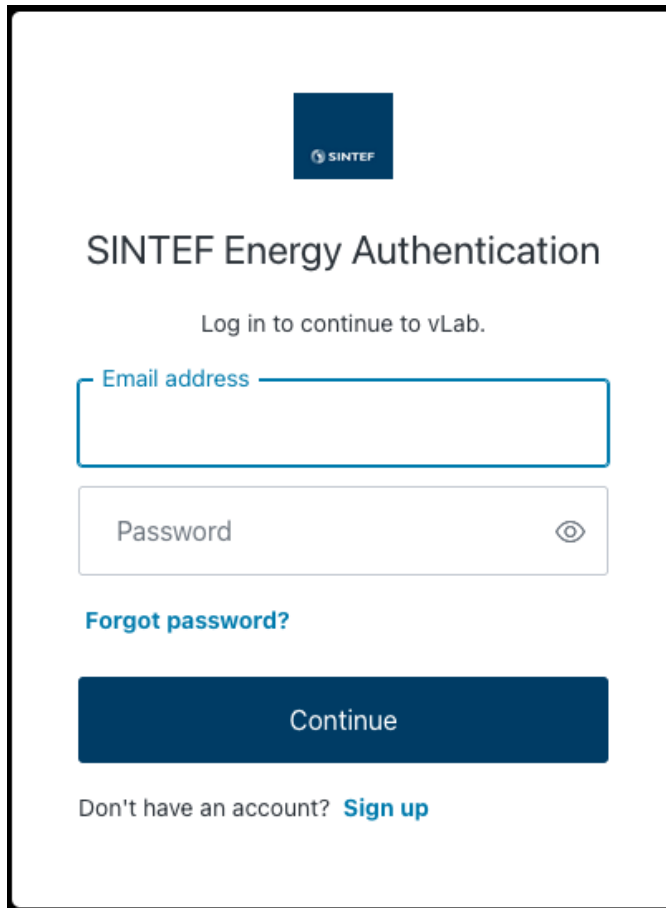
Server Options

- **SHOP stable (15.4.0.1)**
Latest stable version of SINTEF Energy's Short-term Hybrid Optimization Program. Python 3.11, Julia and R kernels. Based on [jupyter/datascience-notebook](#). Visit <https://shop.sintef.energy> for more information, and to connect your vLab to our interactive documentation and examples.
- **SHOP latest (ba12dfd2)**
Latest build of SHOP SINTEF Energy's Short-term Hybrid Optimization Program. Pre-release. Python 3.11, Julia and R kernels. Based on [jupyter/datascience-notebook](#). Visit <https://shop.sintef.energy> for more information, and to connect your vLab to our interactive documentation and examples.
- **Temporarily static SHOP 15.3.4.0 on Jupyterlab v3 and Python 3.10**
Version 15.3.4.0 of SHOP using Python 3.10 and Jupyterlab v3. Use this image if you need the git GUI, as it is not yet available in Jupyterlab v4.
- **ProdRisk 10.5.1 + SHOP 15.4.0.1**
Python 3.11, Julia and R kernels. Based on [jupyter/datascience-notebook](#). ProdRisk examples pre-deployed.

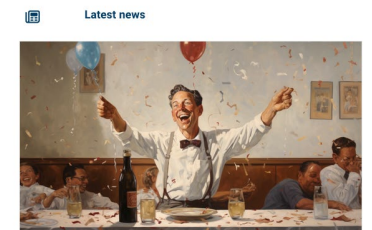
vlab.sintef.energy



shop.sintef.energy



prodrisk.sintef.energy



ltm.sintef.energy



Significantly faster

The image displays two side-by-side screenshots of a Jupyter Notebook interface, illustrating a performance comparison. Both screenshots show the same notebook content, which includes code for generating data and plotting it. The code is as follows:

```
(1): from matplotlib import rcParams, cycler
import matplotlib.pyplot as plt
import numpy as np
plt.ion()

(2): # Fixing random state for reproducibility
np.random.seed(19680001)

N = 10
data = [np.logspace(0, 1, 100) + np.random.randn(100) + ii for ii in range(N)]
data = np.array(data).T
cmap = plt.cm.coolwarm
rcParams['axes.prop_cycle'] = cycler(color=cmap(np.linspace(0, 1, N)))

from matplotlib.lines import Line2D
custom_lines = [Line2D([0], [0], color=cmap(0.), lw=4),
                Line2D([0], [0], color=cmap(.5), lw=4),
                Line2D([0], [0], color=cmap(1.), lw=4)]

fig, ax = plt.subplots(figsize=(10, 5))
lines = ax.plot(data)
ax.legend(custom_lines, ['Cold', 'Medium', 'Hot'])
```

The left screenshot shows the notebook running, with a 'stopwatch' window overlaid. The stopwatch shows a total time of 00:00:00.000 and a lap time of 00:00:00.000. The right screenshot shows the same notebook with a 'Would you like to receive official jupyter news?' dialog box overlaid.



SINTEF

Real-time collaboration / sharing

File Edit View Run Kernel Tabs Settings Help

User info

t
vex

Online Collaborators

p pike • collab.ipynb

collab.ipynb

[1]: `print("Hello, pike")`
Hello, pike

[2]: `print("Hello, vex")` pike
Hello, vex

[]:



SINTEF

AI support (soon)

The screenshot shows a Jupyter Notebook interface with a sidebar on the left and a main code editor on the right. The sidebar contains a list of instructions for solving the LCM problem, a summary, and a button to 'Rewrite this code with comments added to it'. The main editor shows a series of five code cells:

- [1]: `%reload_ext jupyter_ai`
- [2]: `%ai chatgpt --format code`
A program that asks me for my name and then greets me by my name, in Polish
- [3]: `name = input("Jak masz na imię? ")`
`print("Cześć " + name + "!")`
Jak masz na imię? foo
Cześć foo!
- [4]: `%ai chatgpt --format code`
A function that computes the lowest common multiples of two integers, and a function that runs 5 test cases of the lowest
- [5]:

```
def lcm(x, y):  
    if x > y:  
        greater = x  
    else:  
        greater = y  
  
    while True:  
        if (greater % x == 0) and (greater % y == 0):  
            lcm = greater  
            break  
        greater += 1  
  
    return lcm  
  
def test_lcm():  
    assert lcm(3, 5) == 15  
    assert lcm(7, 9) == 63  
    assert lcm(18, 24) == 72  
    assert lcm(10, 15) == 30  
    assert lcm(12, 16) == 48  
  
test_lcm()
```

The status bar at the bottom indicates 'Simple', '0', '1', 'Python 3 (ipykernel) | Idle', 'Mode: Command', 'Ln 1, Col 1', 'code.ipynb', and a notification bell icon.



SINTEF

Use vLab in your local VS Code

The screenshot shows the VS Code interface with a Jupyter notebook open. The notebook content includes:

```

Instantiating SHOP

In order to have SHOP receive our inputs, run the model we create and give us results, we need to create an active, running SHOP session.
You may create multiple SHOP sessions simultaneously if needed.

# Creating a new SHOP session to the instance 'shop'
shop = ShopSession()

We can also check the current versions of SHOP and its solvers.

# Writing out the current version of SHOP and its solvers
shop.shop_api.GetVersionString()

'15.4.0.1 Cplex 20.1.0 Gurobi 7.5 OSI/CBC 2.9 2023-09-29'

Setting time resolutions for the model

We set the time resolution for the model ourselves, as we generate all input as we go. The start and end time are important, in addition to the resolution of the time steps.

# Setting the start time of the model
starttime = pd.Timestamp('2018-01-23 00:00:00')

```

The kernel selection menu is open, showing options like Julia 1.9.3, Python 3 (ipykernel), and Jupyter Session.

Request new API token

Note

note to identify your new token

This note will help you keep track of what your tokens are for.

Token expires in

Never

You can configure when your token will expire.

API Tokens

These are tokens with access to the JupyterHub API. Permissions for each token may be viewed via the JupyterHub tokens API. Revoking the API token for a running server will require restarting that server.

Note	Last used	Created	Expires
vlab	2 minutes ago	21 hours ago	Never revoke
Server at /user/auth0%7Cportal%7C1/	4 minutes ago	4 minutes ago	Never revoke
Server at /user/auth0%7Cportal%7C1/	18 hours ago	21 hours ago	Never revoke
Server at /user/auth0%7Cportal%7C1/	2 days ago	2 days ago	Never revoke
Server at /user/auth0%7Cportal%7C1/	2 days ago	2 days ago	Never revoke
vscode	Never	21 hours ago	Never revoke

Authorized Applications

These are applications that use OAuth with JupyterHub to identify users (mostly notebook servers). OAuth tokens can generally only be used to identify you, not take actions on your behalf.

Application	Last used	First authorized
-------------	-----------	------------------



Demo



SINTEF

Technology for a better society