

hw2, Durable_rules

사용 규칙기반 시스템 구현

이 준 혁 (2023250419)
산 업 인 공 지 능 학 과

문제1: CNC가공기의 Z axis 영점 잡기 문제(Mil touch)

설명: Z 축 Zero 를 잡기 위하여 모터를 적정치 만큼 회전시키는 문제

- IF ROUTER MIL과 PROBE 가 접촉하지 않았는가 THEN Z 스테퍼 모터를 계속 회전시킨다
- IF ROUTER MIL과 PROBE 가 접촉하였는가 THEN Z 스테퍼 모터 회전을 정지한다(0점 조절 완료)



문제1: CNC가공기의 Z axis 영점 잡기 문제(Mil touch)



산업인공지능개론_과제_Durable ☆

파일 수정 보기 삽입 런타임 도구 도움말 모든 변경사항이 저장됨

☰ 목차

🔍 섹션

{

☐

<>

☰

+ 코드 + 텍스트

패키지 설치

✓ [43] !pip install durable_rules

5초

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>
Requirement already satisfied: durable_rules in /usr/local/lib/python3.9/dist-packages (2.0.28)

규칙 정의

✓ [48] from durable.lang import *

0초

with ruleset('cnc_machine'):

@when_all((m.router_mil == 'contacted') & (m.probe == 'contacted'))

def stop_z_stepper(c):

print('Router mil과 Probe가 접촉하였습니다. Z 스테퍼 모터 회전을 정지합니다.')

Z 스테퍼 모터 회전을 정지하는 코드를 작성

@when_all((m.router_mil == 'not_contacted') & (m.probe == 'not_contacted'))

def keep_rotating_z_stepper(c):

print('Router mil과 Probe가 접촉하지 않았습니다. Z 스테퍼 모터를 계속 회전시킵니다.')

Z 스테퍼 모터를 계속 회전시키는 코드를 작성

규칙 실행

✓ [50] post('cnc_machine', {'router_mil': 'contacted', 'probe': 'contacted'})

0초

post('cnc_machine', {'router_mil': 'not_contacted', 'probe': 'not_contacted'})

Router mil과 Probe가 접촉하였습니다. Z 스테퍼 모터 회전을 정지합니다.

Router mil과 Probe가 접촉하지 않았습니다. Z 스테퍼 모터를 계속 회전시킵니다.

{'sid': '0', 'id': 'sid-0', '\$s': 1}

✓ 0초 오후 7:10에 완료됨


문제2: 배터리 인디케이터의 완충 표시

설명: 리튬이온 셀의 배터리 전압 표시를 위한 장치

- IF 배터리 전압이 3.9V 이상인가 THEN 4개의 LED를 점등한다
- IF 배터리 전압이 3.7V 이상인가 THEN 3개의 LED를 점등한다
- IF 배터리 전압이 3.5V 이상인가 THEN 2개의 LED를 점등한다
- IF 배터리 전압이 3.3V 이상인가 THEN 1개의 LED를 점등한다
- IF 배터리 전압이 3.3V 미만인가 THEN 0개의 LED를 점등한다



문제2: 배터리 인디케이터의 완충 표시

 산업인공지능개론_과제_Durable ☆

파일 수정 보기 삽입 런타임 도구 도움말 저장하지 못함

☰ 목차

🔍 섹션

{x}

📁

+ 코드 + 텍스트

✓ [50] 4초 !pip install durable_rules

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>
Requirement already satisfied: durable_rules in /usr/local/lib/python3.9/dist-packages (2.0.28)

규칙 정의

✓ [59] 0초 from durable.lang import *

with ruleset('battery_voltage'):
 @when_all(m.voltage >= 3.9)
 def high_voltage(c):
 print('Turn on all 4 LEDs')

 @when_all(m.voltage >= 3.7, m.voltage < 3.9)
 def medium_high_voltage(c):
 print('Turn on 3 LEDs')

 @when_all(m.voltage >= 3.5, m.voltage < 3.7)
 def medium_low_voltage(c):
 print('Turn on 2 LEDs')

 @when_all(m.voltage >= 3.3, m.voltage < 3.5)
 def low_voltage(c):
 print('Turn on 1 LED')

 @when_all(m.voltage < 3.3)
 def very_low_voltage(c):
 print('Turn off all LEDs')

문제3: 태양광 컨트롤 장치의 배터리 충전

설명: 태양광 컨트롤 장치의 배터리 충전

- **IF** SOLAR CHARGER 와 BATTERY 가 연결 되었는가 **AND** 배터리 전압이 14V 이하인가 **THEN** 배터리를 충전한다.
- **IF** SOLAR CHARGER 와 BATTERY 가 연결 되었는가 **AND** 배터리 전압이 14V 초과인가 **THEN** 배터리 충전을 중지한다.



문제3: 태양광 컨트롤 장치의 배터리 충전

```
Untitled0.ipynb ☆
파일 수정 보기 삽입 런타임 도구 도움말 모든 변경사항이 저장됨

+ 코드 + 텍스트

!pip install durable_rules

from durable.lang import *

# 규칙 정의
with ruleset('battery_charge'):

    # 규칙 1: 배터리 전압이 14V 이하이면 배터리를 충전한다.
    @when_all(Battery << {'subject': 'BATTERY', 'voltage': float, 'connected': True})
    @when_all(lambda m: m['Battery']['voltage'] <= 14)
    @when_all(Solar_charger << {'subject': 'SOLAR CHARGER', 'connected': True})
    def charge_battery(c):
        print('Battery is being charged')

    # 규칙 2: 배터리 전압이 14V 초과이면 배터리 충전을 중지한다.
    @when_all(Battery << {'subject': 'BATTERY', 'voltage': float, 'connected': True})
    @when_all(lambda m: m['Battery']['voltage'] > 14)
    @when_all(Solar_charger << {'subject': 'SOLAR CHARGER', 'connected': True})
    def stop_charging(c):
        print('Battery charging is stopped')

# 사실적인 시나리오를 위해 13.8V로 가정합니다.
post('battery_charge', {'subject': 'BATTERY', 'voltage': 13.8, 'connected': True})
post('battery_charge', {'subject': 'SOLAR CHARGER', 'connected': True})

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: durable_rules in /usr/local/lib/python3.9/dist-packages (2.0.28)


NameError                                Traceback (most recent call last)
<ipython-input-3-547a6c20e140> in <cell line: 6>()
      7
      8 # 규칙 1: 배터리 전압이 14V 이하이면 배터리를 충전한다.
----> 9 @when_all(Battery << {'subject': 'BATTERY', 'voltage': float, 'connected': True})
     10 @when_all(lambda m: m['Battery']['voltage'] <= 14)
     11 @when_all(Solar_charger << {'subject': 'SOLAR CHARGER', 'connected': True})
```

문제4: 스마트 팜 시스템에서의 급수

- IF SOIL MOSTURE PROBE 값이 40% 이하인가 AND 물탱크가 차 있는가 THEN 급수를 진행한다.
- IF SOIL MOSTURE PROBE 값이 40% 초과인가 AND 물탱크가 차 있는가 THEN 급수를 중단한다.
- IF SOIL MOSTURE PROBE 값이 40% 이하인가 AND 물탱크가 비었는가 THEN 급수를 중단한다.





문제4: 스마트 팜 시스템에서의 급수

 산업인공지능개론_과제_Durable ☆

파일 수정 보기 삽입 런타임 도구 도움말 저장하지 못함

+ 코드 + 텍스트

 0초 

```
import time
from durable.lang import *

# 규칙 엔진을 초기화합니다.
with ruleset('irrigation'):


    # 규칙 1: 흙의 수분이 40% 이하이면서 물탱크가 차 있는 경우 급수를 시작합니다.
    @when_all((m.Soil_moisture_probe <= 40) & (m.Water_tank == 'full'))
    def start_irrigation(c):
        print('규칙 1: 급수를 시작합니다.')
        time.sleep(1)
        print('급수 완료')

    # 규칙 2: 흙의 수분이 40% 초과이면서 물탱크가 차 있는 경우 급수를 중단합니다.
    @when_all((m.Soil_moisture_probe > 40) & (m.Water_tank == 'full'))
    def stop_irrigation(c):
        print('규칙 2: 급수를 중단합니다.')

    # 규칙 3: 흙의 수분이 40% 이하이면서 물탱크가 비어 있는 경우 급수를 중단합니다.
    @when_all((m.Soil_moisture_probe <= 40) & (m.Water_tank == 'empty'))
    def stop_irrigation_empty_tank(c):
        print('규칙 3: 급수를 중단합니다.')

# 규칙 엔진을 실행합니다.
post('irrigation', {'Soil_moisture_probe': 30, 'Water_tank': 'full'})
post('irrigation', {'Soil_moisture_probe': 50, 'Water_tank': 'full'})
post('irrigation', {'Soil_moisture_probe': 30, 'Water_tank': 'empty'})
```

<>



규칙 1: 급수를 시작합니다.
급수 완료
규칙 2: 급수를 중단합니다.
규칙 3: 급수를 중단합니다.

문제5: 모터사이클의 Neutral indicating

- **IF** 기어 PROBE 와 N단 전극이 접촉 하였는가 **THEN** 녹색 LED 를 점등한다.
- **IF** 기어 PROBE 와 N단 전극이 접촉하지 않았는가 **THEN** 녹색 LED 를 소등한다



문제5: 모터사이클의 Neutral indicating

```
+ 코드 + 텍스트

from durable.lang import *

# 기어 PROBE가 접촉한 경우
Gear_probe = {'subject': 'Gear PROBE', 'contact': True}

with ruleset('gear_rules'):
    # 규칙 1: 기어 PROBE 와 N단 전극이 접촉 하였는가
    @when_all(Gear_probe << {'subject': 'Gear PROBE', 'contact': True},
              N_electrode << {'subject': 'N단 전극', 'connected': True})
    def gear_contact_on(c):
        print('녹색 LED를 켜다.')

    # 규칙 2: 기어 PROBE 와 N단 전극이 접촉하지 않았는가
    @when_all({'subject': 'Gear PROBE', 'contact': False},
              {'subject': 'N단 전극', 'connected': False})
    def gear_contact_off(c):
        print('녹색 LED를 끈다.')

# 사실적인 데이터 입력 및 규칙 적용
post('gear_rules', {'subject': 'Gear PROBE', 'contact': True})
post('gear_rules', {'subject': 'N단 전극', 'connected': True})

-----
TypeError                                Traceback (most recent call last)
<ipython-input-16-648d90212b02> in <cell line: 6>()
      6 with ruleset('gear_rules'):
      7     # 규칙 1: 기어 PROBE 와 N단 전극이 접촉 하였는가
----> 8     @when_all(Gear_probe << {'subject': 'Gear PROBE', 'contact': True},
      9                 N_electrode << {'subject': 'N단 전극', 'connected': True})
     10     def gear_contact_on(c):

TypeError: unsupported operand type(s) for <<: 'dict' and 'dict'
```

문제6: 도색 후 열처리 문제

- **IF** 열처리 시간이 100분을 미만인가 **THEN** OVEN 을 계속 ON 한다.
- **IF** 열처리 시간이 100분을 경과 하였는가 **THEN** OVEN 을 OFF 한다.



문제6: 도색 후 열처리 문제



산업인공지능개론_과제_Durable ☆

파일 수정 보기 삽입 런타임 도구 도움말 저장하지 못함

+ 코드 + 텍스트



4줄

```
!pip install durable_rules

from durable.lang import *

# 초기 상태
oven = {'status': 'OFF', 'processing_time': 0}

# Rule 선언
with ruleset('oven_rules'):

    # 규칙 1: 열처리 시간이 100분 미만인가
    @when_all(m.processing_time < 100)
    def oven_on(c):
        oven['status'] = 'ON'
        print('OVEN ON')

    # 규칙 2: 열처리 시간이 100분 이상인가
    @when_all(m.processing_time >= 100)
    def oven_off(c):
        oven['status'] = 'OFF'
        print('OVEN OFF')

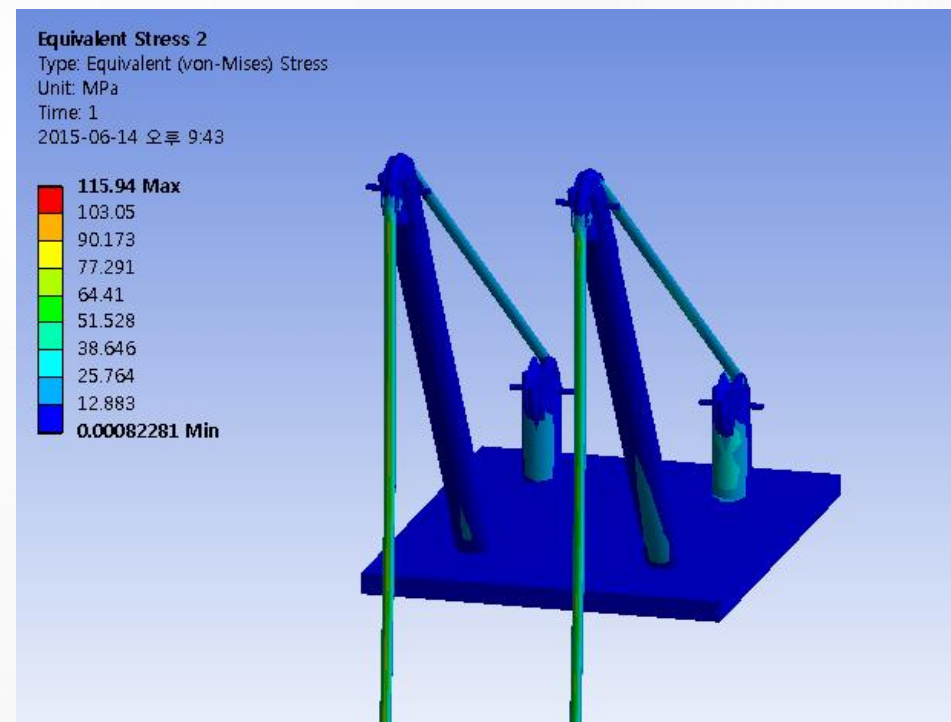
# 사실적인 데이터 입력 및 규칙 적용
oven['processing_time'] = 90
assert_fact('oven_rules', oven)

oven['processing_time'] = 110
assert_fact('oven_rules', oven)
```

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>
Requirement already satisfied: durable_rules in /usr/local/lib/python3.9/dist-packages (2.0.28)
OVEN ON
OVEN OFF
{'sid': '0', 'id': 'sid-0', '\$s': 1}

문제7: 응력을 이용하여 위험요소 경고하기

- **IF** 구조물에 응력이 기준치 (100mpa) 이하인가 **THEN** 경고 기능을 OFF 한다.
- **IF** 구조물에 응력이 기준치 (100mpa) 초과인가 **THEN** 경고 기능을 ON 한다.



문제7: 응력을 이용하여 위험요소 경고하기

← → ↺ 🔒 colab.research.google.com/drive/1zyb9sXAcJt_R9X0CuVqs1gEP02BwHcEW#scrollTo=oR-YQs

CO

산업인공지능개론_과제_Durable ☆

파일 수정 보기 삽입 런타임 도구 도움말 저장하지 못함

☰

+ 코드 + 텍스트

🔍

더블클릭 또는 Enter 키를 눌러 수정

{x}

✓ [74] !pip install durable_rules
4초

📁

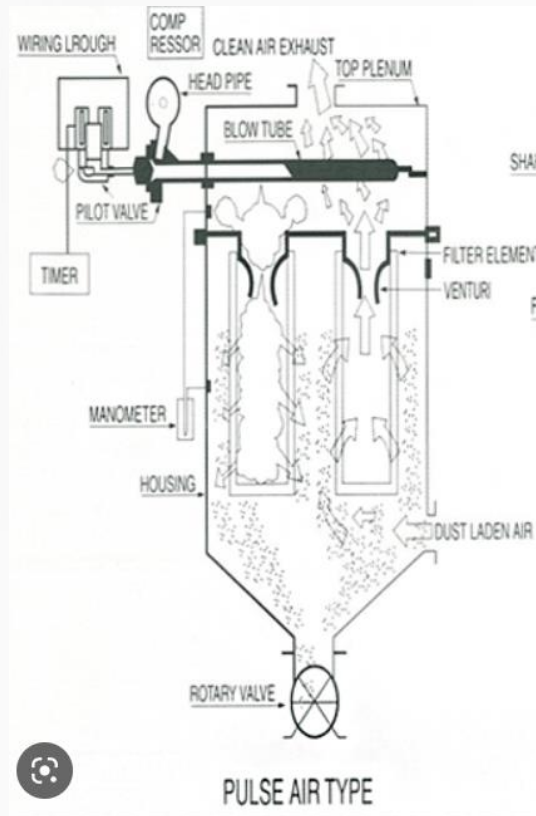
Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>
Requirement already satisfied: durable_rules in /usr/local/lib/python3.9/dist-packages (2.0.28)

✓ [81] with ruleset('stress_rules'):
0초
 @when_all(m.subject == '구조물', m.stress <= 100)
 def stress_ok(c):
 print('경고 기능을 OFF 합니다.')
 @when_all(m.subject == '구조물', m.stress > 100)
 def stress_warning(c):
 print('경고 기능을 ON 합니다.')
post('stress_rules', {'subject': '구조물', 'stress': 80})
post('stress_rules', {'subject': '구조물', 'stress': 120})

경고 기능을 OFF 합니다.
{'sid': '0', 'id': 'sid-0', '\$s': 1}

문제8: 집진기 필터 교환주기의 문제

- **IF** 집진기 배출부와 흡입부의 차압이 교체차압 이하인가 **THEN** 교체 알람을 OFF 한다.
- **IF** 집진기 배출부와 흡입부의 차압이 교체차압 초과인가 **THEN** 교체 알람을 ON 한다



문제8: 집진기 필터 교환주기의 문제

```
Untitled0.ipynb ☆
파일 수정 보기 삽입 런타임 도구 도움말 모든 변경사항이 저장됨

+ 코드 + 텍스트

[1] !pip install durable_rules

Looking in indexes: https://pypi.org/simple, https://us-python.ekg.dev/colab-wheels/public/simple/
Collecting durable_rules
  Downloading durable_rules-2.0.28.tar.gz (57 kB)
----- 57.6/57.6 kB 483.3 kB/s eta 0:00:00
Preparing metadata (setup.py) ... done
Building wheels for collected packages: durable_rules
  Building wheel for durable_rules (setup.py) ... done
  Created wheel for durable_rules: filename=durable_rules-2.0.28-cp39-cp39-linux_x86_64.whl size=203285 sha256=4f24e3988650173e434e8c70051265203a654064455e5ae
  Stored in directory: /root/.cache/pip/wheels/1d/1b/a0/d71690e6081651215bc87359ad03f8127d70a314516560e0e3
Successfully built durable_rules
Installing collected packages: durable_rules
Successfully installed durable_rules-2.0.28

from durable.lang import *

# 규칙 정의
with ruleset('replace_alarm'):

    # 규칙 1: 교체 알람을 OFF 한다.
    @when_all(
        m.p_diff <= m.replace_diff
    )
    def off_alarm(c):
        print("Replace alarm is turned OFF")

    # 규칙 2: 교체 알람을 ON 한다.
    @when_all(
        m.p_diff > m.replace_diff
    )
    def on_alarm(c):
        print("Replace alarm is turned ON")

# 테스트를 위한 데이터
data = {
    'p_diff': 1.2, # 집진기 배출부와 흡입부의 차압
    'replace_diff': 1.5 # 교체 차압
}

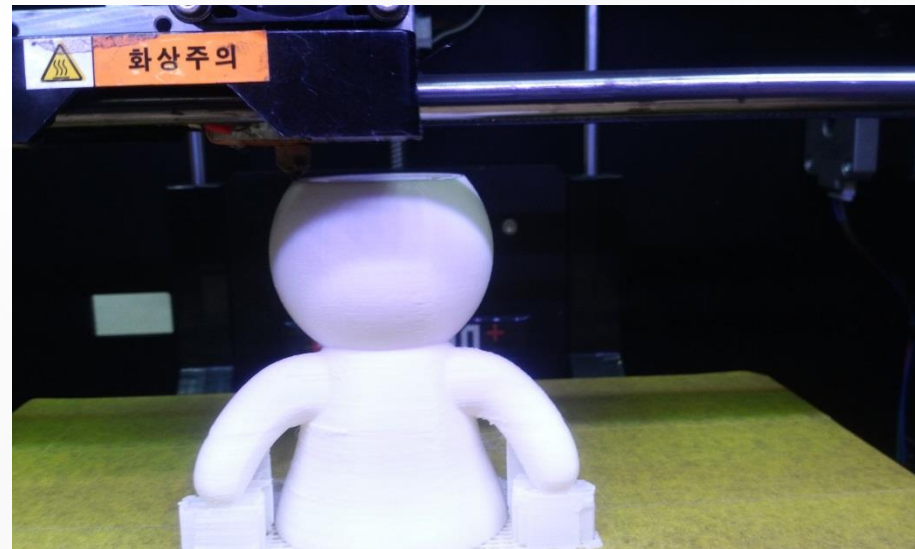
# 규칙 실행
post('replace_alarm', data)

# 규칙 실행 후 결과
# Replace alarm is turned ON

Replace alarm is turned OFF
{'sid': '0', 'id': 'sid-0', '$s': 1}
```

문제9: 스마트 팜 시스템에서의 급수

- **IF** 3D PRINTER G코드상 Z 축 레이어의 적층이 1000 미만인가 **THEN** 계속 작업을 진행한다.
- **IF** 3D PRINTER G코드상 Z 축 레이어의 적층이 1000 초과인가 **THEN** END CODE 를 수행한다.



문제9: 스마트 팜 시스템에서의 급수



산업인공지능개론_과제_Durable ☆

파일 수정 보기 삽입 런타임 도구 도움말 저장하지 못함

+ 코드 + 텍스트



더블클릭 또는 Enter 키를 눌러 수정

{x}



!pip install durable_rules

from durable.lang import *

규칙 엔진 초기화

with ruleset('3D_printer'):

규칙1: 적층이 1000 미만인 경우 계속 작업 진행

@when_all(c.z < 1000)

def continue_job(c):

print('현재 적층:', c.z, '- 작업 계속 진행')

규칙2: 적층이 1000 이상인 경우 작업 종료

@when_all(c.z >= 1000)

def end_job(c):

print('현재 적층:', c.z, '- 작업 종료')

halt()

규칙 엔진 실행

post('3D_printer', {'z': 800})

post('3D_printer', {'z': 1200})

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>
Requirement already satisfied: durable_rules in /usr/local/lib/python3.9/dist-packages (2.0.28)

Exception Traceback (most recent call last)

<ipython-input-91-4427f836c0b9> in <cell line: 6>()

7

<>

문제10: 사출에서의 자동화

- IF 사출 횟수가 3000SHOT 미만인가 THEN 다음 작업을 진행한다.
- IF 사출 횟수가 3000SHOT 이상인가 THEN 작업을 중지하고 완료 메시지를 전송한다.



문제10: 사출에서의 자동화



Untitled1.ipynb ☆

파일 수정 보기 삽입 런타임 도구 도움말 모든 변경사항이 저장됨

+ 코드 + 텍스트



```
[ ]

!pip install durable_rules

from durable.lang import *

with ruleset('injection_molding'):
    @when_all(m.ShotCount < 3000)
    def continue_operation(c):
        print('사출성형기 작동 중입니다.')
        c.assert_fact({'status': 'continue'})

    @when_all(m.ShotCount >= 3000)
    def stop_operation(c):
        print('사출성형기 작동 중지합니다. 작업 완료 메시지를 전송합니다.')
        c.assert_fact({'status': 'stop'})
```

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>

Collecting durable_rules

Downloading durable_rules-2.0.28.tar.gz (57 kB)

57.6/57.6 kB 1.7 MB/s eta 0:00:00

Preparing metadata (setup.py) ... done

Building wheels for collected packages: durable_rules

Building wheel for durable_rules (setup.py) ... done

Created wheel for durable_rules: filename=durable_rules-2.0.28-cp39-cp39-linux_x86_64.whl size=203283 sha256=ec58cbb10dfcc15be7181de8acc1742cfad25d98c633ad7c08325fa49345eff0

Stored in directory: /root/.cache/pip/wheels/1d/1b/a0/d71690e6081651215bc87359ad03f8127d70a314516560e0e3

Successfully built durable_rules

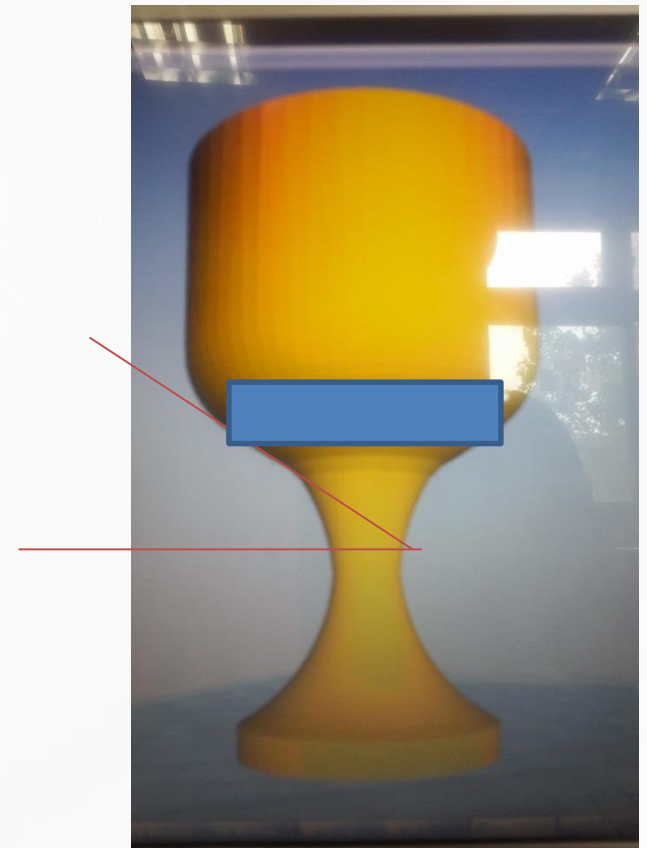
Installing collected packages: durable_rules

Successfully installed durable_rules-2.0.28


Windows ?

문제11: G코드 슬라이서에서의 언더컷 가시화 문제

- **IF** 형상의 특정 위치에서의 지면과의 각도가 90도 이상인가 **THEN** 노란색으로 표시한다
- **IF** 형상의 특정 위치에서의 지면과의 각도가 90도 미만인가 **THEN** 파란색으로 표시한다 (언더컷)



문제11: G코드 슬라이서에서의 언더컷 가시화 문제

 Untitled1.ipynb ☆

파일 수정 보기 삽입 런타임 도구 도움말

+ 코드 + 텍스트

⋮

🔍

{x}

📁

✓ 4초

▶

```
!pip install durable_rules

from durable.lang import *

# 형상의 특정 위치에서의 지면과의 각도 변수 정의
angle = 100

# 색상 변수 정의
color = None

# 규칙 엔진 정의
with ruleset('color_rule'):

    # 각도가 90도 이상인 경우 노란색으로 표시
    @when_all(m.angle >= 90)
    def yellow(c):
        global color
        color = 'yellow'
        print('Angle is greater than or equal to 90, color is', color)

    # 각도가 90도 미만인 경우 파란색으로 표시
    @when_all(m.angle < 90)
    def blue(c):
        global color
        color = 'blue'
        print('Angle is less than 90, color is', color)

# 규칙 실행
post('color_rule', {'angle': angle})
```

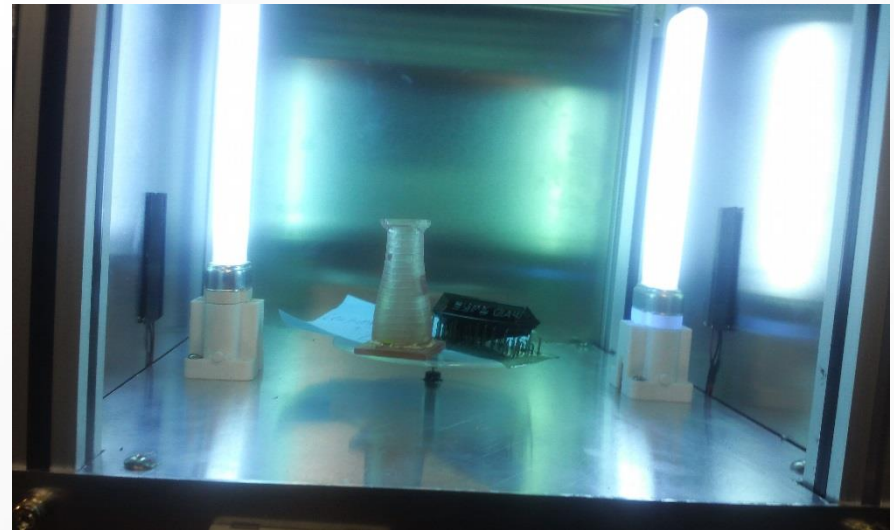
<>

☰

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>
Requirement already satisfied: durable_rules in /usr/local/lib/python3.9/dist-packages (2.0.28)
Angle is greater than or equal to 90, color is yellow
{'sid': '0', 'id': 'sid-0', '\$s': 1}

문제12: 산업용 경화 장치에서의 경화율 문제

- IF 물체의 경화시간이 120분 미만인가 THEN UV광을 점등하고 턴테이블을 회전한다
- IF 물체의 경화시간이 120분 이상인가 THEN UV광을 소등하고 턴테이블을 정지한다



문제12: 산업용 경화 장치에서의 경화율 문제



Untitled0.ipynb ☆

파일 수정 보기 삽입 런타임 도구 도움말

+ 코드 + 텍스트



✓ 17
줄

```
[1] !pip install durable_rules
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.ekg.dev/colab-wheels/public/simple/
Collecting durable_rules
  Downloading durable_rules-2.0.28.tar.gz (57 kB)
----- 57.6/57.6 kB 483.3 kB/s eta 0:00:00
Preparing metadata (setup.py) ... done
Building wheels for collected packages: durable_rules
  Building wheel for durable_rules (setup.py) ... done
  Created wheel for durable_rules: filename=durable_rules-2.0.28-cp39-cp39-linux_x86_64.whl size=203285 sha256=4f24e3988650f73e434e8c70051265203a654064455e5a66e51f973571eb3bce
  Stored in directory: /root/.cache/pip/wheels/1d/1b/a0/d71690e6081651215bc87359ad0318127d70a314516560e0e3
Successfully built durable_rules
Installing collected packages: durable_rules
Successfully installed durable_rules-2.0.28
```

✓ 0
줄

```
[6] from durable.lang import *

with ruleset('rule_set'):
    @when_all(m.subject == '물체', m.time_to_harden < 120)
    def rule1(c):
        print('물체의 경화시간이 120분 미만입니다. UV광을 점등하고 턴테이블을 회전합니다.')

    @when_all(m.subject == '물체', m.time_to_harden >= 120)
    def rule2(c):
        print('물체의 경화시간이 120분 이상입니다. UV광을 소등하고 턴테이블을 정지합니다.')
```

✓ 0
줄

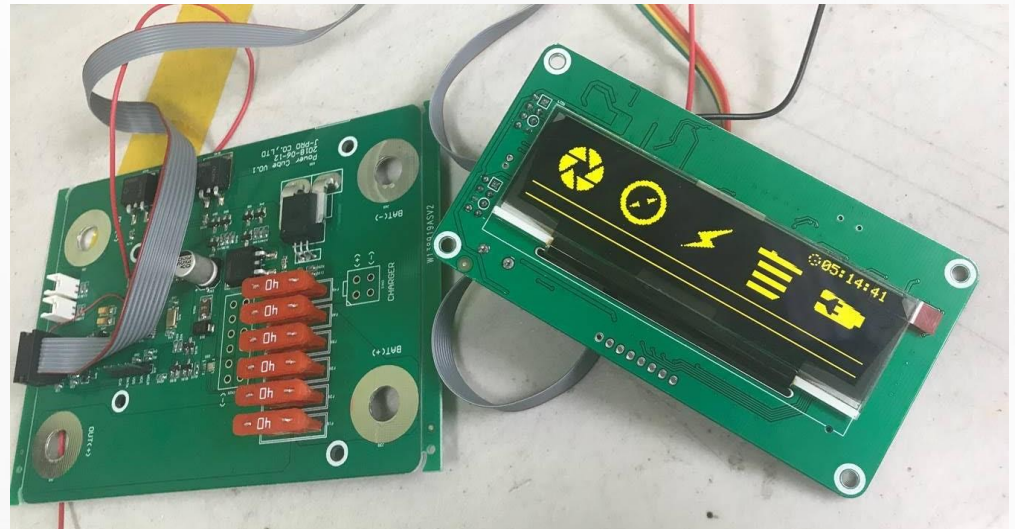
```
data = {'subject': '물체', 'time_to_harden': 90}
post('rule_set', data)

data = {'subject': '물체', 'time_to_harden': 150}
post('rule_set', data)
```

```
물체의 경화시간이 120분 미만입니다. UV광을 점등하고 턴테이블을 회전합니다.
{'sid': '0', 'id': 'sid-0', '$s': 1}
```

문제13: 회로에서의 OLED 경고문 도출

- IF 배터리 전압이 14V 미만인가 THEN 기존 코드에 따라 작동한다
- IF 배터리 전압이 14V 초과인가 THEN 기존 코드와 별개로 경고문구를 작동한다



문제13: 회로에서의 OLED 경고문 도출

Untitled3.ipynb ☆

파일 수정 보기 삽입 런타임 도구 도움말

+ 코드 + 텍스트

[1] !pip install durable_rules

Looking in indexes: <https://pypi.org/simple>, <https://us-python.ekg.dev/colab-wheels/public/simple/>
Collecting durable_rules
Downloading durable_rules-2.0.28.tar.gz (57 kB)
----- 57.6/57.6 kB 1.9 MB/s eta 0:00:00
Preparing metadata (setup.py) ... done
Building wheels for collected packages: durable_rules
Building wheel for durable_rules (setup.py) ... done
Created wheel for durable_rules: filename=durable_rules-2.0.28-cp39-cp39-linux_x86_64.whl size=203273 sha256=1628441ce41a5f1558e8c26856c5c19d71b0643181382b3287ecc1681fdea202
Stored in directory: /root/.cache/pip/wheels/1d/1b/a0/d71690e6081651215bc87359ad03f8127d70a314516560e0e3
Successfully built durable_rules
Installing collected packages: durable_rules
Successfully installed durable_rules-2.0.28

from durable.lang import *

with ruleset('battery_rules'):

 # 14V 미만인 경우
 @when_all(m.voltage < 14)
 def low_voltage(c):
 print('기존 코드에 따라 작동합니다.')
 # 14V 초과인 경우
 @when_all(m.voltage > 14)
 def high_voltage(c):
 print('경고문구를 작동합니다.')
engine = post('battery_rules', {})

배터리 전압이 13.5V인 경우
engine.assert_fact('battery_rules', {'voltage': 13.5})

배터리 전압이 14.5V인 경우
engine.assert_fact('battery_rules', {'voltage': 14.5})

MessageNotHandledException Traceback (most recent call last)
<ipython-input-2-0ec735704f2f> in <cell line: 15>()
13 print('경고문구를 작동합니다.')14
--> 15 engine = post('battery_rules', {})
16
17 # 배터리 전압이 13.5V인 경우

문제14: 경면가공물의 클리닝

- IF 가공물의 표면반사율이 80%미만인가 THEN 초음파 세척을 계속 실시한다
- IF 가공물의 표면반사율이 80%이상인가 THEN 초음파 세척을 계속 정지한다



문제14: 경면가공물의 클리닝

Untitled4.ipynb ☆

파일 수정 보기 삽입 런타임 도구 도움말 모든 변경사항이 저장됨

+ 코드 + 텍스트

+ 코드 + 텍스트

```
!pip install durable_rules

from durable.engine import Engine

# Rule 선언
with DurableEngine() as engine:

    @engine.rule('Cleaning Rule')
    def ultrasonic_cleaning(c):
        if c.data['surface_reflectance'] < 80:
            print("Performing ultrasonic cleaning")
            c.assert_('cleaning_in_progress')
        else:
            print("Stopping ultrasonic cleaning")
            c.retract_('cleaning_in_progress')

# Rule 실행을 위한 데이터 입력
data = {'surface_reflectance': 75}

# 실행 결과 확인
engine.run_all(data)
```

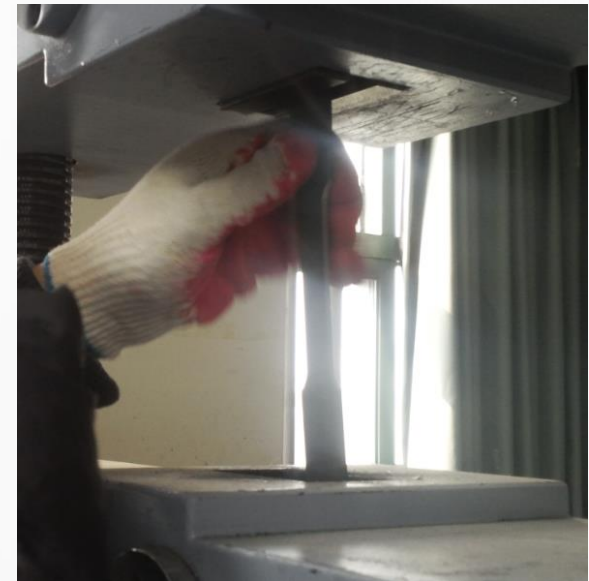
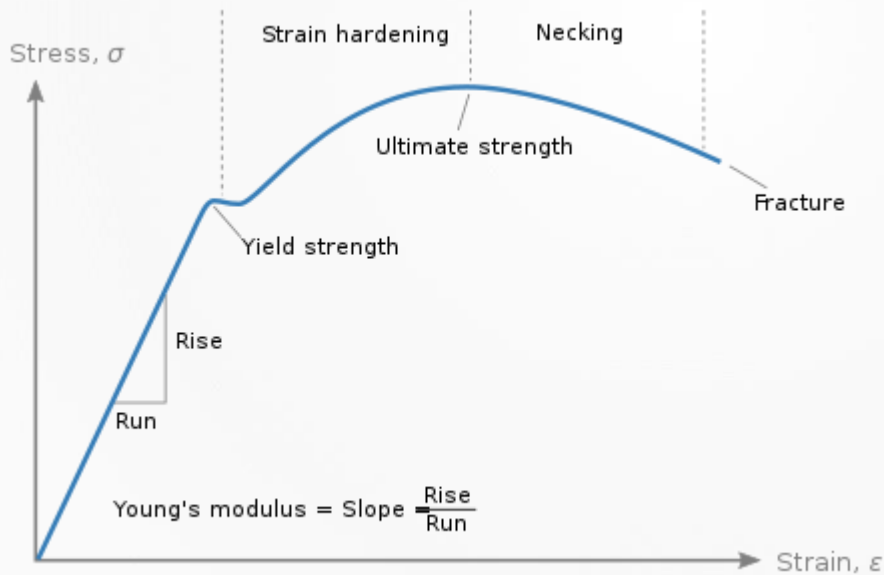
Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>
Requirement already satisfied: durable_rules in /usr/local/lib/python3.9/dist-packages (2.0.28)

```
ImportError                                Traceback (most recent call last)
<ipython-input-5-a3441fc11716> in <cell line: 3>()
      1 get_ipython().system('pip install durable_rules')
      2
----> 3 from durable.engine import Engine
      4
      5 # Rule 선언

ImportError: cannot import name 'Engine' from 'durable.engine' (/usr/local/lib/python3.9/dist-packages/durable/engine.py)
```

문제15: 인장강도 시험(UTM)에서의 활용1

- **IF** 스트레인에 증가에 따라 스트레스가 지속적인 증가를 보이는가 **THEN** 계속 인장한다
- **IF** 스트레인에 증가에 따라 스트레스가 지속적인 증가후 감소를 보이는가 **THEN** 극한점 통과 문구



문제15: 인장강도 시험(UTM)에서의 활용1



Untitled5.ipynb

파일 수정 보기 삽입 런타임 도구 도움말

+ 코드 + 텍스트

[6] !pip install durable_rules

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>
Requirement already satisfied: durable_rules in /usr/local/lib/python3.9/dist-packages (2.0.28)

```
from durable.lang import *
from durable.lang import durablerequest

with ruleset('stress_rules'):

    # 첫 번째 규칙: 스트레인 증가에 따라 스트레스가 지속적으로 증가하는 경우
    @when_all(+m.stress, m.strain > 0)
    def increase_stress(c):
        print('계속 인장한다 메세지 송출')
        c.reset() # 줄 실행 후, 모든 줄을 초기화

    # 두 번째 규칙: 스트레인 증가에 따라 스트레스가 지속적으로 증가한 후 감소하는 경우
    @when_all(+m.stress, m.strain > 0, m.stress < durablerequest.previous('stress'))
    def pass_extreme_point(c):
        print('극한점 통과 문구 송출')
        c.reset() # 줄 실행 후, 모든 줄을 초기화
```

```
ImportError                                Traceback (most recent call last)
<ipython-input-8-a646996e8393> in <cell line: 2>()
      1 from durable.lang import *
----> 2 from durable.lang import durablerequest
      3
      4 with ruleset('stress_rules'):
      5
```

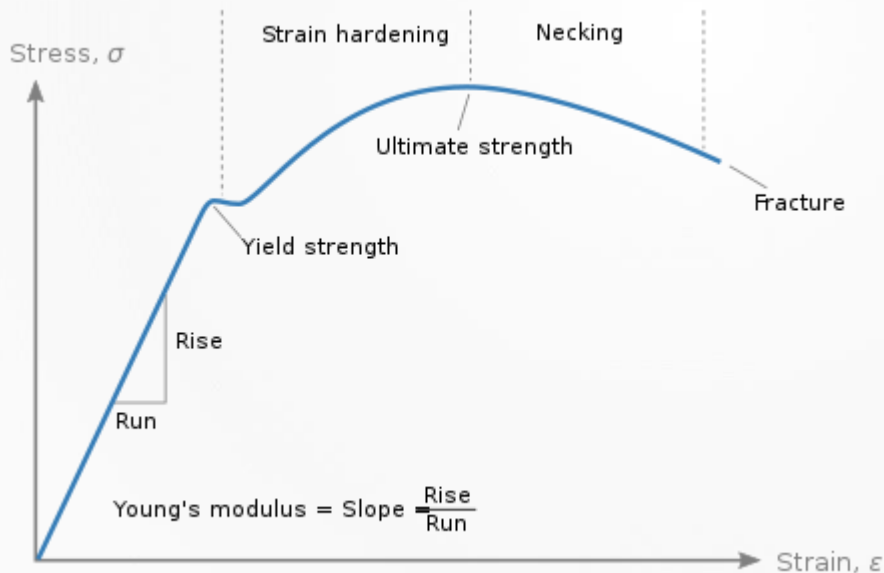
ImportError: cannot import name 'durablerequest' from 'durable.lang' (/usr/local/lib/python3.9/dist-packages/durable/lang.py)

NOTE: If your import is failing due to a missing package, you can manually install dependencies using either !pip or !apt.

To view examples of installing some common dependencies, click the "Open Examples" button below.

문제16: 인장강도 시험(UTM)에서의 활용2

- **IF** 스트레인에 증가에 따라 스트레스가 지속적인 감지 되는가 **THEN** 계속 인장한다
- **IF** 스트레인에 증가에 따라 스트레스가 지속적인 감지가 불가능한가 **THEN** 인장 중지하며 파단 문구 도출



문제16: 인장강도 시험(UTM)에서의 활용2

CO Untitled5.ipynb ☆

파일 수정 보기 삽입 런타임 도구 도움말 모든 변경사항이 저장됨

+ 코드 + 텍스트

5초 [9] !pip install durable_rules

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>
Requirement already satisfied: durable_rules in /usr/local/lib/python3.9/dist-packages (2.0.28)

0초 from durable.lang import *

with ruleset('stress_rules'):

 @when_all(+m.stress)
 def stress_detected(c):
 if c.m['strain'] > c.m['previous_strain']:
 print("Continuously reinforcing")
 c.m['previous_strain'] = c.m['strain']
 else:
 print("Stop reinforcing, fracture detected")
 c.m['previous_strain'] = 0
 c.m['fracture_detected'] = True

with ruleset('stress_rules'):
 assert_fact('stress', {'strain': 10, 'previous_strain': 5})
 assert_fact('stress', {'strain': 15, 'previous_strain': 10})
 assert_fact('stress', {'strain': 20, 'previous_strain': 15})
 assert_fact('stress', {'strain': 15, 'previous_strain': 20})
 assert_fact('stress', {'strain': 10, 'previous_strain': 15})

Exception Traceback (most recent call last)
<ipython-input-13-cafc07a59f69> in <cell line: 15>()
 14
--> 15 with ruleset('stress_rules'):
 16 assert_fact('stress', {'strain': 10, 'previous_strain': 5})
 17 assert_fact('stress', {'strain': 15, 'previous_strain': 10})
 18 assert_fact('stress', {'strain': 20, 'previous_strain': 15})

2 frames
/usr/local/lib/python3.9/dist-packages/durable/engine.py in register_rulesets(self, ruleset_definitions)
 888 for ruleset_name, ruleset in rulesets.items():
 889 if ruleset_name in self._ruleset_directory:
--> 890 raise Exception('Ruleset with name {0} already registered'.format(ruleset_name))
 891 else:
 892 self._ruleset_directory[ruleset_name] = ruleset

문제17: 산업용 디스펜서의 자동정지

- IF 용기내 주입량이 10L 미만인가 THEN 계속 주입한다
- IF 용기내 주입량이 10L 이상인가 THEN 주입 정지한다
- IF 용기1내 주입량이 10L 이상인가 AND 용기2내 주입량이 10L 이상인가 . . .
용기6내 주입량이 10L 이상인가 THEN 완료 문구를 전송한다



문제17: 산업용 디스펜서의 자동정지

```
Untitled6.ipynb ☆
파일 수정 보기 삽입 런타임 도구 도움말 모드 변경사항이 저장됨

+ 코드 + 텍스트

# 용기내 주입량 초기화
tank_volume = 0

# 규칙 엔진 초기화
durable_rules_engine = DurableEngine()

# 규칙 정의
with ruleset('tank_rules'):

    # 용기내 주입량이 10L 미만인 경우 계속해서 주입한다.
    @when_all(m.volume < 10)
    def continue_injection(c):
        global tank_volume
        tank_volume += c.volume
        print(f"현재 용기내 주입량: {tank_volume}L")
        c.restart()

    # 용기내 주입량이 10L 이상인 경우 주입을 정지한다.
    @when_all(m.volume >= 10)
    def stop_injection(c):
        global tank_volume
        tank_volume += c.volume
        print(f"현재 용기내 주입량: {tank_volume}L")
        print("주입을 중지합니다.")
        c.halt()

# 규칙 엔진 실행
durable_rules_engine.post('tank_rules', {'volume': 5})
durable_rules_engine.post('tank_rules', {'volume': 3})
durable_rules_engine.post('tank_rules', {'volume': 2})
durable_rules_engine.post('tank_rules', {'volume': 10})

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting durable_rules
  Downloading durable_rules-2.0.28.tar.gz (57 kB)
----- 57.6/57.6 kB 2.8 MB/s eta 0:00:00
Preparing metadata (setup.py) ... done
Building wheels for collected packages: durable_rules
  Building wheel for durable_rules (setup.py) ... done
  Created wheel for durable_rules: filename=durable_rules-2.0.28-cp39-cp39-linux_x86_64.whl size=203280 sha256=49d5a1bf51a27073e0b7de5872b38b3152148471da29d6c4aa5ee8160d48d294
  Stored in directory: /root/.cache/pip/wheels/1d/1b/a0/d71690e6081651215bc87359ad03f8127d70a314516560e0e3
Successfully built durable_rules
Installing collected packages: durable_rules
Successfully installed durable_rules-2.0.28

NameError                                Traceback (most recent call last)
<ipython-input-1-5cad08c351e2> in <cell line: 9>()
      7
      8 # 규칙 엔진 초기화
----> 9 durable_rules_engine = DurableEngine()
     10
     11 # 규칙 정의
```

문제18: 엔진오일 경고등 점등

- IF 엔진내 압력 센서의 값이 A Mpa 이상인가 THEN 경고등을 점등하지 않는다
- IF 엔진내 압력 센서의 값이 A Mpa 미만인가 THEN 경고등을 점등한다
- IF 엔진내 압력 센서의 값이 A Mpa 이상인가 OR 기포가 검출 되었는가 THEN 경고등을 점등한다



문제18: 엔진오일 경고등 점등

Untitled6.ipynb ☆

파일 수정 보기 삽입 런타임 도구 도움말 모든 변경사항이 저장됨

+ 코드 + 텍스트

```
!pip install durable_rules

from durable.lang import *

# 엔진 내 압력 센서 값 초기화
engine_pressure = 0.0

# 기포 검출 여부 초기화
bubble_detected = False

# 규칙 엔진 정의
with ruleset('engine_monitoring'):

    # 엔진내 압력 센서의 값이 A Mpa 이상인가 THEN 경고등을 점등하지 않는다
    @when_all(m.pressure >= engine_pressure)
    def no_warning(c):
        print("압력이 안전합니다.")

    # 엔진내 압력 센서의 값이 A Mpa 미만인가 THEN 경고등을 점등한다
    @when_all(m.pressure < engine_pressure)
    def warning(c):
        print("압력이 낮습니다. 경고등이 켜집니다.")

    # 엔진내 압력 센서의 값이 A Mpa 이상인가 OR 기포가 검출 되었는가 THEN 경고등을 점등한다
    @when_any(all(m.pressure >= engine_pressure), all(m.bubbles_detected == True))
    def warning_bubbles(c):
        print("압력이나 기포 검출 문제가 있습니다. 경고등이 켜집니다.")

# 규칙 엔진 초기화
engine = get_engine('engine_monitoring')

# 규칙 엔진 실행
assert_fact(engine, 'm', {'pressure': 2.0, 'bubbles_detected': False})
assert_fact(engine, 'm', {'pressure': 1.0, 'bubbles_detected': False})
assert_fact(engine, 'm', {'pressure': 2.0, 'bubbles_detected': True})

# 규칙 엔진 종료
engine.halt()
```

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>
Requirement already satisfied: durable_rules in /usr/local/lib/python3.9/dist-packages (2.0.28)

NameError Traceback (most recent call last)
 <ipython-input-2-21b3f8158ed7> in <cell line: 30>()
 28
 29 # 규칙 엔진 초기화
--> 30 engine = get_engine('engine_monitoring')
 31
 32 # 규칙 엔진 실행

문제19: 프린터의 용지 없음 문구

- **IF** 프린터 내부의 포토 인터럽터 센서가 용지를 감지하는가 **THEN** 문구 도출하지 않는다
- **IF** 프린터 내부의 포토 인터럽터 센서가 용지를 감지하지 않는가 **THEN** 용지없음 문구 도출한다



문제19: 프린터의 용지 없음 문구



Untitled6.ipynb ☆

파일 수정 보기 삽입 런타임 도구 도움말 모든 변경사항이 저장됨

+ 코드 + 텍스트



✓ [8] !pip install durable_rules

4초



Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>
Requirement already satisfied: durable_rules in /usr/local/lib/python3.9/dist-packages (2.0.28)



✓ 0초 ▶

```
from durable.lang import *

with ruleset('printer_rules'):
    # 기존 룰 초기화

    def reset():
        reset_ruleset('printer_rules')

    # 포토 인터럽터 센서가 용지를 감지하는 경우
    @when_all(m.predicate == 'detects_paper')
    def no_output(c):
        print('문구 도출하지 않는다')

    # 포토 인터럽터 센서가 용지를 감지하지 않는 경우
    @when_all(m.predicate == 'no_paper')
    def has_output(c):
        print('용지없음 문구 도출한다')

# 룰 엔진 시작
post('printer_rules', {'predicate': 'detects_paper'})
post('printer_rules', {'predicate': 'no_paper'})
```

문구 도출하지 않는다
용지없음 문구 도출한다
{'sid': '0', 'id': 'sid-0', '\$s': 1}

문제20: 낙석 위험지역의 센싱

- **IF** 1점과 2점 간 연결된 스트레인 게이지의 저항변화값이 A 이하인가 **THEN** 경고문구 도출하지 않는다
- **IF** 1점과 2점 간 연결된 스트레인 게이지의 저항변화값이 A 초과인가 **THEN** 경고문구 전송한다



문제20: 낙석 위험지역의 센싱

Untitled6.ipynb ☆

파일 수정 보기 삽입 런타임 도구 도움말

+ 코드 + 텍스트

✓ (8) !pip install durable_rules

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>
Requirement already satisfied: durable_rules in /usr/local/lib/python3.9/dist-packages (2.0.28)

▶ !pip install durable_rules

```
from durable.lang import *
```

```
# 규칙 정의
```

```
with ruleset('strain_gauge'):
```

```
    def do_not_send_warning(c):  
        print('경고문구 호출하지 않는다')
```

```
    def send_warning(c):  
        print('경고문구 전송한다')
```

```
# 규칙 활성화
```

```
load('strain_gauge')
```

```
# 데이터 입력 및 규칙 실행
```

```
post('strain_gauge', {'sensor1': {'resistance': 0.5}, 'sensor2': {'resistance': 0.7}})
```

Looking in indexes: <https://pypi.org/simple>, <https://us-python.pkg.dev/colab-wheels/public/simple/>
Requirement already satisfied: durable_rules in /usr/local/lib/python3.9/dist-packages (2.0.28)

NameError Traceback (most recent call last)

<jupyter-input-14-c6598cffff03> in <cell line: 18>()
16

17

```
# 규칙 활성화
```

```
----> 18 load('strain_gauge')
```

19

```
20 # 데이터 입력 및 규칙 실행
```

NameError: name 'load' is not defined

SEARCH STACK OVERFLOW

감사합니다