به نام خدا

تهیه کننده: ابراهیم صدیقی شماره دانشجویی: ۹۹۳۱۰۹۸

بخش اول:

توضیح فرمت ورودی و خروجی:

```
/usr/bin/python3 /home/lwall/Documents/uni/1402-1403-second-term/Soft test/bonus/code/q1.py
----- Implementation of input domain modeler program ----
    1) Add a Characteristic
    2) Add a Abstract Block
    3) Choose a Mode
    00) Exit
    Enter: 1
    Pls type a characteristic in the following format:
        <name>=<characteristic>
        example:
            A=hair color
    0) back
    Enter: A=hair color
    1) Add a Characteristic
    2) Add a Abstract Block
    3) Choose a Mode
    00) Exit
    Enter: 2
    Pls type a Abstract Block in the following format:
        <characteristic name>=(<block>, <block>, ...)
        example:
            A=(blue, black, brown, yellow)
    0) back
    Enter: A=(blue, black, brown, yellow)
```

```
    Enter: A=(blue, black, brown, yellow)
    1) Add a Characteristic
    2) Add a Abstract Block
    3) Choose a Mode
    00) Exit
    Enter: 2
    Pls type a Abstract Block in the following format:
        <characteristic name>=(<block>, <block>, ...)
        example:
            A=(blue, black, brown, yellow)
    0) back
    Enter: B=(cs, swe, ce, math, ist, st)
    1) Add a Characteristic
    2) Add a Abstract Block
    3) Choose a Mode
    00) Exit
    Enter: 3
    Pls type a Mode:
    1) ACoC
    2) ECC
    3) BCC
    4) MBCC
    0) back
    Enter: 1
[('blue', 'cs'), ('blue', 'swe'), ('blue', 'ce'), ('blue', 'math'), ('blue', 'ist'), ('blue', 'st'), ('black', 'cs'), ('black', 'swe'), ('black', 'ce'), ('black', 'math'), ('black', 'ist'), ('black', 'st'), ('brown',
 'cs'), ('brown', 'swe'), ('brown', 'ce'), ('brown', 'math'), ('brown', 'ist'), ('brown', 'st'), ('yellow', 'cs'), ('yellow', 'swe'), ('yellow', 'ce'), ('yellow', 'math'), ('yellow', 'ist'), ('yellow', 'st')]
Total is 24
```

همان طور که در تصویر مشخص است. یک رابط کاربری بسیار ساده طراحی شده و همان جا خروجی را به کاربر نشان داده میشود

عکس از سه مد کاری:

Bcc:

```python
from q1 import InputRangeModel

# Example of how to use the InputRangeModel class
model = InputRangeModel()
model.add_characteristic("A=hair color")
model.add_characteristic("B=major")
model.add_characteristic("C=number")
# model.add_characteristic("D=num in text")
# model.add_characteristic("E=total")
model.add_block_abstract("A=(blue, black, brown, yellow)")
model.add_block_abstract("B=(cs, swe, ce, math, ist, st)")
model.add_block_abstract("C=(1, 2, 3, 4)")
# model.add_block_abstract("D=(one, two, three)")
# model.add_block_abstract("E=(29, 54)")

# Call different working modes
if model.check_characteristic_and_block():

    model.add_base_choice("base=(black, math, 3)")
    print(model.working_mode_bcc())

    ##
    # model.add_multiple_base_choice("base1=(black, math, 3, three, 54)")
    # model.add_multiple_base_choice("base2=(brown, ce, 3, one, 54)")
    # model.working_mode_mbcc()

    # done
    # model.working_mode_ecc()

    # done
    # model.working_mode_acoc()
```

```
/usr/bin/python3 /home/lwall/Documents/uni/1402-1403-second-term/Soft test/bonus/code/q3.py
Generated BCC Test Cases for ('black', 'math', '3'):
[('blue', 'math', '3'), ('brown', 'math', '3'), ('yellow', 'math', '3'), ('black', 'cs', '3'), ('black', 'swe', '3'), ('black', 'ce', '3'), ('black', 'ist', '3'), ('black', 'st', '3'), ('black', 'math', '1'), ('black',
  'math', '2'), ('black', 'math', '4')]
lenth is 11
```

Mbcc:

```python
from q1 import InputRangeModel

# Example of how to use the InputRangeModel class
model = InputRangeModel()
model.add_characteristic("A=hair color")
model.add_characteristic("B=major")
model.add_characteristic("C=number")
# model.add_characteristic("D=num in text")
# model.add_characteristic("E=total")
model.add_block_abstract("A=(blue, black, brown, yellow)")
model.add_block_abstract("B=(cs, swe, ce, math, ist, st)")
model.add_block_abstract("C=(1, 2, 3, 4)")
# model.add_block_abstract("D=(one, two, three)")
# model.add_block_abstract("E=(29, 54)")

# Call different working modes
if model.check_characteristic_and_block():

    # model.add_base_choice("base=(black, math, 3)")
    # print(model.working_mode_bcc())

    ##
    model.add_multiple_base_choice("base1=(black, math, 3)")
    model.add_multiple_base_choice("base2=(brown, ce, 2)")
    print(model.working_mode_mbcc())

    # done
    # model.working_mode_ecc()

    # done
    # model.working_mode_acoc()
```

```
/usr/bin/python3 /home/lwall/Documents/uni/1402-1403-second-term/Soft test/bonus/code/q3.py
["Generated BCC Test Cases for ('black', 'math', '3'): \n[('blue', 'math', '3'), ('brown', 'math', '3'), ('yellow', 'math', '3'), ('black', 'cs', '3'), ('black', 'swe', '3'), ('black', 'ce', '3'), ('black', 'ist',
 '3'), ('black', 'st', '3'), ('black', 'math', '1'), ('black', 'math', '2'), ('black', 'math', '4')]\nlenth is 11" "Generated BCC Test Cases for ('brown', 'ce', '2'): \n[('blue', 'ce', '2'), ('black', 'ce', '2'),
 ('yellow', 'ce', '2'), ('brown', 'cs', '2'), ('brown', 'swe', '2'), ('brown', 'math', '2'), ('brown', 'ist', '2'), ('brown', 'st', '2'), ('brown', 'ce', '1'), ('brown', 'ce', '3'), ('brown', 'ce', '4')]\nlenth is 11"]

Process finished with exit code 0
```

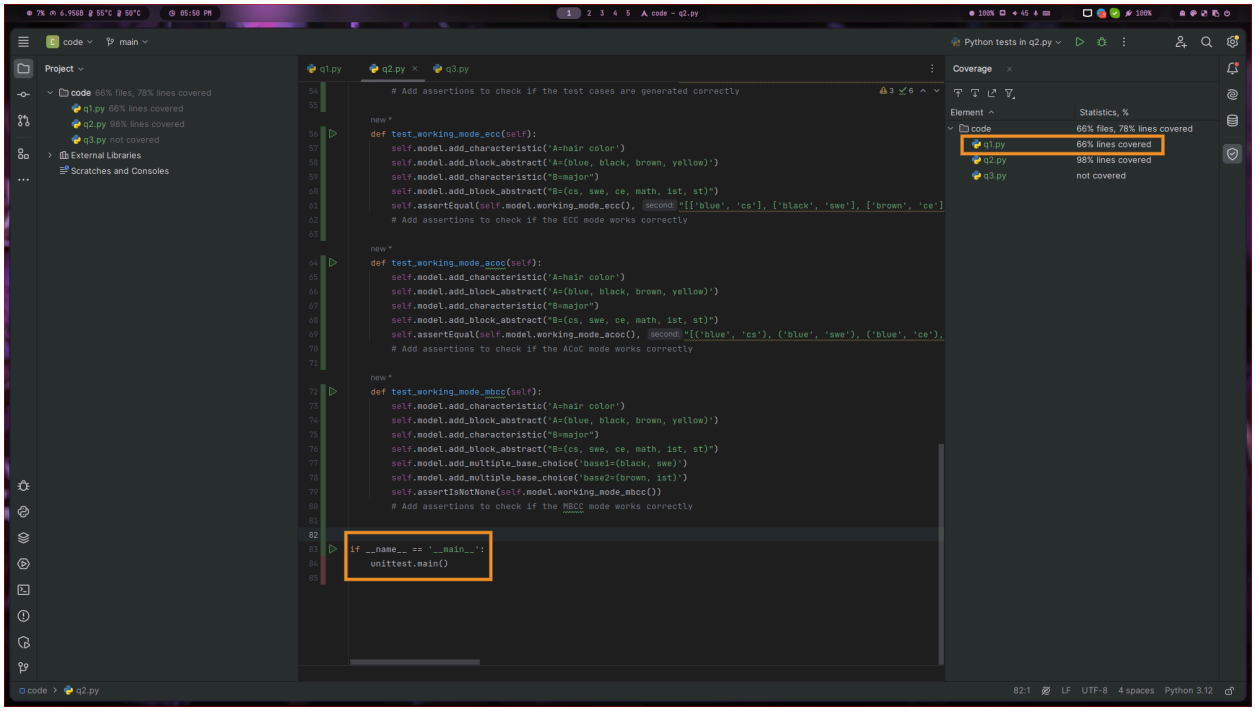
Ecc:

```python
from q1 import InputRangeModel

# Example of how to use the InputRangeModel class
model = InputRangeModel()
model.add_characteristic("A=hair color")
model.add_characteristic("B=major")
model.add_characteristic("C=number")
# model.add_characteristic("D=num in text")
# model.add_characteristic("E=total")
model.add_block_abstract("A=(blue, black, brown, yellow)")
model.add_block_abstract("B=(cs, swe, ce, math, ist, st)")
model.add_block_abstract("C=(1, 2, 3, 4)")
# model.add_block_abstract("D=(one, two, three)")
# model.add_block_abstract("E=(29, 54)")

# Call different working modes
if model.check_characteristic_and_block():

    # model.add_base_choice("base=(black, math, 3)")
    # print(model.working_mode_bcc())

    ##
    # model.add_multiple_base_choice("base1=(black, math, 3)")
    # model.add_multiple_base_choice("base2=(brown, ce, 2)")
    # print(model.working_mode_mbcc())

    # done
    print(model.working_mode_ecc())

    # done
    # model.working_mode_acoc()
```

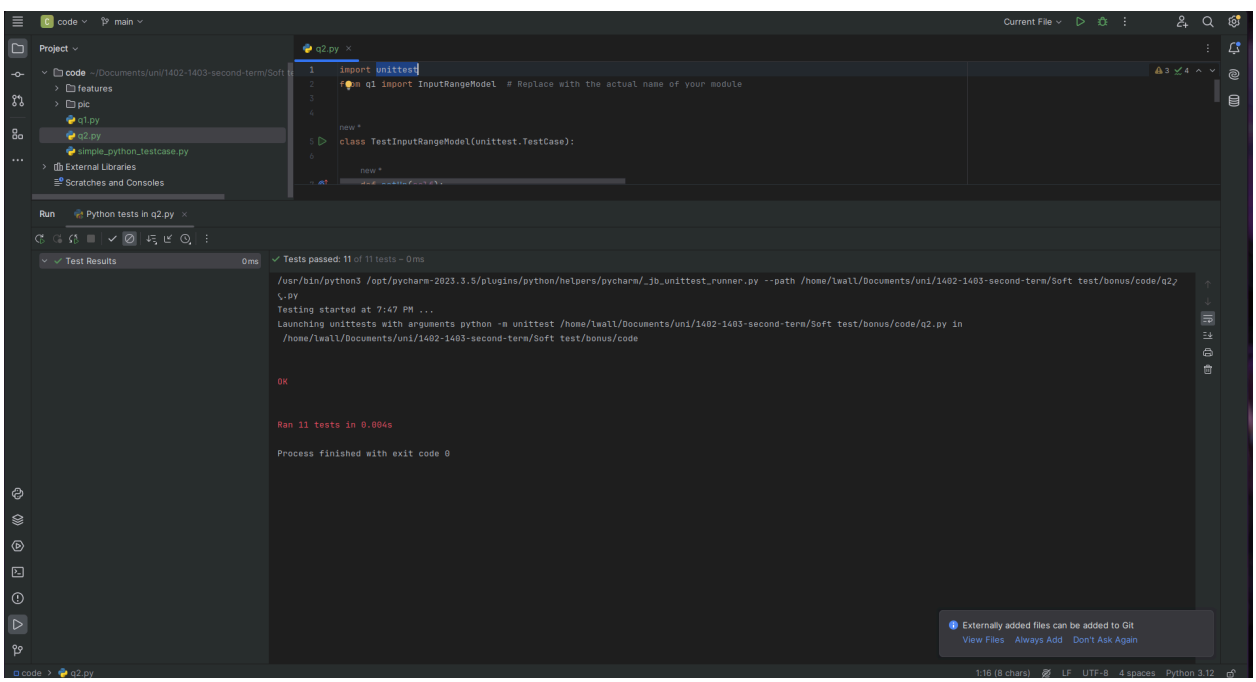
```
/usr/bin/python3 /home/lwall/Documents/uni/1402-1403-second-term/Soft test/bonus/code/q3.py
[['blue', 'cs', '1'], ['black', 'swe', '2'], ['brown', 'ce', '3'], ['yellow', 'math', '4'], ['blue', 'ist', '1'], ['black', 'st', '2']]
Total is 6

Process finished with exit code 0
```

بخش دوم:

در این بخش از ما پیاده سازی unit test را میخواهند که با کتابخانه unit test در پایتون قابل پیاده سازی است. همچنین به coverage ۶۵ درصدی از کد بخش اول رسیدیم.



و همچنین عکس تایدد تست هایمان



بخش سوم:

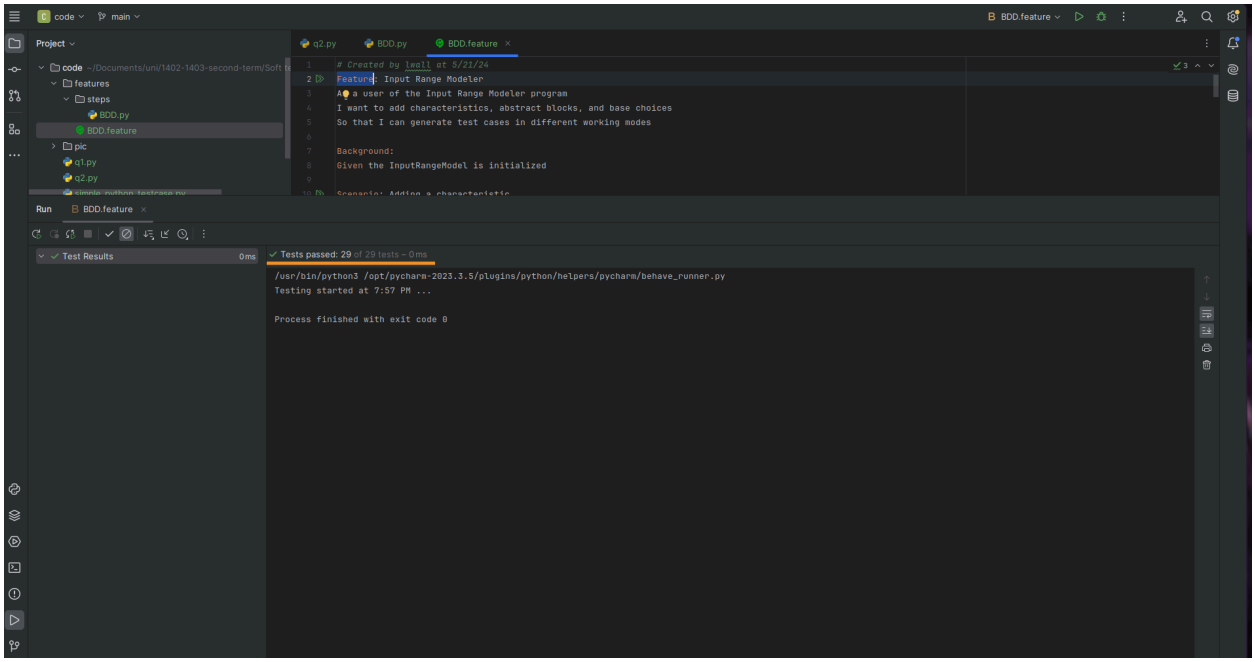
در این بخش ابتدا برای پایتون کتابخانه behave را نصب کرده تا بتوان از cucumber در زبان پایتون استفاده کرد

ابن کتابخانه یک فایل یا فایل های اصلی .feature دارد که با استفاده از کلید واژه هایی همچون , Scenario, Feature , Given , When Then تست کیس های خودمان را تعریف میکنیم. سپس درون فولدری به اسم steps یک یا چند فایل پایتون هست که به فایل ,feature وصل است و تسک ها را پیاده سازی میکند

```
# Created by lwall at 5/21/24
Feature: Input Range Modeler
As a user of the Input Range Modeler program
I want to add characteristics, abstract blocks, and base choices
So that I can generate test cases in different working modes

Background:
Given the InputRangeModel is initialized

Scenario: Adding a characteristic
When I add a characteristic "A=hair color"
Then the characteristic "A" should be "hair color" in the model

Scenario: Adding an abstract block
When I add an abstract block "A=(blue, black, brown, yellow)"
Then the abstract block "A" should contain "blue, black, brown, yellow"

Scenario: Checking characteristic and block compatibility
Given I have added a characteristic "A=hair color"
And I have added an abstract block "A=(blue, black, brown, yellow)"
When I check characteristic and block compatibility
Then the result should be True

Scenario: Generating BCC test cases
Given I have added a characteristic "A=hair color"
And I have added an abstract block "A=(blue, black, brown, yellow)"
And I have added a base choice "base=(black, math, 3, three)"
When I generate BCC test cases
Then I should get a list of test cases with length greater than 0

Scenario: Generating ECC test cases
Given I have added multiple abstract blocks
When I generate ECC test cases
Then I should get a list of test cases that are not None

Scenario: Generating ACoC test cases
Given I have added multiple abstract blocks
When I generate ACoC test cases
Then I should get a list of all possible combinations from the abstract blocks

Scenario: Generating MBCC test cases
Given I have added multiple base choices
```

```python
from behave import given, when, then

from q1 import InputRangeModel


####################################
new *
@given('the InputRangeModel is initialized')
def step_impl(context):
    context.model = InputRangeModel()


new *
@when('I add a characteristic "A=hair color"')
def step_impl(context):
    context.result = context.model.add_characteristic("A=hair color")


new *
@then('the characteristic "A" should be "hair color" in the model')
def step_impl(context):
    assert f"Expected: {'hair color'}, Actual: {context.model.characteristics['A']}"


####################################
new *
@when('I add an abstract block "A=(blue, black, brown, yellow)"')
def step_impl(context):
    context.result = context.model.add_block_abstract("A=(blue, black, brown, yellow)")


new *
@then('the abstract block "A" should contain "blue, black, brown, yellow"')
def step_impl(context):
    assert f"Expected: {'blue', 'black', 'brown', 'yellow'}, Actual: {context.model.blocks_abstracts['A']}"


####################################

new *
@given('I have added a characteristic "A=hair color"')
step_impl()
```

همچنین تصویر نهایی تست ها به صورت زیر است


code
main
BDD.feature
Project
code ~/Documents/uni/1402-1403-second-term/Soft
features
steps
BDD.py
BDD.feature
pic
q1.py
q2.py
q2.py
BDD.py
BDD.feature
Created by lwoll at 5/21/24
Feature: Input Range Modeler
As a user of the Input Range Modeler program
I want to add characteristics, abstract blocks, and base choices
So that I can generate test cases in different working modes
Background:
Given the InputRangeModel is initialized
Run
BDD.feature
Test Results
0 ms
Tests passed: 29 of 29 tests – 0 ms
/usr/bin/python3 /opt/pycharm-2023.3.5/plugins/python/helpers/pycharm/behave_runner.py
Testing started at 7:57 PM ...
Process finished with exit code 0