```
# File komega-v02a.py

# Author: G.Doeben-Henisch
# First date: September 4, 2020
# Last change: 11.September 2020

##################################
# Execution Environment of my local machine:
# (venv) gerd@gerd-ub2:~/env/komega/tst$ python3 komega-v01d.py
#

####################################
# GITHUB
#
# We use a github repository at:
# https://github.com/szmt/komega.git
#
# Im working from a unix-shell using the following github-commands:
# https://git-scm.com/docs/git

####################################
# BACKGROUND THEORY
#
# This code is a translation of a theory described in the blog
# https://www.uffmm.org
#
# Last document for the specification of this code:
#
'''
https://www.uffmm.org/2020/09/10/komega-requirements-no-4-version-4-basic-application-scenario/
'''

####################################
# ACTOR STORY
#
# In the specifications an actor story [AS] has been specified. This AS requires # some basic states which are dedicated for certain tasks to do:

'''
ACTOR STORY

S1: START
S2: EDIT P(roblem description)
S3: EDIT S (actual state) AND X (change rules)
S4: SIMULATION (Applying X to S)
S5: EVALUATION (After the simulation)
S6: STOP
'''

# MAIN IDEA
'''
```

```
According to the above mentioned actor story the user will be sitting in front of a system interface
[SI] which works first only as a console.
In the beginning the user is placed in a start state S1 showing all options available.
The user can select one of these options and can from start state S1 reach all other states S2-S6.
'''

############################
# IMPORTS

############################
# SUPPORTING FUNCTION
#

# No funtions yet

# CLASSES
#
'''
For every state there exists one working class to do the job.
The special class 'Publish' in this code exists only because the interaction of the user with the
system will happen with an interactive website which uses HTML and javascript. Here in this
experimental environment a simple unix-console is used.
'''

import kcv2a as kc

############################
# Main Programm
#


###############
# Start main loop
#
# The loop will work as long as the value of the variable 'loop' is different to 'N'

loop='Y'
while loop!='N':

################################
# STATE 1 : START
# Show available options
# Get feedback for selection
# Confirm the selection
# Distribute to different states

        kc.ast.menushow()

# Ask back for selection number
        opt=input('Enter a Number [1-6] for Menu Option \n')

# Evaluate the selection
```

```
        kc.ast.badoption(opt)

# Call to a class instance

        if opt=='2':
                # Where You are
                kc.pub.show(kc.ap)

                #Interaction with Problem Class
                inp=input('Enter your problem as it is now given in plain text\n')
                kc.app.getproblem(inp)

                inp=input('Enter your vision of a better state in the future in plain text\n')
                kc.app.getvision(inp)

                inp=input('Enter the name of the city you are in\n')
                kc.app.getregion(inp)

                inp=input('Time model [From, Until,Cycleunit [Y or M or D or H]]: ')
                kc.app.gettime(inp)

                inp=input('Which kinds of persons are important? Write a list, comma separated
please : ')
                kc.app.getperson(inp)

                kc.app.problemTotal()



        elif opt=='3':
                kc.pub.show(kc.asx)

        elif opt=='4':
                kc.pub.show(kc.asim)

        elif opt=='5':
                kc.pub.show(kc.aev)

        elif opt=='6':
                kc.pub.show(kc.astp)

# Clarify how to continue
        loop=input("STOP = 'N', CONTINUE != 'N'  \n")

'''
TEST WITH CLASS PROBLEM
(venv) gerd@gerd-ub2:~/env/komega/tst$ python3 komega-v02a.py
1  is  START
2  is  EDIT P
3  is  EDIT S and X
4  is  SIMULATION
```

```
5  is  EVALUATION
6  is  STOP
Enter a Number [1-6] for Menu Option
2
!!You have selected the state EDIT P
Role : "Pedit"
Name : "ap"
Enter your problem as it is now given in plain text
Mary needs a book
Feedback Problem Now :
 Mary needs a book
Enter your vision of a better state in the future in plain text
Getting the book from the library
Feedback Problem Future :
 Getting the book from the library
Enter the name of the city you are in
Dieburg
Feedback Problem Region :
 Dieburg
Time model [From, Until,Cycleunit [Y or M or D or H]]: 2020,2020,D
Feedback Problem TimeModel :
 ['2020', '2020', 'D']
Which kinds of persons are important? Write a list, comma separated please : Students, Librarians
Feedback Problem Persons :
 ['Students', ' Librarians']
Feedback Problem All :
 ['Mary needs a book', 'Getting the book from the library', 'Dieburg', '2020,2020,D', 'Students,
Librarians']
STOP = 'N', CONTINUE != 'N'
N
(venv) gerd@gerd-ub2:~/env/komega/tst$

'''
```

```
# File kcv2a.py

# Author: G.Doeben-Henisch
# First date: September 6, 2020
# Last date: September  11, 2020

#################################
# CLASS DEFINITIONS

class Start:
        def __init__(self):
                self.menulist = ['START','EDIT P','EDIT S and
X','SIMULATION','EVALUATION','STOP']

        def menushow(self):
                i=0  # Counter for menu-loop
```

```
            for state in self.menulist:
                i=i+1
                print(i,' is ',state)

    def badoption(self,opt):
        if int(opt)<1 or int(opt)>6:
            print('!!You have selected a bad option')

        if int(opt)>0 and int(opt)<7:
            print('!!You have selected the state',self.menulist[int(opt)-1])

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

class Actor:
    def __init__(self,role,name):
        self.role = role
        self.name = name

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

class Publish():

    def show(self,other):
        print('Role : "%s"'%other.role)
        print('Name : "%s"'%other.name)


#####################################################################
# CLASS PROBLEM
'''
MAIN IDEA

A main window W1 with a menu showing all possible questions to be
answered.

(a) Describe the problem P: What is given and what is the intended future state?
(b) Describe the intended real part of the world (space).
(c) Describe the time model T : which time period, which cycles.
(d) Which kinds of actors are seen as being important for the problem and its  future?
(e) Some other assumptions.
'''


class Problem(Actor):

    def getproblem(self,inp):
        self.problemNow = inp
        print('Feedback Problem Now :\n',self.problemNow)

    def getvision(self,inp):
        self.problemFuture = inp
        print('Feedback Problem Future :\n',self.problemFuture)
```

```
        def getregion(self,inp):
                self.problemRegion = inp
                print('Feedback Problem Region :\n',self.problemRegion)

        def gettime(self,inp):
                self.problemTime = inp
                self.problemTM = self.problemTime.split(',')
                print('Feedback Problem TimeModel :\n',self.problemTM)

        def getperson(self,inp):
                self.problemPerson = inp
                self.problemPRS = self.problemPerson.split(',')
                print('Feedback Problem Persons :\n',self.problemPRS)

        def problemTotal(self):
                self.problemAll =[]
                self.problemAll.append(self.problemNow)
                self.problemAll.append(self.problemFuture)
                self.problemAll.append(self.problemRegion)
                self.problemAll.append(self.problemTime)
                self.problemAll.append(self.problemPerson)
                print('Feedback Problem All :\n',self.problemAll)

##################################
# CLASS INSTANCES

ast=Start()

ap=Actor("Pedit","ap")
app=Problem("PPedit","app")

asx=Actor('SXedit','asx')
asim=Actor('SIM','asim')
aev=Actor('EVAL','aev')
astp=Actor('STOP','astp')

pub=Publish()
```