AI Assignment – 3 :

(Udit Kumar – MT21148)

1.Working program :

```
# forward chaining rule engine
# trying out durable rules engine
#

#pip install durable_rules

from durable.lang import *

with ruleset('M.tech'):
    # will be triggered by 'grades' and 'selected field' facts

    #biology  and music
    @when_all((m.grades >=80) & (m.interested_field == 'Biology') &
(m.passion == 'music') )
    def bio(c):
        c.assert_fact('extra',{'data' :'music'})
        c.assert_fact({ 'course_1': 'Introduction to Quantative
biology', 'course_2': 'Algo in Computation biology' })


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'Biology') & (m.passion =='music'))
    def bio(c):
        c.assert_fact('extra',{'data' :'music'})
        c.assert_fact({ 'course_1': 'Network Biology', 'course_2':
'Data Structures for Genomics' })


     #biology  and sports
    @when_all((m.grades >=80) & (m.interested_field == 'Biology') &
(m.passion == 'sports'))
    def bio(c):
```

```
        c.assert_fact('extra',{'data' :'sports'})
        c.assert_fact({ 'course_1': 'Introduction to Quantative
biology', 'course_2': 'Algo in Computation biology' })


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'Biology') & (m.passion =='sports'))
    def bio(c):
        c.assert_fact('extra',{'data' :'sports'})
        c.assert_fact({ 'course_1': 'Network Biology', 'course_2':
'Data Structures for Genomics' })

     #biology  and sports
    @when_all((m.grades >=80) & (m.interested_field == 'Biology') &
(m.passion == 'body-building'))
    def bio(c):
        c.assert_fact('extra',{'data' :'body-building'})
        c.assert_fact({ 'course_1': 'Introduction to Quantative
biology', 'course_2': 'Algo in Computation biology' })


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'Biology') & (m.passion =='body-building'))
    def bio(c):
        c.assert_fact('extra',{'data' :'body-building'})
        c.assert_fact({ 'course_1': 'Network Biology', 'course_2':
'Data Structures for Genomics' })


    @when_all((m.grades >=80) & (m.interested_field == 'Biology') &
(m.passion =='public-speaking'))
    def bio(a):
        a.assert_fact('extra',{'data' :'public-speaking'})
        a.assert_fact({ 'course_1': 'Introduction to Quantative
biology', 'course_2': 'Algo in Computation biology' })


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'Biology') & (m.passion =='public-speaking'))
    def bio(a):
        a.assert_fact('extra',{'data' :'public-speaking'})
```

```
        a.assert_fact({ 'course_1': 'Network Biology', 'course_2':
'Data Structures for Genomics' })

    #Ai
    @when_all((m.grades >=80) & (m.interested_field ==
'Artificial_Intelligence') & (m.passion =='music'))
    def ai(a):
        a.assert_fact('extra',{'data' :'music'})
        a.assert_fact({ 'course_1': 'Game thoery', 'course_2':
'Robotics' })


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'Artificial_Intelligence') & (m.passion =='music'))
    def ai(a):
        a.assert_fact('extra',{'data' :'music'})
        a.assert_fact({ 'course_1': 'Computer Vision ', 'course_2':
'Natural language processing' })

    #ai - sports
    @when_all((m.grades >=80) & (m.interested_field ==
'Artificial_Intelligence') & (m.passion =='sports'))
    def ai(a):
        a.assert_fact('extra',{'data' :'sports'})
        a.assert_fact({ 'course_1': 'Game thoery', 'course_2':
'Robotics' })

    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'Artificial_Intelligence') & (m.passion =='sports'))
    def ai(a):
        a.assert_fact('extra',{'data' :'sports'})
        a.assert_fact({ 'course_1': 'Computer Vision ', 'course_2':
'Natural language processing' })

    #ai- body-building


    @when_all((m.grades >=80) & (m.interested_field ==
'Artificial_Intelligence') & (m.passion =='body-building'))
    def ai(a):
        a.assert_fact('extra',{'data' :'body-building'})
```

```
        a.assert_fact({ 'course_1': 'Game thoery', 'course_2':
'Robotics' })


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'Artificial_Intelligence') & (m.passion =='body-building'))
    def ai(a):
        a.assert_fact('extra',{'data' :'body-building'})
        a.assert_fact({ 'course_1': 'Computer Vision ', 'course_2':
'Natural language processing' })


    @when_all((m.grades >=80) & (m.interested_field ==
'Artificial_Intelligence') & (m.passion =='public-speaking'))
    def ai(a):
        a.assert_fact('extra',{'data' :'public-speaking'})
        a.assert_fact({ 'course_1': 'Game thoery', 'course_2':
'Robotics' })


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'Artificial_Intelligence') & (m.passion =='public-speaking'))
    def ai(a):
        a.assert_fact('extra',{'data' :'public-speaking'})
        a.assert_fact({ 'course_1': 'Computer Vision ', 'course_2':
'Natural language processing' })


    #circuits and Electronics



    @when_all((m.grades >=80) & (m.interested_field ==
'circuits_electronics') & (m.passion =='music'))
    def cir(a):
        a.assert_fact('extra',{'data' :'music'})
        a.assert_fact({ 'course_1': 'Advanced Embedded logic
design', 'course_2': 'Digital VLSI Design' })


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'circuits_electronics') & (m.passion =='music'))
```

```
    def cir(a):
        a.assert_fact('extra',{'data' :'music'})
        a.assert_fact({ 'course_1': 'Quantum material and devices ',
'course_2': 'Circuit theory and devices' })

    #cir - sports
    @when_all((m.grades >=80) & (m.interested_field ==
'circuits_electronics') & (m.passion =='sports'))
    def cir(a):
        a.assert_fact('extra',{'data' :'sports'})
        a.assert_fact({ 'course_1': 'Advanced Embedded logic
design', 'course_2': 'Digital VLSI Design' })

    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'circuits_electronics') & (m.passion =='sports'))
    def cir(a):
        a.assert_fact('extra',{'data' :'sports'})
        a.assert_fact({ 'course_1': 'Quantum material and devices ',
'course_2': 'Circuit theory and devices' })

    #cir- body-building


    @when_all((m.grades >=80) & (m.interested_field ==
'circuits_electronics') & (m.passion =='body-building'))
    def cir(a):
        a.assert_fact('extra',{'data' :'body-building'})
        a.assert_fact({ 'course_1': 'Advanced Embedded logic
design', 'course_2': 'Digital VLSI Design' })


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'circuits_electronics') & (m.passion =='body-building'))
    def cir(a):
        a.assert_fact('extra',{'data' :'body-building'})
        a.assert_fact({ 'course_1': 'Quantum material and devices ',
'course_2': 'Circuit theory and devices' })

    @when_all((m.grades >=80) & (m.interested_field ==
'circuits_electronics') & (m.passion =='public-speaking'))
    def cir(a):
        a.assert_fact('extra',{'data' :'public-speaking'})
```

```
        a.assert_fact({ 'course_1': 'Advanced Embedded logic
design', 'course_2': 'Digital VLSI Design' })


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'circuits_electronics') & (m.passion =='public-speaking'))
    def cir(a):
        a.assert_fact('extra',{'data' :'public-speaking'})
        a.assert_fact({ 'course_1': 'Quantum material and devices ',
'course_2': 'Circuit theory and devices' })




    #Design and Animation



    @when_all((m.grades >=80) & (m.interested_field ==
'design_animation') & (m.passion =='music'))
    def ani(a):
        a.assert_fact('extra',{'data' :'music'})
        a.assert_fact({ 'course_1': 'Introduction to 3D animation',
'course_2': 'Game Design and Development' })


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'design_animation') & (m.passion =='music'))
    def ani(a):
        a.assert_fact('extra',{'data' :'music'})
        a.assert_fact({ 'course_1': 'Design of Interactive System',
'course_2': 'Visual Design and Communication' })

    #ani - sports
    @when_all((m.grades >=80) & (m.interested_field ==
'design_animation') & (m.passion =='sports'))
    def ani(a):
        a.assert_fact('extra',{'data' :'sports'})
        a.assert_fact({ 'course_1': 'Introduction to 3D animation',
'course_2': 'Game Design and Development' })
```

```
    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'design_animation') & (m.passion =='sports'))
    def ani(a):
        a.assert_fact('extra',{'data' :'sports'})
        a.assert_fact({ 'course_1': 'Design of Interactive System',
'course_2': 'Visual Design and Communication' })

    #ani- body-building


    @when_all((m.grades >=80) & (m.interested_field ==
'design_animation') & (m.passion =='body-building'))
    def ani(a):
        a.assert_fact('extra',{'data' :'body-building'})
        a.assert_fact({ 'course_1': 'Introduction to 3D animation',
'course_2': 'Game Design and Development' })


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'design_animation') & (m.passion =='body-building'))
    def ani(a):
        a.assert_fact('extra',{'data' :'body-building'})
        a.assert_fact({ 'course_1': 'Design of Interactive System',
'course_2': 'Visual Design and Communication' })


    @when_all((m.grades >=80) & (m.interested_field ==
'design_animation') & (m.passion =='public-speaking'))
    def ani(a):
        a.assert_fact('extra',{'data' :'public-speaking'})
        a.assert_fact({ 'course_1': 'Introduction to 3D animation',
'course_2': 'Game Design and Development' })


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'design_animation') & (m.passion =='public-speaking'))
    def ani(a):
        a.assert_fact('extra',{'data' :'public-speaking'})
        a.assert_fact({ 'course_1': 'Design of Interactive System',
'course_2': 'Visual Design and Communication' })

    #SDE
```

```
    @when_all((m.grades >=80) & (m.interested_field == 'algorithm')
& (m.passion =='music'))
    def sde(a):
        a.assert_fact('extra',{'data' :'music'})
        a.assert_fact({ 'course_1': 'Advanced Algorithms',
'course_2': 'Advanced Programming'})


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'algorithm') & (m.passion =='music'))
    def sde(a):
        a.assert_fact('extra',{'data' :'music'})
        a.assert_fact({ 'course_1': 'Mordern algorithms',
'course_2': 'Algorithm Design and Analysis'})

    #sde - sports
    @when_all((m.grades >=80) & (m.interested_field == 'algorithm')
& (m.passion =='sports'))
    def sde(a):
        a.assert_fact('extra',{'data' :'sports'})
        a.assert_fact({ 'course_1': 'Advanced Algorithms',
'course_2': 'Advanced Programming'})

    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'algorithm') & (m.passion =='sports'))
    def sde(a):
        a.assert_fact('extra',{'data' :'sports'})
        a.assert_fact({ 'course_1': 'Mordern algorithms',
'course_2': 'Algorithm Design and Analysis'})

    #sde- body-building


    @when_all((m.grades >=80) & (m.interested_field == 'algorithm')
& (m.passion =='body-building'))
    def sde(a):
        a.assert_fact('extra',{'data' :'body-building'})
        a.assert_fact({ 'course_1': 'Advanced Algorithms',
'course_2': 'Advanced Programming'})
```

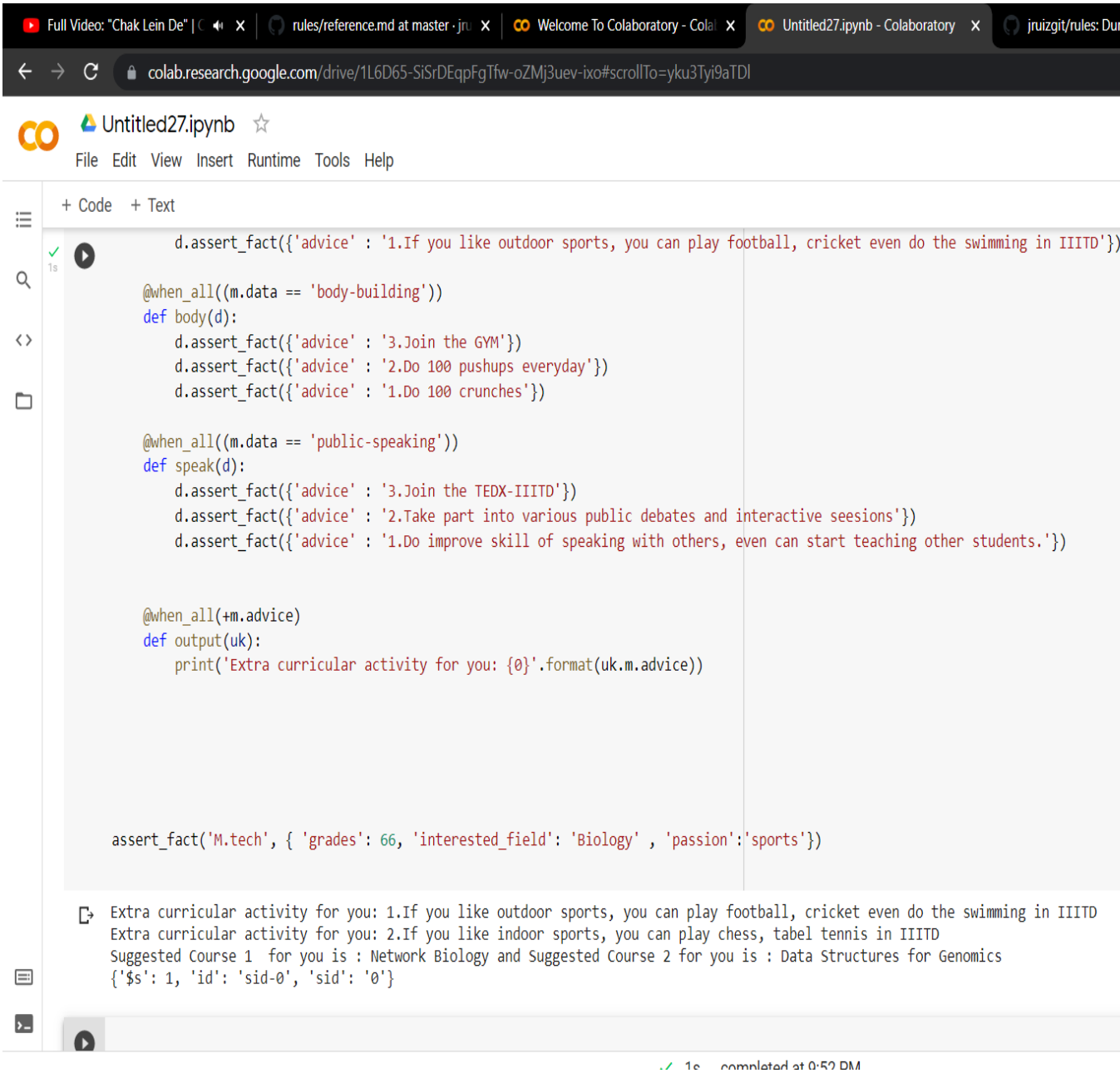
```
    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'algorithm') & (m.passion =='body-building'))
    def sde(a):
        a.assert_fact('extra',{'data' :'body-building'})
        a.assert_fact({ 'course_1': 'Mordern algorithms',
'course_2': 'Algorithm Design and Analysis'})


    @when_all((m.grades >=80) & (m.interested_field == 'algorithm')
& (m.passion =='public-speaking'))
    def sde(a):
        a.assert_fact('extra',{'data' :'public-speaking'})
        a.assert_fact({ 'course_1': 'Advanced Algorithms',
'course_2': 'Advanced Programming'})


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'algorithm') & (m.passion =='public-speaking'))
    def sde(a):
        a.assert_fact('extra',{'data' :'public-speaking'})
        a.assert_fact({ 'course_1': 'Mordern algorithms',
'course_2': 'Algorithm Design and Analysis'})

    #Network and system

    @when_all((m.grades >=80) & (m.interested_field ==
'network_system') & (m.passion =='music'))
    def net(a):
        a.assert_fact('extra',{'data' :'music'})
        a.assert_fact({ 'course_1': 'Computer Network', 'course_2':
'Compiler'})


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'network_system') & (m.passion =='music'))
    def net(a):
        a.assert_fact('extra',{'data' :'music'})
        a.assert_fact({ 'course_1': 'Mobile Computing', 'course_2':
'Operating System'})

    #net - sports
```

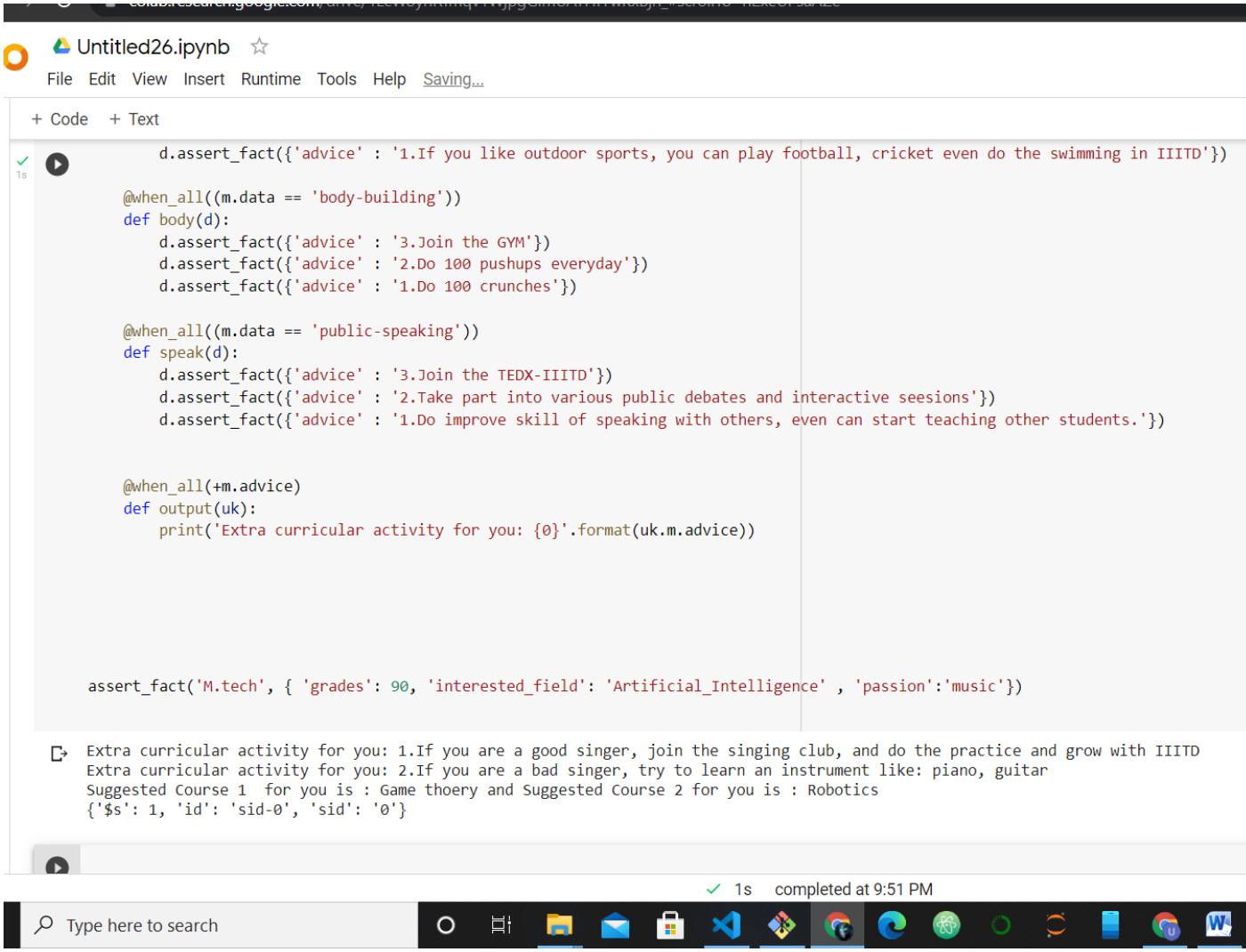
```
    @when_all((m.grades >=80) & (m.interested_field ==
'network_system') & (m.passion =='sports'))
    def net(a):
        a.assert_fact('extra',{'data' :'sports'})
        a.assert_fact({ 'course_1': 'Computer Network', 'course_2':
'Compiler'})

    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'network_system') & (m.passion =='sports'))
    def net(a):
        a.assert_fact('extra',{'data' :'sports'})
        a.assert_fact({ 'course_1': 'Mobile Computing', 'course_2':
'Operating System'})

    #net- body-building


    @when_all((m.grades >=80) & (m.interested_field ==
'network_system') & (m.passion =='body-building'))
    def net(a):
        a.assert_fact('extra',{'data' :'body-building'})
        a.assert_fact({ 'course_1': 'Computer Network', 'course_2':
'Compiler'})


    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'network_system') & (m.passion =='body-building'))
    def net(a):
        a.assert_fact('extra',{'data' :'body-building'})
        a.assert_fact({ 'course_1': 'Mobile Computing', 'course_2':
'Operating System'})

#net- body-building


    @when_all((m.grades >=80) & (m.interested_field ==
'network_system') & (m.passion =='public-speaking'))
    def net(a):
        a.assert_fact('extra',{'data' :'public-speaking'})
        a.assert_fact({ 'course_1': 'Computer Network', 'course_2':
'Compiler'})
```

```
    @when_all((m.grades >=60) & (m.grades <80) & (m.interested_field
== 'network_system') & (m.passion =='public-speaking'))
    def net(a):
        a.assert_fact('extra',{'data' :'public-speaking'})
        a.assert_fact({ 'course_1': 'Mobile Computing', 'course_2':
'Operating System'})

    @when_all(+m.course_1)
    def output(uk):
        print('Suggested Course 1  for you is : {0} and Suggested
Course 2 for you is : {1}'.format(uk.m.course_1,uk.m.course_2))

# for extra curricular activities :
with ruleset('extra'):

    @when_all((m.data == 'music'))
    def music(d):
        d.assert_fact({'advice' : '2.If you are a bad singer, try to
learn an instrument like: piano, guitar'})
        d.assert_fact({'advice' : '1.If you are a good singer, join
the singing club, and do the practice and grow with IIITD'})

    @when_all((m.data == 'sports'))
    def sports(d):
        d.assert_fact({'advice' : '2.If you like indoor sports, you
can play chess, tabel tennis in IIITD'})
        d.assert_fact({'advice' : '1.If you like outdoor sports, you
can play football, cricket even do the swimming in IIITD'})

    @when_all((m.data == 'body-building'))
    def body(d):
        d.assert_fact({'advice' : '3.Join the GYM'})
        d.assert_fact({'advice' : '2.Do 100 pushups everyday'})
        d.assert_fact({'advice' : '1.Do 100 crunches'})

    @when_all((m.data == 'public-speaking'))
    def speak(d):
        d.assert_fact({'advice' : '3.Join the TEDX-IIITD'})
        d.assert_fact({'advice' : '2.Take part into various public
debates and interactive seesions'})
```

```
 d.assert_fact({'advice' : '1.Do improve skill of speaking with
others, even can start teaching other students.'})


    @when_all(+m.advice)
    def output(uk):
        print('Extra curricular activity for you:
{0}'.format(uk.m.advice))




assert_fact('M.tech', { 'grades': 66, 'interested_field': 'Biology'
, 'passion':'sports'})
```

2.Screenshots of Input and Ouput :


Full Video: "Chak Lein De" |
rules/reference.md at master · jru
Welcome To Colaboratory - Colal
Untitled27.ipynb - Colaboratory
jruizgit/rules: Du
colab.research.google.com/drive/1L6D65-SiSrDEqpFgTfw-oZMj3uev-ixo#scrollTo=yku3Tyi9aTDl
Untitled27.ipynb
File Edit View Insert Runtime Tools Help
+ Code + Text
d.assert_fact({'advice' : '1.If you like outdoor sports, you can play football, cricket even do the swimming in IIITD'})
@when_all((m.data == 'body-building'))
def body(d):
d.assert_fact({'advice' : '3.Join the GYM'})
d.assert_fact({'advice' : '2.Do 100 pushups everyday'})
d.assert_fact({'advice' : '1.Do 100 crunches'})
@when_all((m.data == 'public-speaking'))
def speak(d):
d.assert_fact({'advice' : '3.Join the TEDX-IIITD'})
d.assert_fact({'advice' : '2.Take part into various public debates and interactive seesions'})
d.assert_fact({'advice' : '1.Do improve skill of speaking with others, even can start teaching other students.'})
@when_all(+m.advice)
def output(uk):
print('Extra curricular activity for you: {0}'.format(uk.m.advice))
assert_fact('M.tech', { 'grades': 66, 'interested_field': 'Biology' , 'passion':'sports'})
Extra curricular activity for you: 1.If you like outdoor sports, you can play football, cricket even do the swimming in IIITD
Extra curricular activity for you: 2.If you like indoor sports, you can play chess, tabel tennis in IIITD
Suggested Course 1 for you is : Network Biology and Suggested Course 2 for you is : Data Structures for Genomics
{'$s': 1, 'id': 'sid-0', 'sid': '0'}


Untitled26.ipynb
File Edit View Insert Runtime Tools Help Saving...
+ Code + Text
d.assert_fact({'advice' : '1.If you like outdoor sports, you can play football, cricket even do the swimming in IIITD'})
@when_all((m.data == 'body-building'))
def body(d):
d.assert_fact({'advice' : '3.Join the GYM'})
d.assert_fact({'advice' : '2.Do 100 pushups everyday'})
d.assert_fact({'advice' : '1.Do 100 crunches'})
@when_all((m.data == 'public-speaking'))
def speak(d):
d.assert_fact({'advice' : '3.Join the TEDX-IIITD'})
d.assert_fact({'advice' : '2.Take part into various public debates and interactive seesions'})
d.assert_fact({'advice' : '1.Do improve skill of speaking with others, even can start teaching other students.'})
@when_all(+m.advice)
def output(uk):
print('Extra curricular activity for you: {0}'.format(uk.m.advice))
assert_fact('M.tech', { 'grades': 90, 'interested_field': 'Artificial_Intelligence' , 'passion':'music'})
Extra curricular activity for you: 1.If you are a good singer, join the singing club, and do the practice and grow with IIITD
Extra curricular activity for you: 2.If you are a bad singer, try to learn an instrument like: piano, guitar
Suggested Course 1 for you is : Game thoery and Suggested Course 2 for you is : Robotics
{'$s': 1, 'id': 'sid-0', 'sid': '0'}
1s completed at 9:51 PM
Type here to search

Untitled25.ipynb

ile Edit View Insert Runtime Tools Help

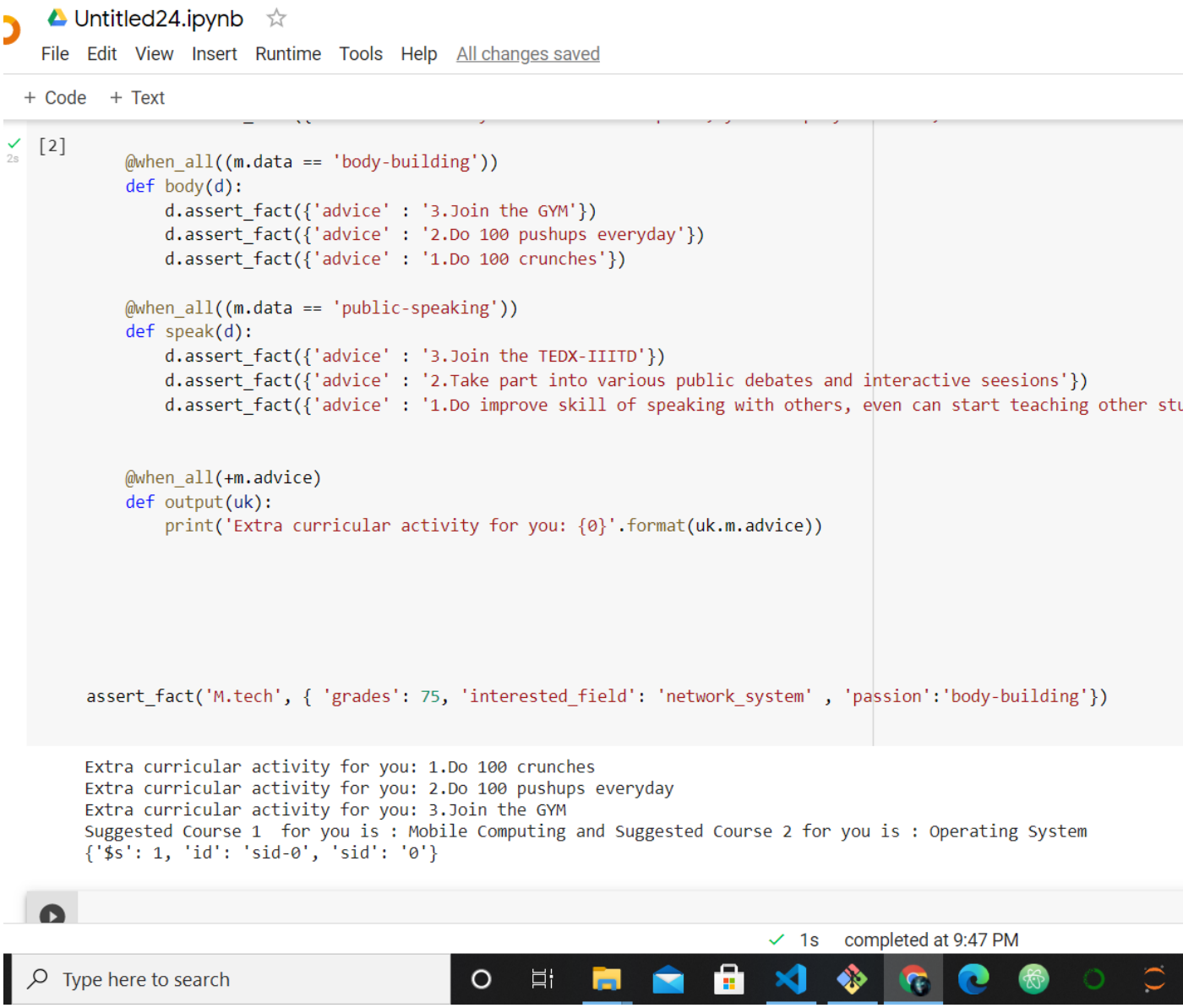
ode + Text

```
        d.assert_fact({'advice' : '2.If you like indoor sports, you can play chess, tabel tennis in IIITD'})
        d.assert_fact({'advice' : '1.If you like outdoor sports, you can play football, cricket even do the swimming in IIITD'})

@when_all((m.data == 'body-building'))
def body(d):
    d.assert_fact({'advice' : '3.Join the GYM'})
    d.assert_fact({'advice' : '2.Do 100 pushups everyday'})
    d.assert_fact({'advice' : '1.Do 100 crunches'})

@when_all((m.data == 'public-speaking'))
def speak(d):
    d.assert_fact({'advice' : '3.Join the TEDX-IIITD'})
    d.assert_fact({'advice' : '2.Take part into various public debates and interactive seesions'})
    d.assert_fact({'advice' : '1.Do improve skill of speaking with others, even can start teaching other students.'})


@when_all(+m.advice)
def output(uk):
    print('Extra curricular activity for you: {0}'.format(uk.m.advice))




assert_fact('M.tech', { 'grades': 90, 'interested_field': 'algorithm' , 'passion':'public-speaking'})
```

```
Extra curricular activity for you: 1.Do improve skill of speaking with others, even can start teaching other students.
Extra curricular activity for you: 2.Take part into various public debates and interactive seesions
Extra curricular activity for you: 3.Join the TEDX-IIITD
Suggested Course 1  for you is : Advanced Algorithms and Suggested Course 2 for you is : Advanced Programming
{'$s': 1, 'id': 'sid-0', 'sid': '0'}
```

✓ 1s completed at 9:49 PM

Type here to search



3.Expanation of the Program :

Explanation of the program :

First, we will start the program by asserting one single fact with M.tech.

We have three attributes from the user: grades, interested field, and passion.

It matches with the Ruleset called M.tech. Then, I have generally taken four passions into account: music, sports, body-building, public speaking.

For every field, for example, Biology, we have divided the courses in that field based on the grades.

Complex top two subjects are shown as the preference if the grades are more than equal to 80. if the grades are in the range of 60 to 80, the program will recommend the other two courses in that particular field.

The program suggests two particular courses, and based on the passion, it will indicate some advice in that extracurricular activities.

4.Forward Chaining used :

Forward chaining has been used in the program when asserts the passion of the user.

Each time the fact asserts the user's passion, we have made a new Ruleset called "edit", which matches the user's passion with the various possibilities of the passion and asserts the facts related to a selected passion. This edit ruleset maintains the suggestion for the user for a particular extracurricular activity.

So this is how we have used forward chaining in our program.

5.Ingenuity :

We have made various interested field for the user :

Artificial Intelligence, Biology, algorithm, Network_system, design_animation.

The way that we have divided our logic on the basis of the grades is the secret of our Ingenuity,

For grades >= 80 program suggest different courses and for grades in the range of (60 to 80) we have suggested different courses.