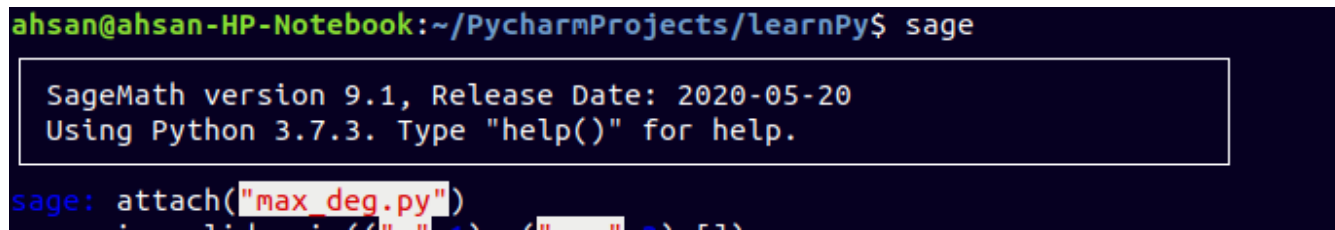


## How to use the max\_deg.py program

**Prerequisite:** The program utilizes functions from Sage math. So, Sage math needs to be installed in the machine.

**Running the program:** The program finds the maximum possible degree of the invariants in the form non-linear equality from the concrete traces.

First go to the directory with the program, run sage math, and attach the program.



```
ahsan@ahsan-HP-Notebook:~/PycharmProjects/learnPy$ sage
SageMath version 9.1, Release Date: 2020-05-20
Using Python 3.7.3. Type "help()" for help.
sage: attach("max_deg.py")
```

To find the maximum degree, you need to call the function *get\_max\_deg*. The function signature is as follows:

```
def get_max_deg(traces_as_dict, input):
```

Here, the *traces\_as\_dict* takes traces as a dictionary. The keys in the dictionary are variable names as strings and value are the traces as lists. Let, a program has variables x,y,z and their traces are as follows:

```
x = [1,2,3,4,5]
y = [2,4,6,8,10]
z = [10,10,10,10,10]
```

The *traces\_as\_dict* would then be:

```
{‘x’: [1,2,3,4,5],
‘y’: [2,4,6,8,10],
‘z’: [10,10,10,10,10]}
```

The *input* parameter expects a list with the names of the input variables (the variables that have constant traces) as string. If z is the input variable in the previous example, argument for *input* parameter would then be: [‘z’]

If multiple traces are generated running the program multiple times, all the traces should still go into in one dictionary. Below is an example of with the traces from Bresenham.java .

When the program is run with input X = 15 and Y = 33, and the traces are passed to the *get\_max\_deg* function, we get following output:

```

sage: get_max_deg({
.....: 'X': [15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15],
.....: 'Y': [33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33],
.....: 'x': [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16],
.....: 'y': [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16],
.....: 'v': [51, 87, 123, 159, 195, 231, 267, 303, 339, 375, 411, 447, 483, 519, 555, 591, 627]
.....: }, ['X', 'Y'])
1

```

When the program is run with input  $X = 11$  and  $Y = 27$ , and the traces are passed to the `get_max_deg` function, we get following output:

```

sage: get_max_deg({
.....: 'X': [11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11],
.....: 'Y': [27, 27, 27, 27, 27, 27, 27, 27, 27, 27, 27, 27, 27, 27],
.....: 'x': [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12],
.....: 'y': [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12],
.....: 'v': [43, 75, 107, 139, 171, 203, 235, 267, 299, 331, 363, 395, 427]
.....: }, ['X', 'Y'])
1

```

Both of the outputs are wrong, because the expected maximum degree is 2, not 1. Now if we add both traces in one dictionary and pass as argument to `get_max_deg`, we get:

```

sage: get_max_deg({
.....: 'X': [15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11],
.....: 'Y': [33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 33, 27, 27, 27, 27, 27, 27, 27, 27, 27, 27, 27],
.....: 'x': [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12],
.....: 'y': [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12],
.....: 'v': [51, 87, 123, 159, 195, 231, 267, 303, 339, 375, 411, 447, 483, 519, 555, 591, 627, 43, 75, 107, 139, 171, 203, 235, 267, 299, 331, 363, 395, 427]
.....: }, ['X', 'Y'])
2

```

Now we got the correct maximum degree 2 as output. Because of the way the algorithm is designed, multiple traces from running the programs with different input values is helpful in finding the relationship between the constant input values and the coefficients of the polynomials. So, using multiple traces is recommended.

At times, the `max_deg` program can output unexpected degree (rare case), this is because there may be a coincidental relationship between the variables for a particular input which is not the universal case for all inputs. That's why running `max_deg` the program multiple times and taking the most occurred output is a surer way of getting the right degree (this isn't usually needed, but helpful).

For, `Dijkstra.java`, `Fermat1.java`, and `Fermat2.java`, the program may give erroneous output. If the program can't find the maximum degree, it will output `None`.