```
"""
Polymorphism: Poly(many) + morph (forms)
Overloading: means a method/operator/constructor behaves differntly
If we conside a method
so we may  have a method with same name and different attributes
"""
class Sample:
  def m1(self): #behaviour_1
    print('No args')
  def m1(self,a,b): ##behaviour_2
    print( 'with 2 variables')
s = Sample()
s.m1(1,2) #its possible bcz program control will only takes last recent declaration from a cl
# and last declaration is with 2 args
# if we use with no args
s.m1() # it will throw an exception related to a and b positional args
# conclusion: we cant perform method overloading/method level polymorphism is nt possible
```

```
with 2 variables
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-3-be63e9d35ffb> in <module>()
     16 # and last declaration is with 2 args
     17 # if we use with no args
---> 18 s.m1() # it will throw an exception related to a and b positional args

TypeError: m1() missing 2 required positional arguments: 'a' and 'b'
```

SEARCH STACK OVERFLOW

```
"""
Now check constrcutor level polymorphism or Constructor overloading
"""
class Sample:
  def __init__(self):
    print(0)
  def __init__(self,a,b,c):
    print(3)
  def __init__(self,x):
    print(1)
Sample(10)
```

```
1
<__main__.Sample at 0x7f2e1af260d0>
```

```
# if we take init with 3 variables
class Sample:
  def __init__(self):
    print(0)
  def __init__(self,a,b,c):
    print(3)
  def __init__(self,x):
    print(1)
Sample(1,5,0) # it wont call 2nd init bcz last recent init is with only 1 arg and its x
#hence constructor overloading is nt possible in python
# Q. What is overloading
# Q. what is constructor/method overloading
# q. is it posible to perform method overloading?
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-8-6274790c30f5> in <module>()
      7   def __init__(self,x):
      8     print(1)
----> 9 Sample(1,5,0) # it wont call 2nd init bcz last recent init is with only 1 arg ar

TypeError: __init__() takes 2 positional arguments but 4 were given
```

SEARCH STACK OVERFLOW

```
# Operator overloading: we can implement different behaviour of an operators
# ex. + - * % /
```

```
print(dir(10))
```

```
shift__', '__lt__', '__mod__', '__mul__', '__ne__', '__neg__', '__new__', '__or__', '__p
```

```
# method with __ double underscore at the prefix and suffix side are used to indicate them
# as a special methods
# We can call them as Dunder method
# also they are known as Magic methods
```

```
12 + 12
```

```
24
```

```
class A:
  def __init__(self,a):
    print(a)

a1 = A(10)
```

```
10
```

```
class Book:
  def __init__(self,pages):
    self.pages = pages
  def __mul__(self,other):
    return self.pages * other.pages
b1 = Book(100)
b2 = Book(200)
print(b1*b2) # add 2 objects
#print('100'+'200')
```

```
20000
```

```
def sample():
  result =  100 +80
  return result
print(sample())
```

```
180
```

✓ 0s completed at 7:44 PM