

- Pa 2 : Friday
- lab 1 : ext to tonight

continue .. Generators

Recursive fn : GCD

Ex find gcd of 80, 45

Euclidean Algorithm:

a	b	r
80	=	$1 \cdot 45 + 35$
45	=	$1 \cdot 35 + 10$
35	=	$3 \cdot 10 + \boxed{5} \leftarrow \text{gcd}$
10	=	$2 \cdot \boxed{5} + 0 \leftarrow \text{stop}$
	↑	
	return b	

```
def gcd(a, b):
```

```
    r = a % b
```

```
    while r > 0:
```

```
        a = b
```

```
        b = r
```

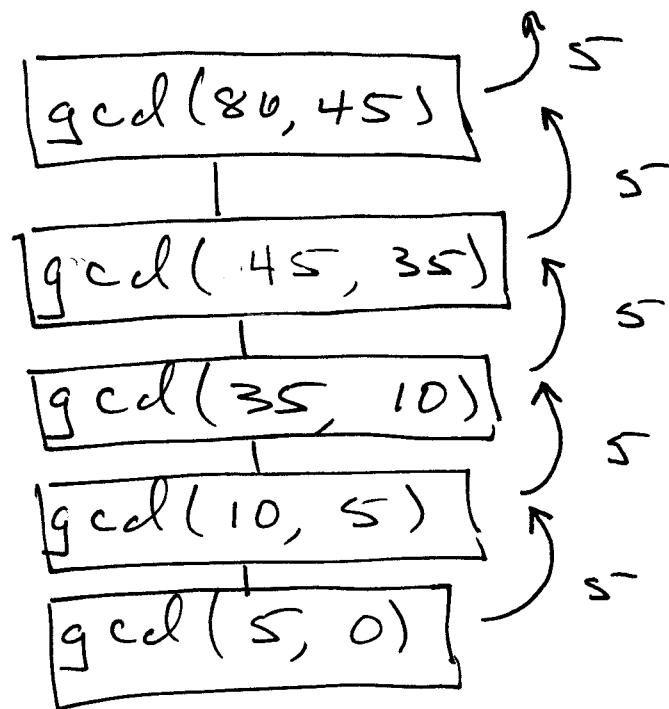
```
        r = a % b
```

```
    # end
```

```
    return b
```

```
# end
```

Recursive version: Box trace



Pa2 stuff:

n - Queens Problem

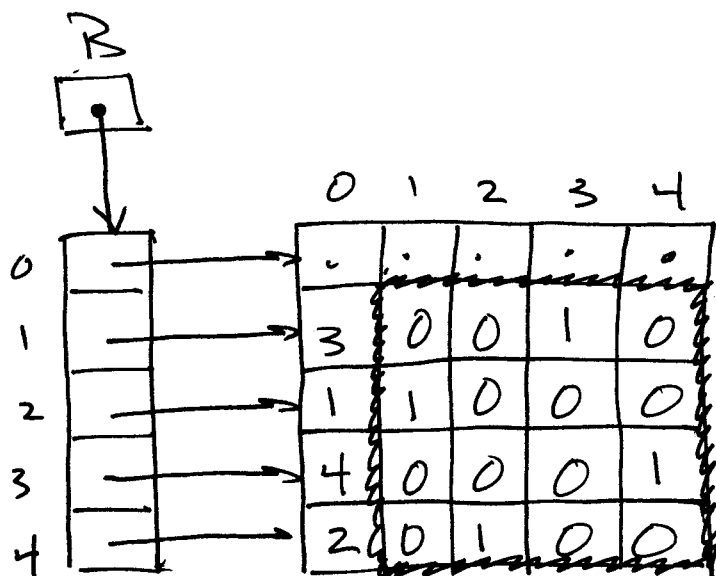
n = 4

	1	2	3	4
1			Q	
2	Q			
3				Q
4		Q		

a solution
to 4-Queens

How to represent chessboard. ?

list of lists :



The diagram shows a queen piece on a board. Arrows point from the queen's position to a list of lists representing the board state. The list of lists is a 5x5 grid where the first column contains the row indices (0-4) and the other columns contain 0 or 1, indicating the presence of a queen.

	0	1	2	3	4
0	0	0	0	0	0
1	3	0	0	1	0
2	1	1	0	0	0
3	4	0	0	0	1
4	2	0	1	0	0

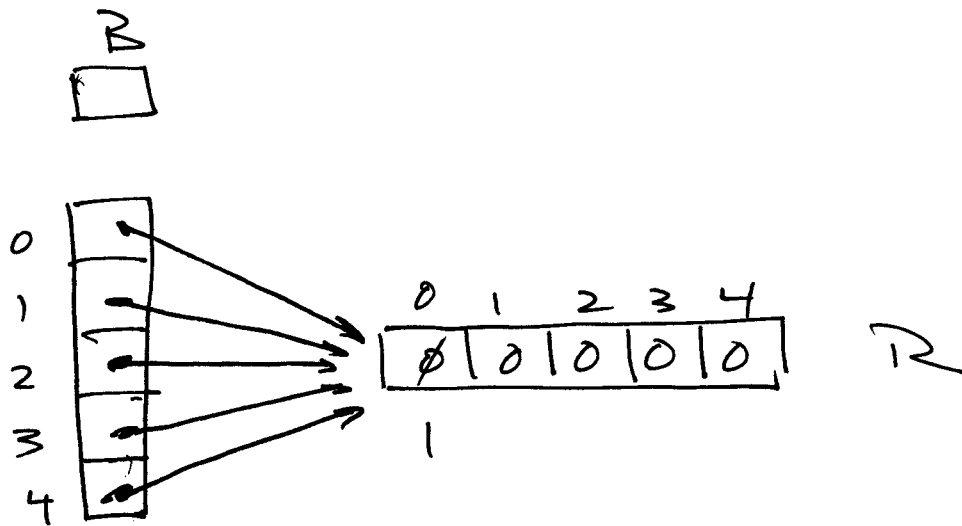
How to create R ?

$$R = 5 * [5 * [0]]$$

$$= 5 * [10, 0, 0, 0, 0] \leftarrow \text{write a loop instead}$$

Try $R[0][0] = 1$

we got this



Do something like

for ----- :

$R.append(R[:])$

end.

← copy of R

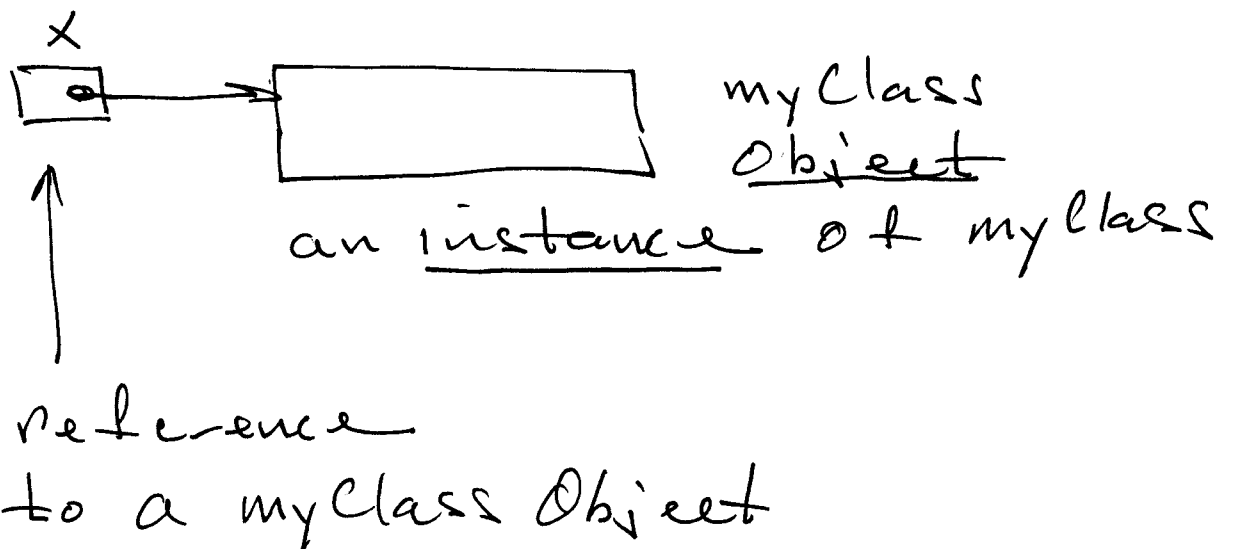
Classes

classes are used to define new data types.

simplest class

```
class myClass:
    Pass
# end
```

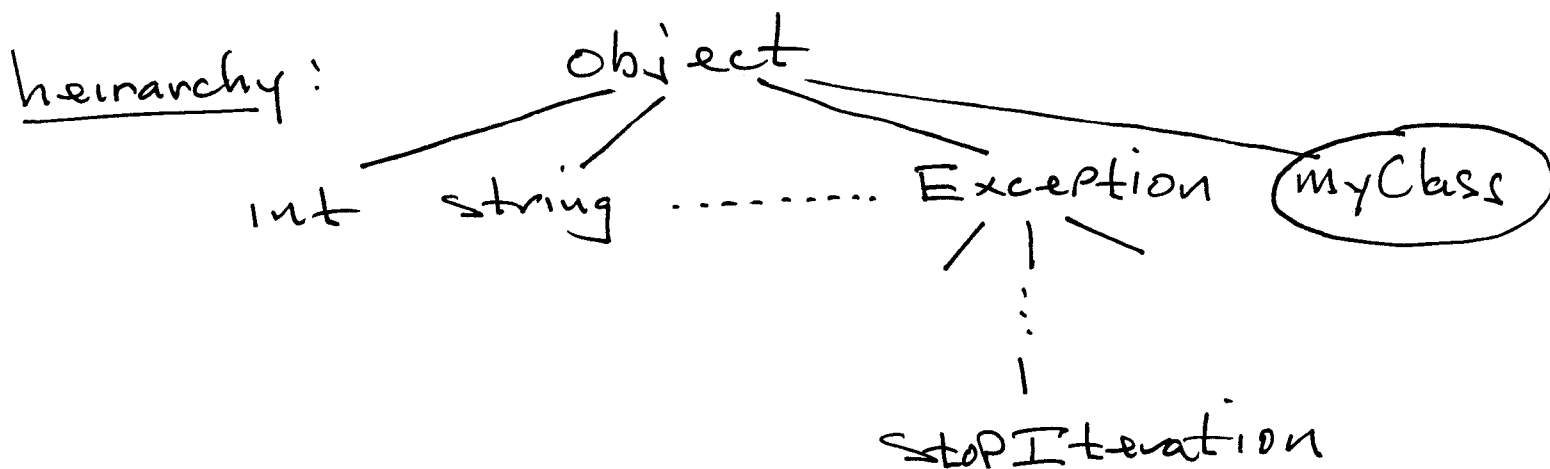
x = myClass() # myClass constructor



The members of a class are called its attributes. they can be

- variables (Properties) :
reference to other objects
- methods (functions) :
operations on variables.

`dir(x)` or `dir(myClass)` we see a bunch of attributes inherited from object superclass :



locally : object
 |
 myClass

To create further subclasses do:

```
class mySubclass(myClass):
    Pass
#end
```

now :

```

      object
      |
myClass ← [Super class
           [Base class
           |
mySubclass ← [Derived class
              [sub-class
              [child class
  
```

Inheritance : every attribute of
 Base class is also an attribute
 of Derived class.

Ex.

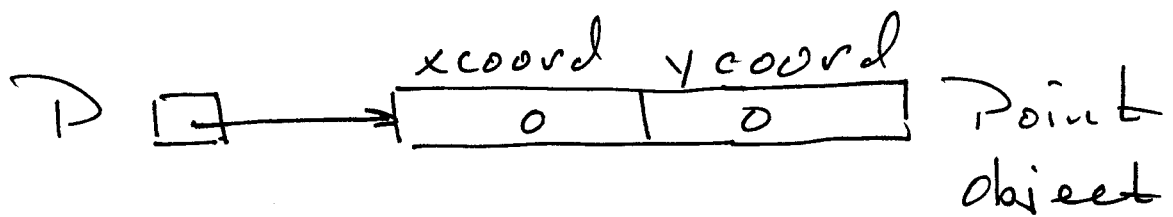
```

class Point(object):
    xcoord = 0
    ycoord = 0
#end

```

optional
↓

P = Point() # Point constructor



Print(P.xcoord) # 0

.. (P.ycoord) # 0

P.xcoord = 1

P.ycoord = 7

Print(P.xcoord) # 1

.. (P.ycoord) # 7

Apparently an instance of a class is like a dictionary.