Python training - lab 9

Functions - nonlocal variables

```
def func1():
    def func2():
        nonlocal x
        x = 6
        print(x)

    x = 7
    func2()
    print(x)


x = 5
func1()
print(x)
```

Functions returned from functions

```
def func(x):
    def func2(y):
        return x * y

    return func2

# x is a function
x = func(5)
print(x(7))
```

Recursive functions

```
def factorial(n):  # non-recursive
    t = 1
    for i in range(1, n + 1):
        t *= i

    return t

def factorial(n):  # recursive
    if n == 1:
        return 1
    return n * factorial(n - 1)

print(factorial(5))
```

Lambda functions

```
# short-lived, one expression, anonymous functions
lambda x, y: x * y
lambda a: a ** 2

r = list(
   map(lambda x, y: x * y,
         ['a', 'b', 'c'],
         [2, 3, 5]
   )
)
print(r)


y = filter(lambda x: x >= 3, (1, 2, 3, 4))
z = filter(None, (-1, 0, 1, '', 2, 3, 4))
```

Comprehensions

```
l = [x ** 2 for x in range(0, 10, 2)]

s = {x * 2 for x in list('abcde')}

l = [x for x in range(50, 30, -3) if x % 2 == 1]

d = {x : 10 / x for x in range(50, 30, -3) if x % 2
== 1}

d = {x: 'par' if x % 2 == 0 else 'impar' for x in
range(10)}

d = {x: 'par' if x % 2 == 0 else 'impar' for x in
range(30) if x % 3}
```

Comprehensions

```
d = {x: 'par' if x % 2 == 0 else 'impar'
     for x in range(30) if x % 3}

# equivalent

d3 = {}
for x in range(30):
   if not x % 3:
      continue
   if x % 2 == 0:
      d3[x] = 'par'
   else:
      d3[x] = 'impar'
```