Anonymous Function

A lambda function in Python is a small, anonymous (unnamed) function defined using the lambda keyword. It can take any number of arguments, but can only contain a single expression.

- Lambda lets you create functions without a name.
- No Need to Name
- Useful when the function is simple and used only once or temporarily.
- List item
- Lambda syntax is much shorter than defining a full def function.

Syntax:

```
# lambda arguments: expression
```

lambda is a keyword that introduces the function.

It returns the result of the expression automatically.

You do not use return in lambda.

Lambda functions are often used for short, throwaway functions.

```
#Add Numbers
add = lambda x, y: x + y
print(add(5, 3))
```

```
8
```

```
#Subtract Numbers
sub = lambda x, y: x - y
print(sub(5, 3))
```

```
2
```

```
#Multiply numbers
mul = lambda x, y: x * y
print(mul(5, 3))
```

```
15
```

```
#Divide numbers
div = lambda x, y: x / y
print(div(5, 3))
```

```
1.6666666666666667
```

```
#floor Division
fdiv = lambda x, y: x // y
print(fdiv(5, 3))
```

```
1
```

```
#power of a number
pw = lambda x, y: x ** y
print(pw(5, 3))
```

```
125
```

```
#mod numbers
md = lambda x, y: x % y
print(md(5, 3))
```

```
2
```

```
#check number is even
is_even = lambda x: x % 2 == 0
print(is_even(4))
```

```
True
```

```
#check number is divisible of 5
is_div5 = lambda x:x%5==0
print(is_div5(15))
```

```
True
```

```
#maximum of two numbers
m2 = lambda x,y:x if x>=y else y
print(m2(5,3))
```

```
5
```

```
#minimum of two numbers
mn2 = lambda x,y: x if x<=y else y
print(mn2(5,6))
```

```
5
```

```
#maximum of three numbers
mx3 = lambda x,y,z: x if x>=y and x>=z else y if y>=x and y>=z else z
print(mx3(2,5,3))
```

```
5
```

```
#minimum of three numbers
mx3 = lambda x,y,z: x if x<=y and x<=z else y if y<=x and y<=z else z
print(mx3(2,5,3))
```

```
2
```

```
#
s = 'python is a programming language'
lw = lambda s:s.split()
print(lw(s))
```

```
['python', 'is', 'a', 'programming', 'language']
```

```
#print first character of each word of sentence
s = 'python is a programming language'
first_char = lambda s:[i[0] for i in s.split()]
print(first_char(s))
```

```
['p', 'i', 'a', 'p', 'l']
```

```
#print last character of each word of sentence
s = 'python is a programming language'
last_char = lambda s:[i[-1] for i in s.split()]
print(last_char(s))
```

```
['n', 's', 'a', 'g', 'e']
```

```
#filter odd numbers
od = lambda l:[i for i in l if i%2!=0]
print(od([3,2,1,7,6,4,8]))
```

```
[3, 1, 7]
```

```
#filter even numbers from range
odr = lambda n:[i for i in range(n+1) if i%2==0]
```

```
print(odr(10))
```

```
[0, 2, 4, 6, 8, 10]
```

```
#find the length of a string
sl = lambda s:len(s)
print(sl('python'))
```

```
6
```

```
#lambda function in function
def make_incrementor(n):
    return lambda x: x + n

inc_by_5 = make_incrementor(5)
print(inc_by_5(10))
```

```
15
```

```
#check number is positive, negative or zero
cn = lambda n:'positive' if n>0 else 'negative' if n<0 else 'zero'
print(cn(10))
```

```
positive
```

```
#get last digit of a number
ln = lambda n:n%10
print(ln(123))
```

```
3
```

```
#total digits in a number
tdn = lambda n:len(str(n))
print(tdn(123))
```

```
3
```

```
#check string is pallidrome
cp = lambda s:'pallidrome' if s==s[::-1] else 'not pallidrome'
print(cp('bob'))
```

```
pallidrome
```

```
#capitalize a string
cw = lambda s:s.capitalize()
print(cw('python is a programming language'))
```

```
Python is a programming language
```

```
#title to a string
ts = lambda s:s.title()
print(ts('python is a programming language'))
```

```
Python Is A Programming Language
```

```
#check if string is numeric
sn = lambda s:'numeric' if s.isnumeric() else 'not numeric'
print(sn('123'))
```

```
numeric
```

```
#check string is numeric
sn1 = lambda s:'numeric' if s.isdigit() else 'not numeric'
print(sn1('123'))
```

```
numeric
```

```
#list all vowels
lw1 = lambda s:[i for i in s if i in 'aeiouAEIOU']
print(lw1('python is a prOgramming lAnguage'))
```

```
['o', 'i', 'a', 'O', 'a', 'i', 'A', 'u', 'a', 'e']
```

```
#count all vowels
cwa = lambda s:len([i for i in s if i in 'aeiouAEIOU'])
print(cwa('python is a prOgramming lAnguage'))
```

```
10
```

```
#count unique vowels
cuw = lambda s:len({i for i in s if i in 'aeiouAEIOU'})
print(cuw('python is a prOgramming lAnguage'))
```

```
7
```

```
#extract username from email
user = lambda s:s.split('@')[0]
print(user('python@gmail.com'))
```

```
python
```

```
#increament every number by 5
in2 = lambda l:[i+5 for i in l]
print(in2([1,2,3,4,5]))
```

```
[6, 7, 8, 9, 10]
```

```
#remove duplicates from list
rdl = lambda l:list(set(l))
print(rdl([1,2,4,3,4,5,1,2,3,4,5]))
```

```
[1, 2, 3, 4, 5]
```

```
#filter words longer than 5 letters from a string
l5 = lambda s : [i for i in s.split() if len(i)>5]
print(l5('python is a programming language'))
```

```
['python', 'programming', 'language']
```

```
#filter pallidromes from a string
fps = lambda s: [i for i in s.split() if i==i[::-1]]
print(fps('python is wow and bob is amazing person'))
```

```
['wow', 'bob']
```

```
#filter pallidromes from a list
fps1 = lambda l: [i for i in l if i==i[::-1]]
print(fps1(['python','bob','grow','wow','madam','not']))
```

```
['bob', 'wow', 'madam']
```

```
#combine first and last word
flw = lambda s:s.split()[0]+s.split()[-1]
print(flw('python is a programming language'))
```

```
pythonlanguage
```

```
#combine first and last character
flw = lambda s:s.split()[0][0]+s.split()[-1][-1]
print(flw('python is a programming language'))
```

```
pe
```

```
#merge two dictionary
md = lambda d1,d2:{**d1,**d2}
print(md({'a':1,'b':2},{'c':3,'d':4}))
```

```
{'a': 1, 'b': 2, 'c': 3, 'd': 4}
```

```
#factorial of a number
f = lambda n: 1 if n==0 else n*f(n-1)
```

```
print(f(5))
```

```
120
```

```
#fibonacci series
fn = lambda n: n if n==0 or n==1 else fn(n-1)+fn(n-2)
for i in range(5):
  print(fn(i))
```

```
0
1
1
2
3
```

```
#Calculate absolute value
av = lambda x: x if x >= 0 else -x
print(av(-7))
```

```
7
```

```
#check leap year
ly = lambda y: 'leap year' if y%4==0 and y%100!=0 or y%400==0 else 'not leap year'
print(ly(2000))
```

```
leap year
```

```
#convert minutes into hours
mh = lambda m:m/60
print(mh(60))
```

```
1.0
```

```
#replace vowel with *
rws = lambda s:''.join(['*' if i in 'aeiouAEIOU' else i for i in s])
print(rws('python is a programming language'))
```

```
pyth*n *s * pr*gr*mm*ng l*ng**g*
```

```
print((lambda x, y: x + y)(5, 3))
```

```
8
```

Functions

filter() - filter values from list

map() - map each element

```
nums = [1, 2, 3, 4, 5, 6]
even_nums = list(filter(lambda x: x % 2 == 0, nums))
print(even_nums)
```

```
[2, 4, 6]
```

```
special_nums = [2, 5, 4, 3, 6, 7, 1, 8]
non_special_nums_1 = list(map(lambda x: x*5, special_nums))
print(non_special_nums_1)
non_special_nums_2 = list(map(lambda x: pow(x, 2), special_nums))
print(non_special_nums_2)
```

```
[10, 25, 20, 15, 30, 35, 5, 40]
[4, 25, 16, 9, 36, 49, 1, 64]
```

```
#lambda function in functions
def func(n):
 return lambda x: x*n
n1 = func(2)
n2 = func(10)
print(f'The multiplication is equal to {n1(5)}.')
print(f'The multiplication is equal to {n2(8)}.')
```

```
The multiplication is equal to 10.
The multiplication is equal to 80.
```

Start coding or generate with AI.