ანონიმური ფუნქციები

ლამბდა ფუნქცია ინახება ცვლადში რადგან მას სახელი არ ექნიჭება

ჩვეულებრივი ფუნქცია

Def add (x, y):

Return x+y

Lambda function – return არ გვჭირდება რადგან დეფოლტად იყენებს

Add\_lambda= lambda x,y: x+y

ფუნქციონალს ვიძახებთ ცვლადის სახელის გამოძახებით და ფრჩხილების დამატებით

Print (add\_lambda(x=10,y=15))

Add\_lambda= (lambda x=10,y=15: x+y) () პირდაპირ იძახებს ფუნქციას

Print (add\_lambda)

Print (add\_lambda()) და ზემოთ წავშლი

შეიქმნა ლისტი, რომელშიც აღწერიალია ლამბდა ფუნქცია , რომელიც 1-11 ის ჩათვლით გაამრავლებს ყველა რიცხვს 10 ზე გამოძახებული ფორ ციკლის მეშვეობით

Lst=[lambda arg=x: arg\*10 for x in range (1,11)]

For func in lst

Print func())

Lst=[ (lambda arg=x: arg\*10)() for x in range (1,11)] - აქაც პირდაპირ გამოიძახებს

Lst2=[5,10,15,16,8,19]

Even\_list=[(lambda x=number: x if x%2==0 else “odd”)() for number in lst2]

აქ დაბეჭდავს მხოლოდ ლუწ რიცხვებს if პირობის გათვალისწინებით და ფორის მეშვეობით დაუვლის ცვლად ლისტში არსებულ სიას

Partial function

From functools import partial

ატრიბუტად გადაეცემა ფუნქცია და არგუმენტი

მაგალითად:

Def mult (x,y):

Return x:y

Multiplication=partial (mult, x=5)

გამოძახების დროს უნდა გადავცე ცვლადი როგორც ფუნქცია და ის ატრიბუტი რაზეც მინდა მოქმედება

ამ მაგალითში იქსი არის 5

Print (multiplication(y=10)) აქ უნდა მივუთითო ქივორდი და მნიშვნელობა მეორე ატრიბუტად მივიღებთ 5\*10= 50ს

Print (multiplication(y=20)) აქ 100

თუ მინდა იქსის შეცვლაც მაშინ

Print (multiplication(x=10, y=20))მივიღებთ 200

From functools import reduce

List=[10, 15,20,25,30]

Sum\_of\_list=reduce(lambda a,b: a+b, list)

აქ დააჯამებს ლისტის რიცხვებს, დაივლის და ჯერ დააჯამებს პირველ ორ რიცხვს 10+15, და დაიმახსოვრებს ა-ში , შემდეგ ას დაუმატებს მომდევნო რიცხვს ბს 20, და ახალ ჯამს დაიმახსოვრებს ა ში და ისე მიუმატებს მომდევნოს და ასე გავა ბოლომდე

Factorial

Fact=reduce (lambda a, b: a\*b, range (1,6) )

Print (fact) 120, ერთმანეთზე გადაამრვალებს

Map უნდა გადაეცეს ლისტი , და ფუნქცია

Number\_list=[10,2,15,3]

Def mult(x)

Return x\*5

Result=list(map(mult, number\_list))

Print(result)

Powers=list (map(lambda x: [x\*\*2, x\*\*3], number\_list)

Print (powers)

დააბრუნებს პატარ პატარა ლისტებს რომელშიც იქნება ლისტის ციფრები კვადრატში და კუბში ახარისხებული სათითაოდ,

ფილტერ ფუნქცია იღებს ფუნქციას და იტერეტირებადს

იღებს თრუს ან ფოლს

Number\_list=[10,2,15,3]

Odd\_numbers= filter (lambda x: x%2==1, number\_list)

თუ იქნება თრუ შეინახავს 10 იანს

Print(list(odd\_numbers))

Zip

Keywords = [ “first\_name”, “last\_name”, “age” ]

Values=[“tom”, “tailor”, 25]

Data=zip(keywords, Values)

Print (list(Data))

აბრუნებს ტუპლის ტიპის ინფორმაციას გაერთიანებულს,

დაბეჭდავს [ (“first\_name”, “tom” ), (“last\_name”, “tailor”), (“age”, 25)]