

توابع در پایتون :

به هر بلوک از کد که به بار های متعدد قابل اجرا شدن هستن توابع گفته میشوند.
توابع میتوانند ورودی و خروجی داشته باشند

سینتکس توابع :

Def name (args):

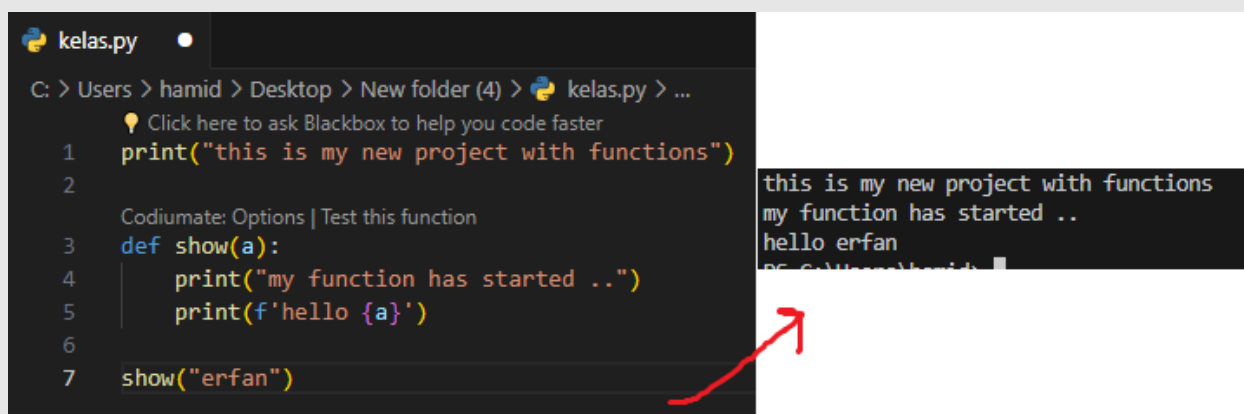
Expressions

Return *(optional)

name(args)

اگر ارگومانی قرار است تابع بگیرد موقع صدا زده شدن باید به آن پاس داده شوند

توابع برای اجرا شدن نیاز دارند که صدا زده شوند :



The screenshot shows a code editor with a file named 'kelas.py'. The code defines a function 'show(a)' that prints 'my function has started ..' and 'hello {a}'. The function is then called with the argument 'erfan'. A red arrow points from the function call to the output window.

```
kelas.py
C: > Users > hamid > Desktop > New folder (4) > kelas.py > ...
Click here to ask Blackbox to help you code faster
1 print("this is my new project with functions")
2
Codiumate: Options | Test this function
3 def show(a):
4     print("my function has started ..")
5     print(f'hello {a}')
6
7 show("erfan")
```

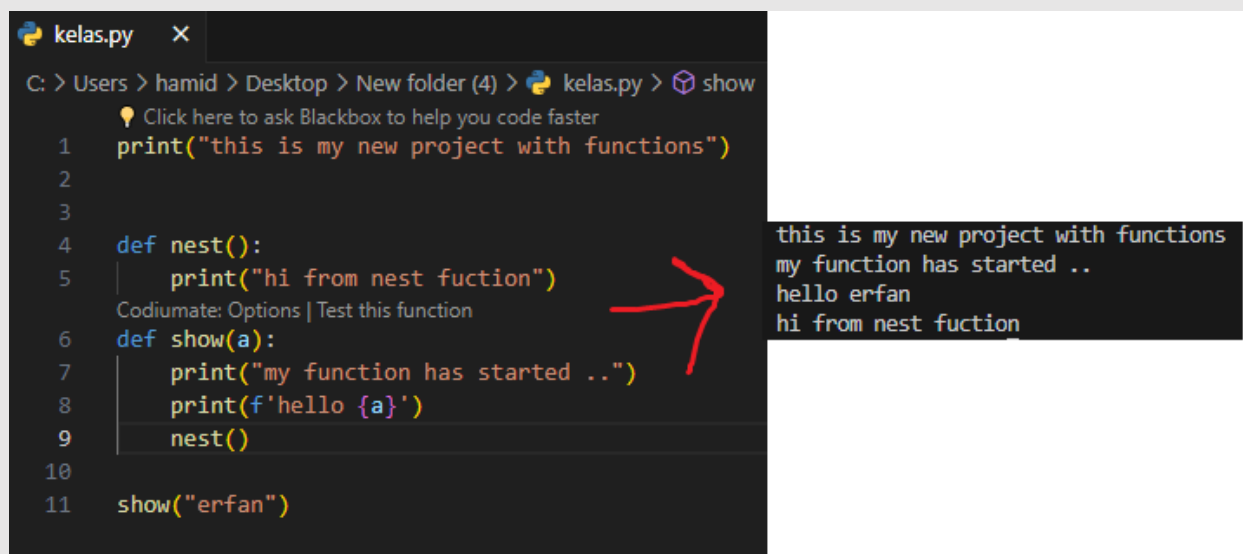
Output:

```
this is my new project with functions
my function has started ..
hello erfan
```

تابع نوشته شده که یک ارگومان `a` رو بهش دادیم و الان هرچی به تابع موقع صدا زدن بدیم داخل `a` ریخته میشود بعد داخل تابع یک سری کد هست که با اجرا شدن اجرا میشوند آخر هم تابع رو صدا زدیم و یک مقدار رو بهش به عنوان ارگومان دادیم تا داخل `a` ریخته بشود

مثل سی پلاس پلاس نیست که ما باید توابع رو بالا بنویسیم و بعد پایین اجرا کنیم توابع هر کجا که دوست داشته باشیم نوشته میشوند و هر کجا که دوست داشته باشیم اجرا میشوند

توابع هم میتوانند تو در تو بشوند :



```
kelas.py x
C: > Users > hamid > Desktop > New folder (4) > kelas.py > show
Click here to ask Blackbox to help you code faster
1 print("this is my new project with functions")
2
3
4 def nest():
5     print("hi from nest fuction")
Codiumate: Options | Test this function
6 def show(a):
7     print("my function has started ..")
8     print(f'hello {a}')
9     nest()
10
11 show("erfan")
```

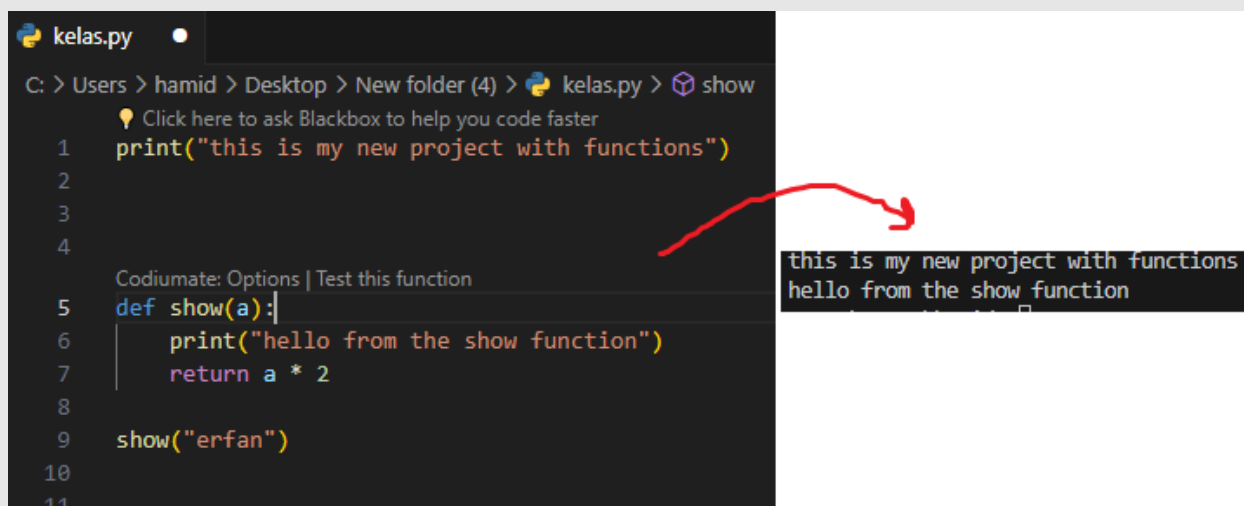
this is my new project with functions
my function has started ..
hello erfan
hi from nest fuction

اینجا اومدیم تابع `nest` رو داخل تابع `show` صدا زدیم

توابع return دار :

اگر بخواهیم تابع بعد از اتمام مقداری را برگرداند و جایی بفرستد یا ذخیره کند ما از return استفاده میکنیم

یه مثال براتون بزنم جا بیوفته : متد clear در توابع رو وقتی پرینت میکردیم None میداد چرا ؟ چون چیزی رو برنمیگردوند و فقط کد های داخلش اجرا میشد (لیست را خالی میکرد) ولی مثلا تابع copy میداد و اون کپی از لیست رو return میکنه و ما مثلا میریزیمش داخل یه متغیر یا پرینت میکنیمش.



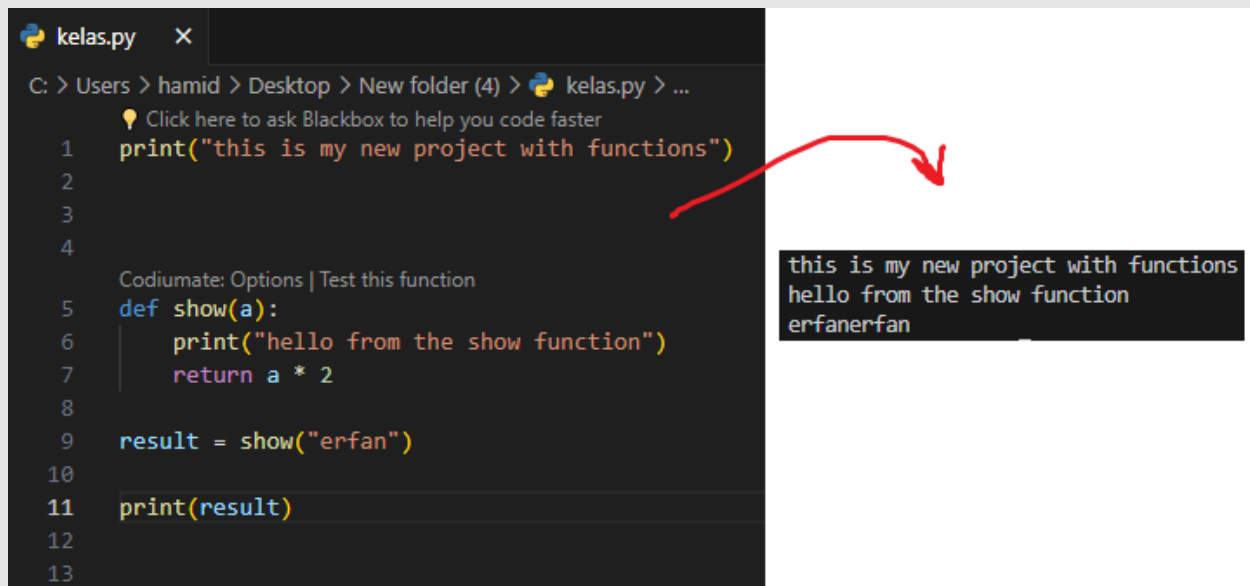
The screenshot shows a Python IDE with a file named 'kelas.py'. The code in the editor is as follows:

```
1 print("this is my new project with functions")
2
3
4
5 def show(a):
6     print("hello from the show function")
7     return a * 2
8
9 show("erfan")
10
11
```

A red arrow points from the function call 'show("erfan")' on line 9 to a terminal window on the right. The terminal displays the output of the function call:

```
this is my new project with functions
hello from the show function
erfan2
```

اینجا میبینید که a ضرب در 2 در خروجی چاپ نشده چرا ؟ چون فقط برگردونده شده یعنی چاپ نشده یا در متغیری ریخته نشده :

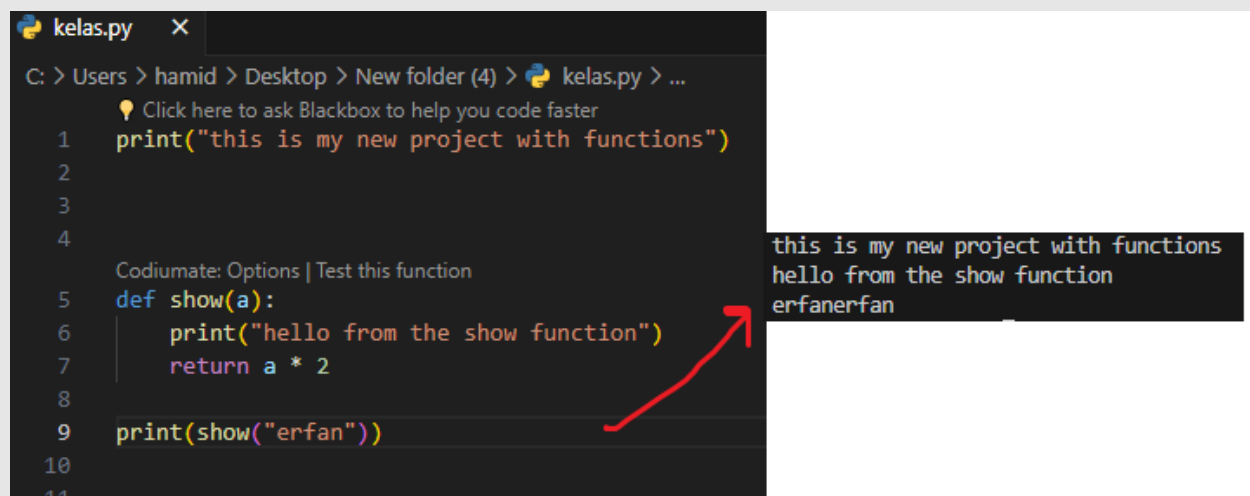


```
kelas.py X
C: > Users > hamid > Desktop > New folder (4) > kelas.py > ...
Click here to ask Blackbox to help you code faster
1 print("this is my new project with functions")
2
3
4
Codiumate: Options | Test this function
5 def show(a):
6     print("hello from the show function")
7     return a * 2
8
9 result = show("erfan")
10
11 print(result)
12
13
```

this is my new project with functions
hello from the show function
erfanerfan

حالا اون چیزی که return میشه داخل متغیر result ذخیره میشه و ا اونو پرینت کردیم

میشه اون رو متسقیما هم پرینت کرد (گفتم زمانی از متغیر استفاده میکنیم که بخواییم اون چیزی که return شده رو بعدا استفاده کنیم) :

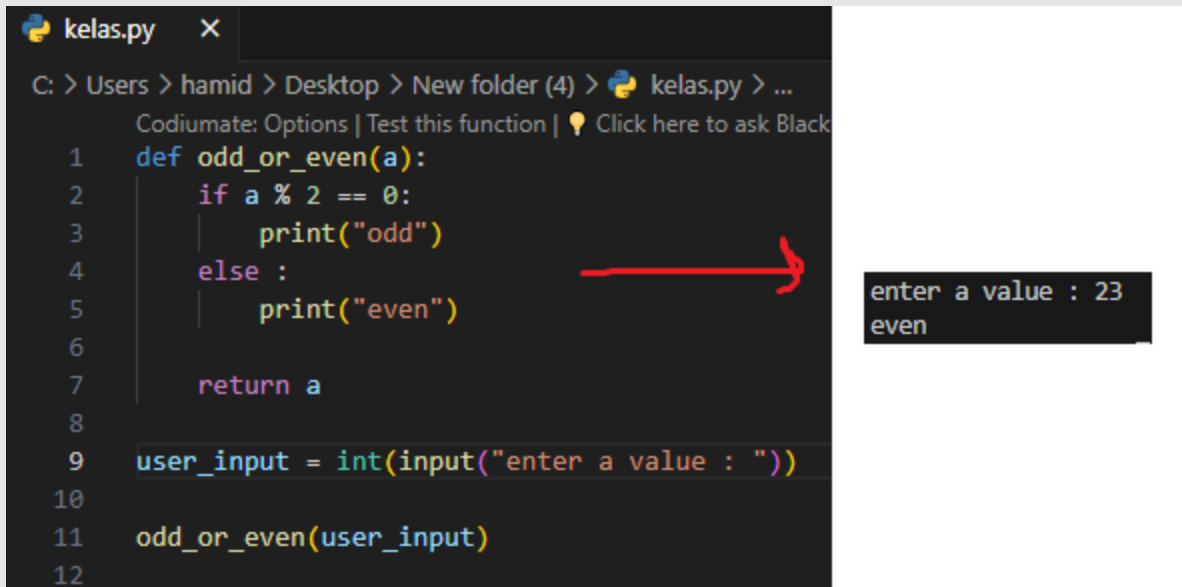


```
kelas.py X
C: > Users > hamid > Desktop > New folder (4) > kelas.py > ...
Click here to ask Blackbox to help you code faster
1 print("this is my new project with functions")
2
3
4
Codiumate: Options | Test this function
5 def show(a):
6     print("hello from the show function")
7     return a * 2
8
9 print(show("erfan"))
10
11
```

this is my new project with functions
hello from the show function
erfanerfan

اینجا اون چیز return میشه داخل print و اونو ذخیره نکردیم و فقط مستقیما چاپ کردیم

گفتیم که توابع وظیفه کوتاه و سازمان دهی کردن کد های مارو دارند :



The image shows a code editor window titled 'kelas.py' with a file path 'C: > Users > hamid > Desktop > New folder (4) > kelas.py > ...'. The code defines a function 'odd_or_even(a)' that checks if a number is odd or even. A red arrow points from the 'print("even")' line to the output window. The output window shows 'enter a value : 23' and 'even'.

```
1 def odd_or_even(a):
2     if a % 2 == 0:
3         print("odd")
4     else :
5         print("even")
6
7     return a
8
9 user_input = int(input("enter a value : "))
10
11 odd_or_even(user_input)
12
```

enter a value : 23
even

اینجا داخل تابع بحث زوج و فرد بودن رو بررسی کردیم که اون ارگومان ورودی اگر باقیمانده بر 2 صفر بود چاپ کنه زوج وگرنه چاپ کنه فرد

پایین هم ورودی رو از کاربر گرفتیم و تبدیل ه عددش کردیم بعد تابع رو صدا زدیم و بهش پاسش دادیم

اون return هم به محض احتیاط گذاشتیم تا اگر برنامه نویس دیگه ای داشت با کد ما کار میکرد و دلش خواست مقدار a رو جایی ذخیره کنه ازش استفاده کنه

حالا 100 بار هم که شده نیازی نیست کد رو هعی بنویسیم بجاش تابع رو 100 بار صدا میزنیم :

```
kelas.py X
C: > Users > hamid > Desktop > New folder (4) > kelas.py > ...
Codiumate: Options | Test this function | Click here to ask Blackbox to help you code
1 def odd_or_even(a):
2     if a % 2 == 0:
3         print("odd")
4     else :
5         print("even")
6
7     return a
8
9 for i in range(100):
10     user_input = int(input("enter a value : "))
11     odd_or_even(user_input)
12
```

الان 99 بار حلقه میخوره و 99 بار تابع اجرا میشه

یک مثال دیگه که جا بیوفته و همچنین کاربری در جنگو :

```
kelas.py X
C: > Users > hamid > Desktop > New folder (4) > kelas.py > ...
Codiumate: Options | Test this function | Click here to ask Blackbox to help you code
1 def login():
2     pass1 = input("enter your password : ")
3     pass2 = input("enter your password again .. : ")
4
5     if pass1 == pass2 :
6         print('you are logged in ... exiting the loop')
7         return False
8     else :
9         print("wrong password ...")
10        return True
11
12 while login() :
13     print("entering the loop ..")
14
```

```
enter your password : 123
enter your password again .. : 12
wrong password ...
entering the loop ..
enter your password : 1234
enter your password again .. : 124
wrong password ...
entering the loop ..
enter your password : 1234
enter your password again .. : 1234
you are logged in ... exiting the loop
```

توی این مثال اومدیم کاری کردیم که اگر پسر 1 با 2 برابر بود False برگردد و اگر نبود True چرا ؟ اگر پسورد غلط بود True برگردونده میشه به شرط حلقه و حلقه ادامه پیدا میکنه و هر دفعه تابع اجرا میشه . اگر پسورد درست بود False برمیگرده به شرط حلقه و حلقه بسه میشه
