آرگومان های متغیر

با استفاده از دستورات خاص *args و **kwargs می‌توان تعداد دلخواهی از آرگومان‌ها را به تابع ارسال کرد. همانطور که در درس‌های قبل ذکر شد، هنگام فراخوانی تابع باید به تعداد پارامترهایی که در داخل پرانتز تعریف شده‌اند، آرگومان به تابع ارسال کرد. گاهی اوقات در برنامه نویسی ممکن است بخواهید که در هر بار فراخوانی تابع تعداد دلخواهی آرگومان به آن ارسال کنید. این کار با استفاده از * و ** ممکن است. به کد زیر توجه کنید :

```
def varArguments(*args):
    total = 0;
    for number in args:
        total = total + number;
    return total;

print("1 + 2 + 3 = {0}".format(varArguments(1, 2, 3)));
print("1 + 2 + 3 + 4 = {0}".format(varArguments(1, 2, 3, 4)));
print("1 + 2 + 3 + 4 + 5 = {0}".format(varArguments(1, 2, 3, 4, 5)));
```

```
1 + 2 + 3 = 6
1 + 2 + 3 + 4 = 10
1 + 2 + 3 + 4 + 5 = 15
```

ابتدا به این نکته توجه کنید که نامهای args و kwargs اختیاری هستند و هم نام دیگری می‌تواند به جای آنها به کار رود. تنها چیزی که مهم است تعداد علامت * می‌باشد که در ادامه کاربرد آنها را توضیح می‌دهیم.

همانطور که در کد بالا مشاهده می‌کنید، با قرار دادن یک علامت ستاره قبل از نام پارامتر، می‌توان هر بار که تابع را فراخوانی کرد، تعداد دلخواهی از آرگومانها را به آن ارسال کرد. وجود یک علامت * باعث می‌شود که آرگومانها در یک متغیر از جنس tuple ذخیره شوند و در نتیجه می‌توان با یک دستور for مقادیر آنها را با هم جمع کرد. این نوع پارامتر را می‌توان با پارامترهای ثابت هم به کار برد. به مثال زیر توجه کنید:

```
def varArguments(number, *args):
    print("number = ", number);
    print("args = ", args);


varArguments(1, 2, 3);
```

```
number =  1
args =  (2, 3)
```

در کد بالا اولین آرگومان به اولین پارامتر (یعنی 1 به number) و بقیه آرگومانها به args اختصاص داده می‌شوند. وقتی از چندین پارامتر در یک تابع استفاده می‌کنید فقط یکی از آنها باید دارای * بوده و همچنین از لحاظ مکانی باید آخرین پارامتر باشد. اگر این پارامتر (پارامتری که دارای علامت * است) در آخر پارامترهای دیگر قرار نگیرد و یا از چندین پارامتر علامت دار استفاده کنید با خطا مواجه می‌شوید. به مثالهای اشتباه و درست زیر توجه کنید :

```
def SomeFunction(*args, *args) #ERROR

def SomeFunction(*args, param1, param2) #ERROR

def SomeFunction(param1, param2, *args) #Correct
```

البته می‌توان پارامتر ستاره دار را در ابتدای پارامترهای دیگر قرار داد ولی پارامترهای بعد از این پارامتر یا باید دارای مقدار پیشفرض باشند

```
def varArguments(*args , number = 10):
    print("args = {0}".format(args));
    print("number = {0}".format(number));

varArguments(1, 3, 5);
```

```
args = (1, 3, 5)
number = 10
```

و یا هنگام فراخوانی تابع، باید پارامترهای بعد از این پارامتر را با استفاده از اسمشان مقداردهی کرد. به کد زیر توجه کنید:

```
def varArguments(*args , number):
    print("args = {0}".format(args));
    print("number = {0}".format(number));

varArguments(1, 3, 5, number = 10);
```

```
args = (1, 3, 5)
number = 10
```

حال فرض کنید که می‌خواهید چند list یا tuple به پارامتر ستاره دار ارسال کنید. به کد زیر توجه نمایید :

```
def varArguments(number, *args):
    print("number = {0}".format(number));
    print("args = {0}".format(args));

varArguments(1, *(2,3,4), *(5,6,7,8));
```

```
number = 1
args = (2, 3, 4, 5, 6, 7, 8)
```

همانطور که در کد بالا مشاهده می‌کنید، کافیست که آرگومان‌ها به صورت list یا tuple ارسال شوند و قبل از آنها علامت * را قرار دهیم. خط آخر کد بالا را به صورت زیر هم می‌توان نوشت :

```
varArguments(1, *[2,3,4], *[5,6,7,8]);
```

kwargs** هم شبیه args* عمل می‌کند با این تفاوت که هنگام فراخوانی تابع باید آرگومان‌ها را به صورت کلید/ مقدار به آن ارسال کرد تا به صورت dictionary ذخیره کند:

```
def varArguments(**kwargs):
    print(kwargs);

varArguments(person1 = "Jack", person2 = "Joe", person3 = "Smith");
```

```
{'person1': 'Jack', 'person2': 'Joe', 'person3': 'Smith'}
```