*args **kwargs

לפני שאסביר מהם השמות המוזרים האלה, נראה דוגמה של מתי כדאי (צריך) ומדוע צריכים אותם.

נניח שרצינו ליצור פונקציה שמחזירה 17% (כמו כמה מע״מ נצטרך לשלם) מסכום קנייה של מוצרים שונים. נניח שיש לנו 3 מוצרים.

```
def maam(prod1, prod2, prod3):
   sum_prods = prod1+prod2+prod2   # or: sum_prods = sum(prod1,prod2,
                                   #                      prod3)
   return sum_prods*0.17
```

מה היה קורה אם היו לנו פעם אחת 3 מוצרים ובפעם אחרת 5 ובעוד פעם 100 מוצרים?
אחת האפשרויות זה להוסיף עוד ועוד פרמטרים עם ברירת מחדל (בכמות המקסימלית שנחשוב שנצטרך). זה מסורבל וגם לא קל לקבוע מראש.
אפשרות אחרת יכולה להיות על ידי העברת רשימה. לשם כך יש ליצור רשימה ולהעביר אותה כפרמטר, ובפונקציה לעבור על הרשימה. זה קצת יותר אלגנטי, על מעט מלאכותי, שהרי אנחנו יוצרים רשימה שלא משמשת לדבר מלבד היכולת להעבירה כפרמטר, ובנוסף כדי להבין מה קורה נצטרך לנבור לתוך הקוד במקום להביט רק על הקריאה לפונקציה.

לפייתון יש פיתרון יותר חכם. args* הוא בעצם מעין טאפל שמכיל את כל הפרמטרים שנירצה להעביר, וכך גם אפשר להתייחס אליו. בקריאה לפונקציה ניכתוב את כל הפרמטרים שנחפוץ ובפונקציה נתייחס רק ל args*.

נכתוב את הדוגמה הנ״ל בעזרת זה:

```
def maam(*args):
   sum_prods = sum(args)
   return sum_prods*0.17
```

ובקריאה לפונקציה maam:

```
def main():
   my_maam = maam(35,48.99, 899.95)
```

ניתן להוסיף כמה מספרים שרוצים בקריאה ללא שינוי בפונקציה.
שימו לב שההתייחסות בתוך הפונקציה הייתה ל args (לא ל args*). הכוכבית מציינת שזהו פרמטר ׳גנרי׳. למעשה המילה args איננה מילת מפתח בפייתון, אלא רק מוסכמה. יכולנו להשתמש ב: parms* או כל משתנה אחר שמתחיל בכוכבית.
בכל זאת נשתמש ב args* - למה? כדי לכתוב קוד עם סטנדרטים מוסכמים בעולם הפייתוני (מאותה סיבה הגדרנו main למרות שכל שם אחר היה גם בסדר)

אם נדפיס את args בתוך maam בדוגמה הנ״ל

```
print(args)
```

נקבל:

(35,48.99, 899.95)

כלומר סידרת המספרים שהעברנו כפרמטרים, מתפרשת כטאפל בתוך הפונקציה.

כעת נותר להסביר מהו: kwargs** (key word arguments)

זה מציין אפשרות להעברת מספר שרירותי כלשהו של פרמטרים עם מילות מפתח. פייתון תפתח את kwargs כמילון בעל מפתחות וערכים צמודים להם.

לדוגמה:

```
def food_sample(**kwargs):
    if ‘fruit’ in kwargs:
        print(‘My favorit fruit is: ‘ + kwargs[‘fruit’])
    else:
        print(‘Could not find any fruits’)
```

והקריאה לפונקציה:

```
def main():
    food_sample(fruit=‘mango’, carb = ‘bread’)
```

במקרה דנן איננו משתמשים בפרמטר מילת המפתח carb , אבל אין תלונות לפייתון והוא זמין באותה צורה כמו fruit בתוך הפונקציה.
יתרה מזאת, אם נדפיס את kwargs נקבל:

{‘fruit’:’mango, ‘carb’:’bread’}

גם כאן kwargs היא קונבנציה של מתכנתי פייתון ולא מילת מפתח. מה שהופך זאת למעין מילון אלה שתי הכוכביות המקדימות.

ניתן גם לשלב את שני המושגים בפוקציה אחת (ובקריאה לה) למשל:

```
def combine(*args, **kwargs):
    print (“I’d like {}  {}”.format(args[0], kwargs[‘food’])
```

ובקריאה:

```
def main():
    combine(10, 20, 30, food=‘grapes’)
```

וכתוצאה מכך נקבל:

I’d like 10 grapes

שימו לב שהייתה התעלמות מהערכים 20 ו 30, כי השתמשנו רק בפרמטר מיקומי ראשון (args[0]) ובפרמטר מילת מפתח כפרמטר שני ופייתון מסוגלת להבחין בהבדלים על ידי צורת הקריאה.

אם משתמשים גם ב args* וגם ב kwargs** חייבים קודם לכתוב את args*.

כאן המקום לציין שגם אם אינכם רואים טעם ביכולות אלה של פייתון ולעולם לא תשתמשו בהן, זה עדיין (כמו כל נושא נוסף) חשוב לדעת על מנת לקרוא קוד של אחרים שאולי כן השתמשו באפשרויות אלה.