# Funksiya

Funksiya ma'lum bir vazifani bajarishga mo'ljallangan kodlar yig'indisi. Biz shu paytgacha bir nechta tayyor funksiyalardan foydalanib keldik. Misol uchun [print()](https://python.sariq.dev/ilk-qadamlar/03-print#print) funksiyasi konsolga matn chiqarish uchun, [range()](https://python.sariq.dev/ozgaruvchilar-va-malumot-turlari/08-list-tuple#range-funktsiyasi) funksiyasi esa ma'lum oraliqdagi sonlarni yaratish uchun ishlatiladi.

Aslida har qanday funksiyaning ortida ham bir necha qatordan iborat kod bo'ladi, lekin biz funksiyaga murojat qilganda uning nomini yozamiz xolos. Funksiya ortidagi kod esa biz uchun yashirin bo'lib qolaveradi. Funksiyalarning qulayligi ham shunda. Dastur davomida ma'lum bir kodlarni qayta-qayta yozmaslik uchun biz ularni jamlab, bitta funksiya ichiga joylashimiz va dastur davomida bu kodlarga funksiya nomi orqali murojat qilishimiz mumkin.

**Eng oddiy funksiya yaratamiz:** salom deb nomlangan funksiya yaratamiz. Bu funksiya murojat qilganimizda konsolga "Assalom alaykum!" degan xabarni chiqarsin.

def salom():

print("Assalomu alaykum!")

Kodni qatroma-qator tahlil qilaylik:

1. Avvalo def operatori yordamida Pythonga funksiya yaratayotganimizni bildirdik. def dan so'ng esa funksiyamizga nom berdik va qavslarni ochib, yopdik. Bizning funksiyamiz foydalanuvchidan hech qanday qiymat qabul qilmaydi, shuning uchun ham qavs ichi bo'sh. Keyingi misollarda foydalanuvchidan qiymat qabul qiluvchi funksiyalarni ham ko'ramiz.
2. def qatoridan keyin o'ngga surib yozilgan har qanday kod funksiyaning badani hisoblanadi. 2-qatorda biz uchta ketma-ket qo'shtirnoq ichida funksiya haqida ma'lumot berdik. Python mana shu ma'lumotni o'qib, dasturchi funksiya haqida bilmoqchi bo'lganda aynan shu matnni ko'rsatadi.
3. Oxirgi qatorimizda esa "Assalomu alaykum!" matnini konsolga chiqarishni buyurdik. Bizning sodda funksiyamizning asosiy vazifasi ham shu.

Avvalgi sodda funksiyamiz foydalanivchidan hech qanday qiymat olmaydi va barchaga birday "Assalomu alaykum!" deb javob qiladi. Keling funksiyaga o'zgartirish kiritamiz, funksiya foydalanuvchi ismini qabul qilib, unga ismi bilan murojat qilsin. Buning uchun funksiya nomidan keyin, qavs ichida foydalanuvchi berishi kerak bo'lgan qiymatni ko'rsatamiz.

def salom(ism):

print(f"Assalomu alaykum, hurmatli {ism.title()}!")

salom\_ber(“fayoz”)

**Natija:** *Assalomu alaykum, hurmatli Fayoz!*

Funksiya yaratishda, qavs ichida berilgan, funksiya to'g'ri ishlashi uchun uzatiladigan qiymat **parameter** deb ataladi. Yuqoridagi misolda ism bu salom funksiyasining parametri.

Foydalanuvchi funksiyaga murojat qilishda funksiyaga uzatgan qiymat esa **argument** deb ataladi. salom\_ber(“fayoz”) kodida 'fayoz' bu argument.

Shunday funksiyalar bor, bir emas bir nechta parameter talab qilishi mumkin, foydalanuvchi esa o'z navbatida bir nechta argumentlar taqdim qilishi kerak. Funksiyaga argument uzatishning bir necha usuli bor, keling ular bilan birma bir tanishamiz.

def toliq\_ism(ism, familiya):

print(f"Foydalanuvchi ismi: {ism.title()}\n"

f"Foydalanuvchi familiyasi: {familiya.title()}")

toliq\_ism('olim','hakimov')

**Natija:**

*Foydalanuvchi ismi: Olim*

*Foydalanuvchi familiyasi: Hakimov*

Funksiyadan tashqarida hosil qilingan o’zgaruvchilar global o’zgaruvchilar hisoblanadi. Global o’zgaruvchilar kodning istalgan qismida (funksiya ichida ham, tashqarisida ham) ishlatilishi mumkin. Quyidagi kodda funksiya tashqarisida o’zgaruvchi hosil qilamiz va uni funksiya ichida ishlatamiz:

x = "qiziq"

def funksiya():

print("Dasturlash juda "+ x)

funksiya()

Oddiy holatda funksiya ichida hosil qillingan o’zgaruvchi lokal o’zgaruvchi hisoblanadi. Ammo funksiya ichida ham global o’zgaruvchi hosil qilish mumkin. Buning uchun global kalit so’zi ishlatiladi.

def funksiya():

global x

x = "shirin"

print("Olma "+x)

funksiya()

print ("Olma "+x)

Agar global o’zgaruvchining qiymatini funksiya ichida o’zgartirmoqchi bo’lsangiz ham global kalit so’zi ishlatiladi:

x = "shirin"

def funksiya():

global x

x = "foydali"

print("Olma "+ x)

funksiya()

print ("Olma "+ x)