

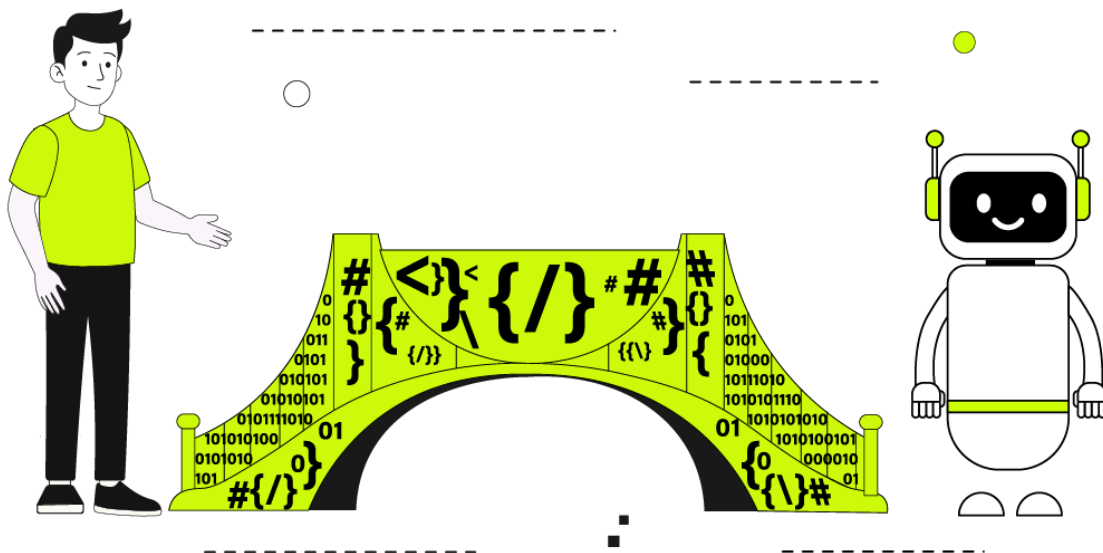
შეხვედრა 2: პირველი ნაბიჯები Python-ში – ჩვენი ჩატბოტის "მეხსიერება"

გამარჯობა! წინა შეხვედრაზე ჩვენ გავეცანით ხელოვნური ინტელექტის საოცარ სამყაროს და ჩვენს პირველ პროგრამასაც კი "ვათქმევინეთ" სალამი. დღეს კი ვინყებთ ჩვენი ჩატბოტისთვის "ტვინის" და "მეხსიერების" შექმნას. ამისთვის გამოვიყენებთ პროგრამირების ენას, რომელსაც Python (პითონი) ჰქვია.

1. რა არის პროგრამირების ენა?

წარმოიდგინე, რომ კომპიუტერი არის რობოტი, რომელსაც ძალიან სწრაფად და ზუსტად შეუძლია დავალებების შესრულება. მაგრამ ის თვითონ ვერ გადაწყვეტს, რა გააკეთოს. მას ყოველთვის სჭირდება მითითებები. სწორედ პროგრამირების ენის დახმარებით ადამიანები კომპიუტერს ვეუბნებით, რა უნდა გააკეთოს.

პროგრამირების ენა არის შუამავალი ადამიანსა და კომპიუტერს შორის. ის ჩვენს იდეებსა და ლოგიკურ აზროვნებას ისეთ ფორმატში აქცევს, რომ კომპიუტერმა გაიგოს და შეასრულოს ჩვენი ბრძანებები. სხვადასხვა ენას (მაგალითად, Python, JavaScript, C++) სხვადასხვა დანიშნულება აქვს, მაგრამ ყველა მათგანის მიზანი ერთია — ხელი შეუწყოს ადამიანისა და მანქანის კომუნიკაციას.



2. რატომ Python-ი?

წარმოიდგინე, რომ გინდა ესაუბრო ადამიანს სხვა ქვეყნიდან. თუ ორივემ იცით ინგლისური, ერთმანეთს ადვილად გაუგებთ, რადგან ის საერთაშორისო ენად ითვლება. პროგრამირების სამყაროში, განსაკუთრებით კი ხელოვნურ ინტელექტთან სამუშაოდ, ასეთი საერთაშორისო ენა არის Python-ი.

2.1. Python-ი, როგორც ადამიანურ ენასთან ყველაზე ახლოს მდგომი პროგრამირების ენა

ბევრი პროგრამირების ენისგან განსხვავებით, რომელთა კოდი რთულ სიმბოლოებსა და მკაცრ წესებს შეიცავს, Python-ის კოდი ძალიან ჰგავს ინგლისური ენისთვის დამახასიათებელ მარტივ წინადადებებს. მისი წაკითხვა და გაგება ბევრად მარტივია.

მაგალითად, თუ გვინდა, ეკრანზე დავბეჭდოთ "გამარჯობა", Python-ში ვწერთ:

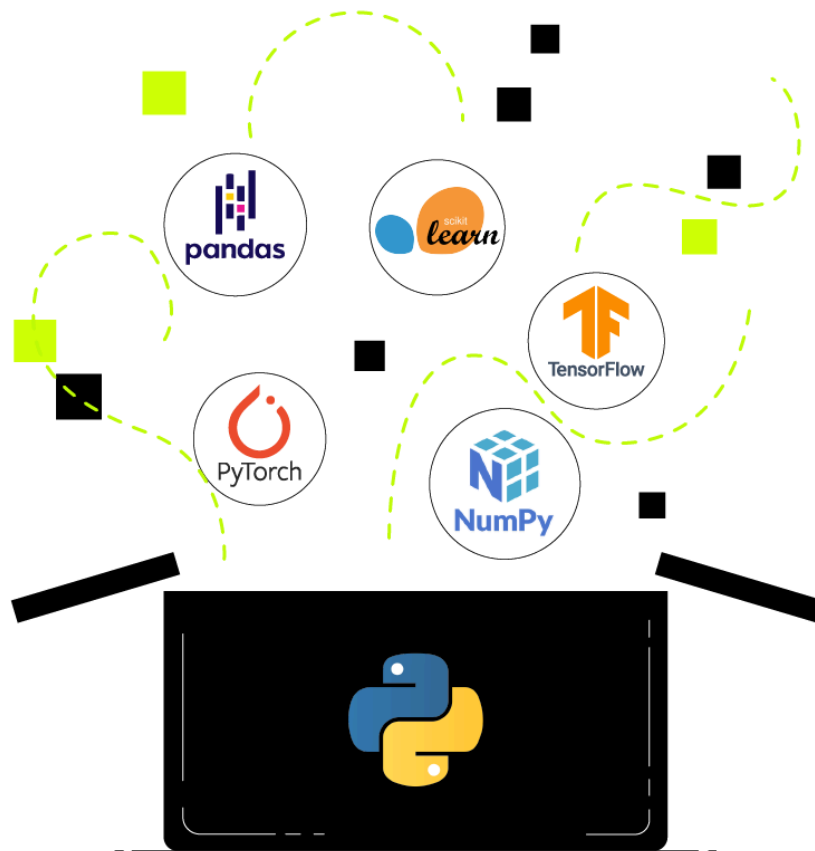
```
print("გამარჯობა")
```

ეს სიმარტივე გვეხმარება, მეტი დრო დავუთმოთ პრობლემის გადაჭრაზე ფიქრს და ნაკლები - რთული სინტაქსის დამახსოვრებას.

2.2. რატომ ირჩევენ Python-ს ხელოვნურ ინტელექტთან სამუშაოდ?

წარმოიდგინე, რომ სახლს აშენებ. შეგიძლია, ყველა იარაღი თავად გამოთალო ხისგან, ან შეგიძლია მიხვიდე მაღაზიაში და იყიდო მზა, პროფესიონალური იარაღების ნაკრები. Python-ს მოჰყვება ასეთი "იარაღების" უზარმაზარი ნაკრები, რომლებსაც **ბიბლიოთეკები** ჰქვია.

ხელოვნური ინტელექტის სფეროში მომუშავე ინჟინრებმა და მეცნიერებმა შექმნეს უამრავი ბიბლიოთეკა (Pandas, NumPy, Scikit-learn, TensorFlow, PyTorch), რომლებიც უამრავ ურთულეს ოპერაციას ჩვენთვის ძალიან ამარტივებს. ჩვენ არ გვინევს "ველოსიპედის თავიდან გამოგონება" და შეგვიძლია, პირდაპირ გამოვიყენოთ ეს მზა ინსტრუმენტები ჩვენი მოდელების ასაგებად. სწორედ ამიტომ, კომპანიები, როგორიცაა Google, Meta, Netflix და NASA, აქტიურად იყენებენ Python-ს.



3. ცვლადები და მონაცემთა ტიპები

კომპიუტერს რომ რაიმე დავამახსოვრებინოთ, გვჭირდება ადგილი, სადაც ამ ინფორმაციას შევინახავთ. პროგრამირებაში ასეთ ადგილს **ცვლადი** ჰქვია.

3.1. ცვლადი, როგორც ინფორმაციის შესანახი "ყუთი"

წარმოიდგინე ცვლადი, როგორც ყუთი. შენ შეგიძლია, ამ ყუთს დააწერო სახელი (ცვლადის სახელი) და შიგნით მოათავსო რაიმე ნივთი (მნიშვნელობა).

მაგალითად, თუ გვინდა, შევინახოთ ჩვენი ბოტის სახელი, შეგვიძლია, შევქმნათ "ყუთი" სახელით `bot_name` და შიგნით ჩავდეთ ტექსტი "ანია":

```
bot_name = "ანია"
```

ამიერიდან, როცა პროგრამაში ვახსენებთ `bot_name`-ს, კომპიუტერი მიხვდება, რომ ჩვენ ვგულისხმობთ ტექსტს "ანაITს". ჩვენ შეგვიძლია, ნებისმიერ დროს შევცვალოთ ყუთის შიგთავსი.

3.2. ძირითადი მონაცემთა ტიპები

ყუთში სხვადასხვა ტიპის ნივთის ჩადება შეგვიძლია. ანუ ცვლადს შეგვიძლია სხვადასხვა ტიპის მნიშვნელობა მივანიჭოთ:

ტექსტი (string, str): ნებისმიერი ტექსტური ინფორმაცია, რომელიც ბრჭყალებშია მოქცეული.

```
user_name = "ლუკა"
```

```
city = "თბილისი"
```

მთელი რიცხვი (integer, int): ნებისმიერი მთელი რიცხვი, ბრჭყალების გარეშე.

```
user_age = 17
```

```
current_year = 2024
```

ათწილადი (float): ნებისმიერი რიცხვი, რომელსაც აქვს წილადი ნაწილი (წერტილით გამოყოფილი).

```
pi = 3.14
```

```
temperature = 25.5
```

კი-არა ტიპის მონაცემი (boolean, bool): ასეთ მონაცემს აქვს მხოლოდ ორი შესაძლო მნიშვნელობა: `True` (მართალია) ან `False` (მცდარია). გამოიყენება პირობების შესამოწმებლად.

```
is_student = True
```

```
is_raining = False
```

3.3. `print()` ფუნქცია: როგორ "ავალაპარაკოთ" ჩვენი პროგრამა

`print()` არის ბრძანება, რომლითაც კომპიუტერს ვეუბნებით: "ეკრანზე გამომიტანე ის, რაც ფრჩხილებშია". ეს ჩვენი მთავარი საკომუნიკაციო საშუალებაა. ჩვენ შეგვიძლია დავბეჭდოთ როგორც პირდაპირ ტექსტი, ასევე ცვლადის მნიშვნელობა.

სავარჯიშო 1:

```
# ამ ველში ჩანერე კოდი, რომელიც დაბეჭდავს ტექსტს "გამარჯობა!"
```

```
# და შემდეგ ცვლადის `user_age`-ის მნიშვნელობას.
```

```
user_age = 17
```

```
# აქ დანერე შენი კოდი:
```

4. მონაცემების დინამიურად გამოყენება

ცვლადების ნამდვილი ძალა მაშინ ვლინდება, როცა მათ ტექსტთან ერთად ვიყენებთ, რათა შევქმნათ დინამიური, ცვალებადი წინადადებები.

4.1. ფორმატირებული სტრიქონები (f-strings)

წარმოიდგინე, რომ გინდა დაბეჭდო წინადადება: "გამარჯობა, მე მქვია ანITA". ამის გაკეთება მარტივია. მაგრამ რა მოხდება, თუ ბოტის სახელი ცვლადში გვაქვს შენახული და მისი გამოყენება გვინდა?

სწორედ აქ შემოდის `f-string`-ი. თუ ტექსტს ბრჭყალების წინ დავუწერთ ასო `f`-ს, ჩვენ შევძლებთ ტექსტში პირდაპირ ჩავსვათ ცვლადები ფიგურული ფრჩხილების `{ }` გამოყენებით.

```
bot_name = "ანITA"
```

```
user_name = "ლუკა"
```

```
print(f"გამარჯობა, {user_name}! მე ვარ {bot_name}.")
```

შედეგი: გამარჯობა, ლუკა! მე ვარ ანITA.

ეს ძალიან მოსახერხებელი და თანამედროვე გზაა ტექსტისა და ცვლადების გასაერთიანებლად.

სავარჯიშო 2:

```
# შექმენი ორი ცვლადი: `product` და `price`.
```

```
# შემდეგ, `f-string`-ის გამოყენებით, დაბეჭდე წინადადება:
```

```
# "პროდუქტის ფასია 200 ლარი."
```

```
# (ოღონდ 200 და "ლარი" ცვლადებიდან უნდა მოდიოდეს)
```

```
product = "ტელეფონი"
```

```
price = 200
```

```
# აქ დაწერე შენი კოდი:
```

4.2. კომენტარების (#) როლი: შენიშვნების დატოვება საკუთარი თავისთვის და სხვებისთვის

როცა პროგრამას წერ, ხშირად დაგჭირდება ჩაინიშნო შენიშვნები, თუ რას აკეთებს კოდის ესა თუ ის ნაწილი. ამისთვის გამოიყენება **კომენტარები**. Python-ში კომენტარი იწყება # სიმბოლოთი. ყველაფერი, რაც ამ სიმბოლოს შემდეგ წერია ერთ ხაზზე, კომპიუტერის მიერ იგნორირდება.

```
# ეს ცვლადი ინახავს მომხმარებლის ასაკს
```

```
user_age = 17
```

```
print("Hello!") # კომენტარის დაწერა შეიძლება ხაზის ბოლოსაც
```

კომენტარები კოდს უფრო გასაგებს და წაკითხვადს ხდის, რაც ძალიან მნიშვნელოვანია განსაკუთრებით მაშინ, როცა დიდ პროექტზე მუშაობ ან შენს კოდს სხვას უზიარებ.

დავალება 2: პერსონალური მისალმება

დროა, ჩვენს ჩატბოტს ვასწავლოთ როგორ მიესალმოს მომხმარებელს. საბოლოოდ, ჩვენი კოდის მიზანია, დაბეჭდოს მისალმება, რომელიც გამოიყენებს როგორც ბოტის, ასევე მომხმარებლის სახელს.

დავალება 2.1: შეცდომის პოვნა და გასწორება

მოცემულ კოდში დაშვებულია შეცდომა. შენი ამოცანაა, იპოვო და გაასწორო ის, რათა პროგრამამ სწორად დაბეჭდოს მისალმება.

```
# შეცდომით დაწერილი კოდი
```

```
bot_name = "ანია"
```

```
user_name = "ლუკა"
```

```
print("გამარჯობა, {user_name}! მე ვარ {bot_name}.")
```

დავალება 2.2: კოდის დასრულება

მოცემულ პროგრამულ კოდს აკლია ერთი მნიშვნელოვანი ნაწილი. დაამატე მხოლოდ ის, რაც აუცილებელია იმისთვის, რომ პროგრამამ სწორად დაბეჭდოს მისალმება.

```
# კოდი, რომელსაც აკლია ერთი სტრიქონი
```

```
bot_name = "ანია"
```

```
user_name = "ლუკა"
```

```
...
```

დავალება 2.3: შეიმუშავე პროგრამული კოდი

დაწერე პროგრამული კოდი, შექმენი პროგრამა, რომელიც შექმნის ორ ცვლადს (`bot_name` და `user_name`) და `f-string`-ის გამოყენებით დაბეჭდავს მათზე დაფუძნებულ მისალმებას.

```
# დაწერე შენი კოდი აქ:
```

სწორი პასუხი (პროგრამული კოდი სრულად):

```
# ამ კოდის საშუალებით შეგიძლია გადაამოწმო შენი ნამუშევარი
```

```
bot_name = "ანია"
```

```
user_name = "ლუკა"
```

```
print(f"გამარჯობა, {user_name}! მე ვარ {bot_name}.")
```


წარმატებებს გისურვებ! შენ ახლახან შექმენი პროგრამის პირველი "მეხსიერება".