



Distance Learning System



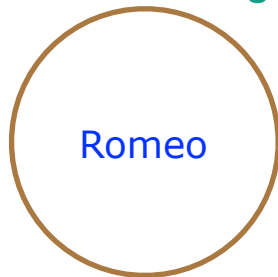
Opsezi i prostori imena

Python and programming fundamentals

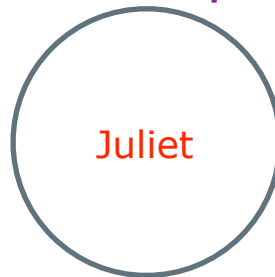
Šta je opseg

- Opseg je deo programa unutar koga je neka vrednost dostupna
- Opsege možemo posmatrati kao skupove čiji se elementi uzajamno ne mogu videti

House of Montague

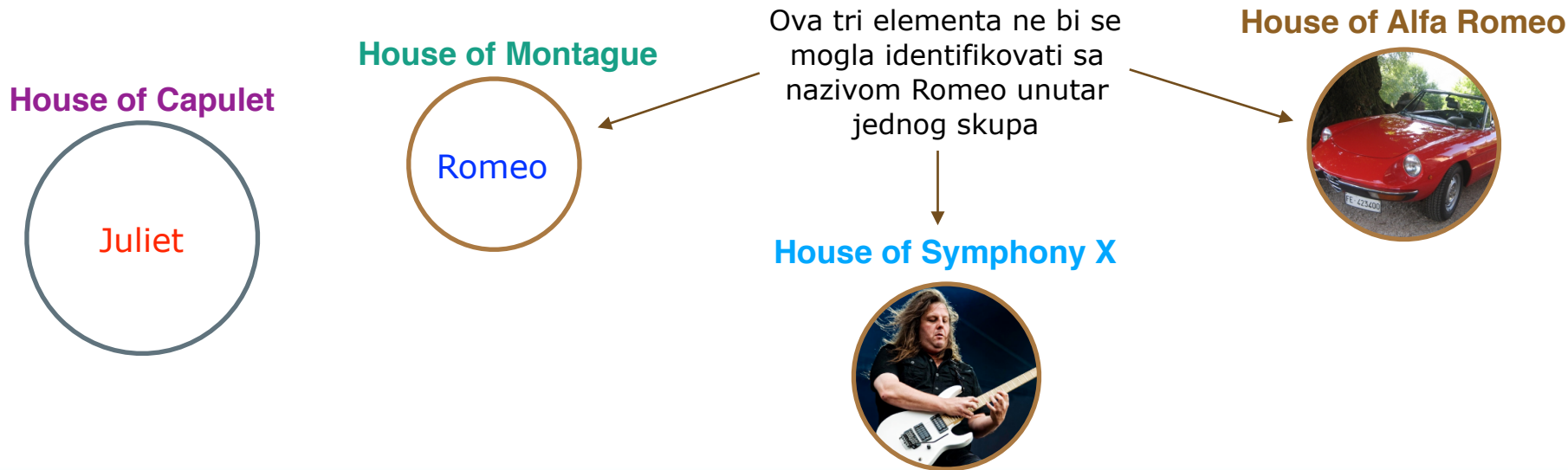


House of Capulet



Zašto postoje opsezi

- Opsezi omogućavaju pojavljivanje istoimenih identifikatora unutar istog programa
- Kada ne bi postojali opsezi, program bi vrlo brzo postao nepregledan



Korišćenje opsega

- Opseg se implicitno formira stvaranjem funkcija koda ili modula
- Obično se sve unutar funkcije ne može videti izvan nje, a sve izvan funkcije može videti unutar funkcije
- Opseg u kome se nalaze elementi vidljivi iz svih delova programa, naziva se **globalni opseg**, dok se opsezi koji su vidljivi samo određenim delovima programa (tela funkcije, klase ili modula), nazivaju **lokalni opsezi**

a je definisano globalno

```
a = 2
def pow():
    print(a*a)
pow()
```

4

a je definisano lokalno

```
def pow():
    a = 2
    pow()
    print(a*a)
```

TypeError: name 'a' is not defined

Objavljivanje globalne promenljive

- Iz funkcije ili metode, možemo promenljivu eksplicitno označiti globalnom ključnom rečju **global**
- Ovo će kreirati novu globalnu promenljivu ukoliko globalna promenljiva ne postoji

```
def pow():  
    global a  
    a = 2
```

```
pow()  
print(a*a)
```

Funkcija nije pozvana

Pa nema globalne promenljive

4

```
def pow():  
    global a  
    a = 2
```

```
print(a*a)
```

NameError: name 'a' is not defined

Python moduli

- Kada bismo pisali kompletan program u jednom fajlu, on bi vrlo brzo prestao da bude pregledan
- Zbog ovoga, programi se često prave u zasebnim fajlovima, koji se zatim učitavaju po potrebi
- Ovakvi fajlovi ili grupe fajlova, nazivaju se **moduli**
- Osim preglednosti, moduli takođe omogućavaju ponovnu upotrebljivost koda, lakše pronalaženje grešaka, lakšu izmenjivost i slično
- Modul se kreira pravljenjem python fajla

mymodule.py

```
print("Hello from module")
```

Učitavanje modula

- Python modul učitava se u tekući program komandom **import**
- Fajl koji se učitava mora da bude u istom direktorijumu u kome je i program koji izvršava naredbu import

myprogram.py

```
import mymodule
```

izlaz

Hello from module

mymodule.py

```
print("Hello from module")
```

Pristupanje elementima modula

- Elementi modula **nisu** direktno dostupni nakon učitavanja modula
- Sledeći kod prijavljuje grešku:

mymodule.py

```
x = 10
```

myprogram.py

```
import mymodule  
print(x)
```

→ NameError: name 'x' is not defined

- Da bi se pristupilo elementima drugog modula, koristi se **operator opsega** (scope operator) **.** :

mymodule.py

```
x = 10
```

myprogram.py

```
import mymodule  
print(mymodule.x)
```

→ 10

← Scope operator

Struktuiranje modula

- Modul može imati bilo kakav sadržaj, sve dok je u pitanju sintaksno validan Python kod
- Moduli ipak ne bi trebalo da sadrže kod koji se izvršava implicitno, već radije samo definicije (funkcije, klase ili promenljive)

mymodule.py

```
def add(a,b):  
    return a + b
```

myprogram.py

```
import mymodule  
print(mymodule.add(2,3))
```

Dobro

mymodule.py

```
print(2+3)
```

myprogram.py

```
import mymodule
```

Loše

Ponovo upotrebljivi moduli

- Module ne moramo vezivati za jedan projekat
- Module koje hoćemo globalno da koristimo, možemo dodati u putanju projekta
- Programabilno dodavanje

```
import sys
sys.path.append("/mydir/someotherdir")
```

- Dodavanje putanje u sistemsku varijablu PYTHONPATH


```
set PYTHONPATH=%PYTHONPATH%;c:/mydir/someotherdir
```

Paketi

- Grupisanje srodnih delova koda po fajlovima rešava problem u manjim projektima
- U većim projektima, može doći do preklapanja samih grupacija
- Tada je dobro grupisati fajlove po direktorijumima
- Direktorijumi koji sadrže tematski srodne module jednog projekta, nazivaju se **paketi**
- Da bi se učitao modul iz nekog paketa, potrebno je navesti naziv tog paketa (ili podpaketa) prilikom importa

```
C:\pythonpractice\myapp.py  
C:\pythonpractice\mypackage\mod1.py  
C:\pythonpractice\mypackage\mod2.py
```

`import mypackage.mod2`



Referenciranje paketa

- U prethodnom primeru, nakon učitavanja modula, trebalo je unositi kompletnu putanju do modula kako bi mu se pristupilo (`mypackage.mod2.myfunction()`)
- Moguće je pridružiti paket kodu, tako da se on podrazumeva

```
from pkg import mymodule
```

- Sada se podrazumeva paketa `pkg`, pa se modul može koristiti na sledeći način:

```
mymodule.myfunction()
```

Inicijalizacija paketa

- Prilikom importa paketa python automatski pokušava da importuje i fajl `__init__.py` koji se nalazi unutar paketa
- Definisanjem ovog fajla i njegovog sadržaja, možemo presresti učitavanje paketa sopstvenim funkcionalnostima

`__init__.py`

```
print("Package is going to be loaded")
```

Alias

- Postoje situacija u kojima želimo da radimo sa istoimenim paketima koje želimo da referenciramo
- Svaki učitani paket, može se nazvati proizvoljno, po konvencijama imenovanja svih ostalih identifikatora, a zatim koristiti pomoću datog naziva

```
import pkg.mymodule as mod  
mod.myfunction( )
```

Ugrađeni paketi

<https://docs.python.org/3/library/index.html>


- Python nakon instalacije, već sadrži mnoštvo paketa
- Ovi paketi zajedno čine standardnu Python biblioteku (Python Standard Library)

```
from urllib import request
import ssl, json

ctx = ssl._create_unverified_context()
res = request.urlopen("https://api.discogs.com/releases/249504", context=ctx)

data = json.loads(res.read())

for vid in data["videos"]:
    print(vid["title"])
    print(">>>", vid["uri"])
```

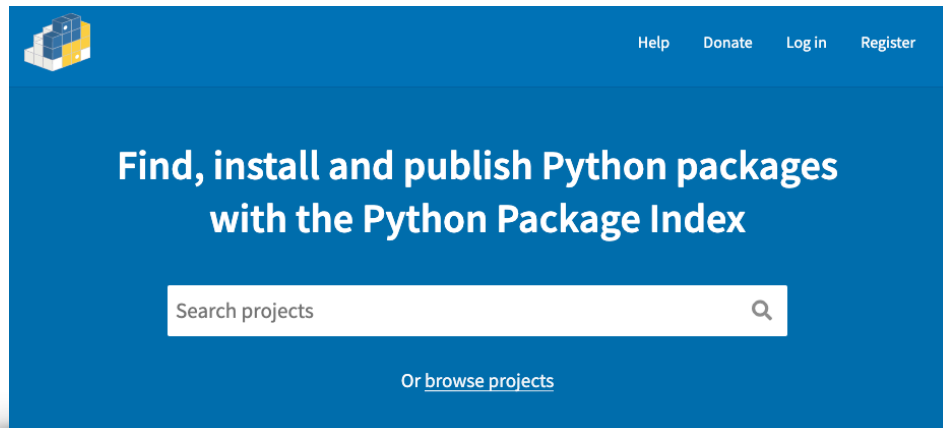


```
Rick Astley - Never Gonna Give You Up (Video)
>>> https://www.youtube.com/watch?v=dQw4w9WgXcQ
Never Gonna Give You Up (Escape From Newton Mix) - Rick Astley
>>> https://www.youtube.com/watch?v=SjlIuoMFPww
Never Gonna Give You Up (Escape To New York Mix) - Rick Astley
>>> https://www.youtube.com/watch?v=_cf3fxUDZvM
```

Python Package Index

<https://pypi.org/>

- Mnoštvo Python biblioteka dostupno je na globalnom repozitorijumu **Python Package Index**
- Biblioteke je moguće preuzimati ručno ili pomoću menadžera paketa **PIP**
- Menadžer paketa **PIP** (Pip Installs Python), često se distribuira uz Python interpreter



PIP - Python Installs Python

- Program PIP omogućava upravljanje Python paketima
- Najčešće se koristi za preuzimanje paketa sa repozitorijuma Pypi
- Takođe se može koristiti za kreiranje virtualnih okruženja, distribuciju sopstvenih biblioteka na Pypi repozitorijum i slično
- Za preuzimanje Python paketa pomoću PIP-a:

pip install PACKAGE

- Za brisanje paketa:

pip uninstall PACKAGE

Kreiranje virtualnog okruženja

- Da se paketi ne bi učitali globalno, mogu se koristiti virtualna okruženja
- Virtualna okruženja izoluju Python okruženja i omogućavaju njihovu lako i brzu izmenu

Kreiranje okruženja —————> `python -m venv myenv`

Aktivacija okruženja —————> `myenv/bin/activate`

Deaktivacija okruženja —————> `deactivate`

Zadatak - Turistička agencija

- Kreirati aplikaciju za potrebe turističke agencije
- Aplikacija po otvaranju prikazuje meni sa opcijama: **Destinacije, Aranžmani, Bukiranja, Pretraga, Statistika i Izlaz**
- Opcija **destinacije**, startuje podmeni za manipulaciju destinacijama. Ovaj podmeni ima opcije za prikaz svih destinacija, unos, ažuriranje ili brisanje destinacije.
- Korisnik unosi destinacije po potrebi. Svaka destinacija sadrži informacije: naziv i državu (iz predefinisane, hard kodirane liste država)
- Opcija **aranžmani**, otvara podmeni za manipulaciju aranžmanima. Stavke menija podrazumevaju opcije za unos, ažuriranje i brisanje aranžmana.
- Aranžmani sadrže podatke: destinacija, cena, datum početka i datum završetka (datum je tip string, sa sledećim formatom: gggg.mm.dd (2020.05.20)), broj slobodnih mesta
- Opcija **bukiranje** otvara podmeni sa stavkama za unos, ažuriranje i brisanje bukiranja.
- Bukiranje je lista svih bukiranja. Svako bukiranje sadrži: username klijenta, aranžman, broj bukiranih mesta i da li je aranžman plaćen
- **Pretraga** aktivira opcije za pretragu bukiranja. Opcije pretrage su: bukiranja jednog korisnika (bira se username klijenta), bukiranja za period (unos se početni i krajnji datum perioda)
- Startovanjem opcije **Statistika**, prikazuju se ukupni podaci sistema: Ukupan broj aranžmana, ukupan broj destinacija, ukupan broj bukiranja, broj preostalih aranžmana po destinacijama, suma plaćenih aranžmana, ukupan dug (suma neplaćenih aranžmana)

