Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования «Пермский национальный исследовательский политехнический университет» (ПНИПУ)

Кафедра вычислительной математики и механики

**Лабораторная работа № 3**

**по дисциплине: «Семантические сети»**

Выполнил

студент группы ИСТ-19-2б

Репин М.Е.

Проверил

ассистент кафедры ВММБ

Нетбай Г.В.

Пермь, 2022

# Семантическая сеть

Семантическая сеть – сеть, в которой элементы связаны между собой определённым типом связей. В такой сети у объекта нет полей, в которую заносится информация, вместо этого связь описывает, как взаимодействуют объекты между собой.

# Задача

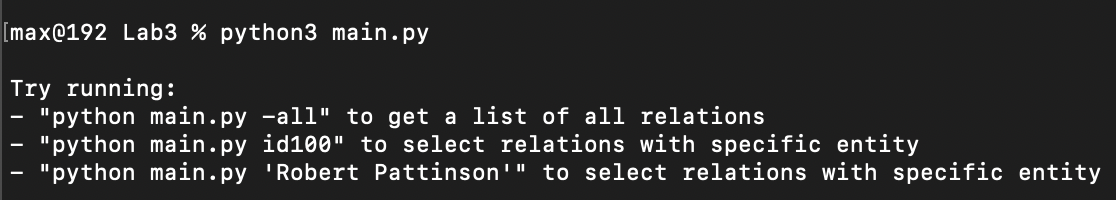
Написать программу, которая выводит все связи, в которых участвует объект. Список объектов хранится в .json файле, ссылка на объект осуществляется с помощью id. В отдельном .json файле хранится список связей – кто, как связан и с кем.

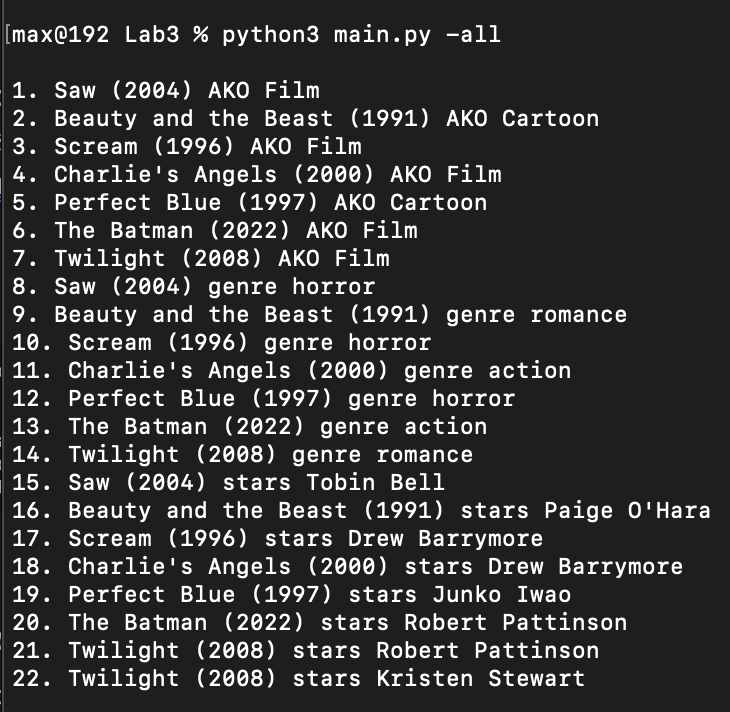
# Особенность программы

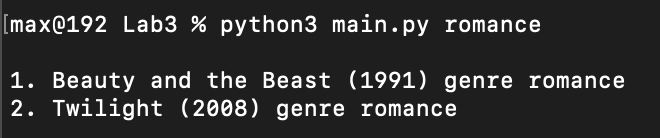
Программа запускается через консоль и принимает следующие параметры:

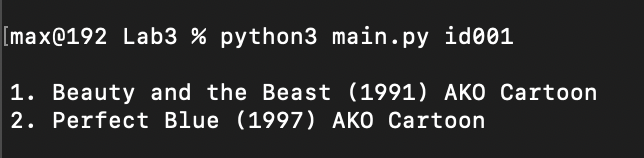
|  |  |
| --- | --- |
| Параметр | Особенность |
| -all | Выводит список всех связей в системе |
| id\*\*\* | Выводит список связей, в которых участвует сущность с указанным id |
| ‘Сущность’ | Выводит список связей с указанной сущность. Если в названии сущности есть скобки или пробелы, необходимо обрамлять в кавычки |

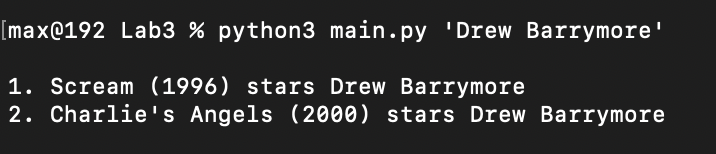
# Пример запуска программы main.py



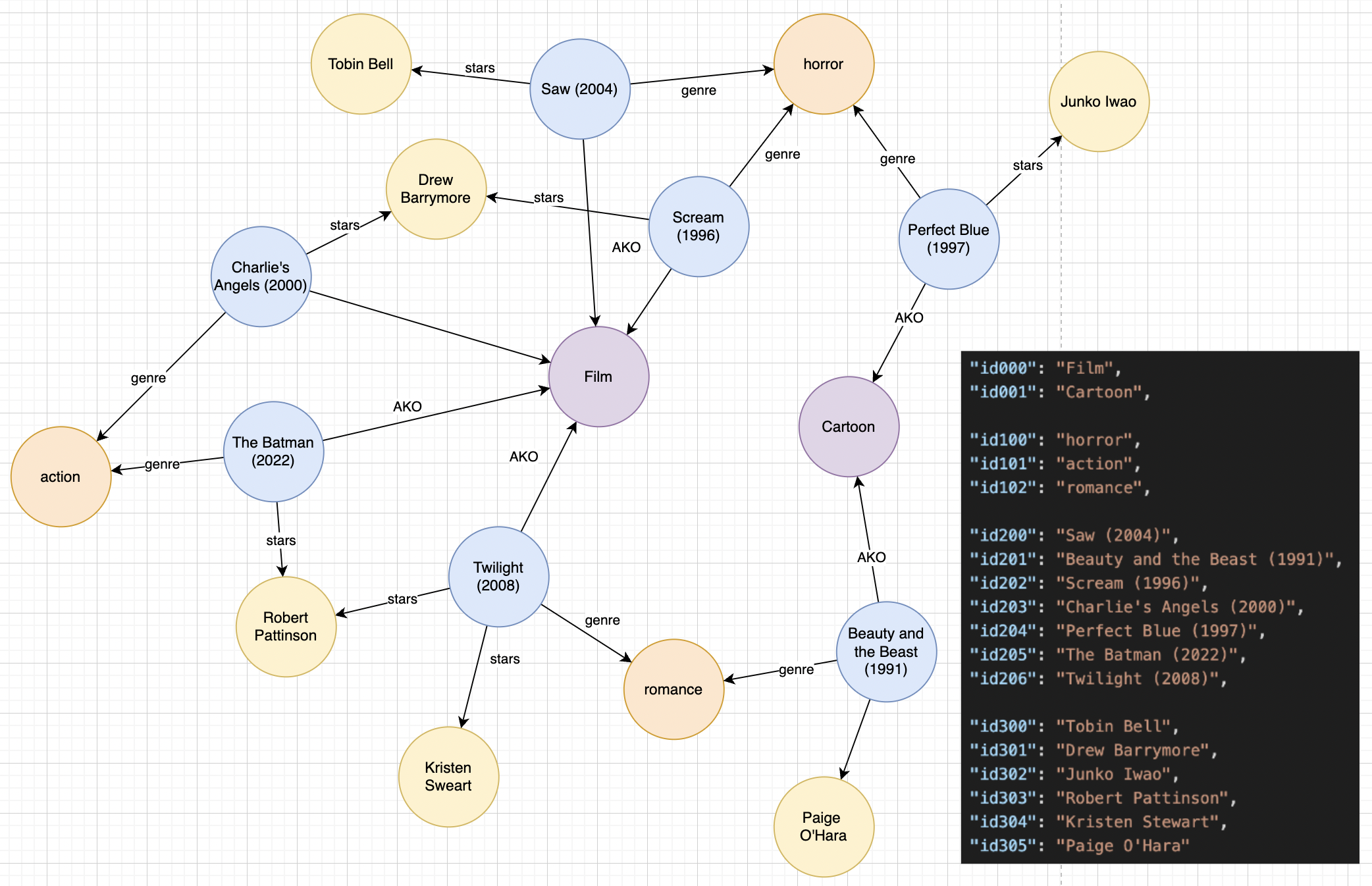








# Семантическая связь



# Листинг файла entities.json

{

"id000": "Film", "id001": "Cartoon",

"id100": "horror", "id101": "action", id102": "romance",

"id200": "Saw (2004)", "id201": "Beauty and the Beast (1991)",

"id202": "Scream (1996)", "id203": "Charlie's Angels (2000)",

"id204": "Perfect Blue (1997)", "id205": "The Batman (2022)",

"id206": "Twilight (2008)",

"id300": "Tobin Bell",

"id301": "Drew Barrymore",

"id302": "Junko Iwao",

"id303": "Robert Pattinson",

"id304": "Kristen Stewart",

"id305": "Paige O'Hara"

}

# Листинг файла relations.json

[

["id200", "AKO", "id000"],

["id201", "AKO", "id001"],

["id202", "AKO", "id000"],

["id203", "AKO", "id000"],

["id204", "AKO", "id001"],

["id205", "AKO", "id000"],

["id206", "AKO", "id000"],

["id200", "genre", "id100"],

["id201", "genre", "id102"],

["id202", "genre", "id100"],

["id203", "genre", "id101"],

["id204", "genre", "id100"],

["id205", "genre", "id101"],

["id206", "genre", "id102"],

["id200", "stars", "id300"],

["id201", "stars", "id305"],

["id202", "stars", "id301"],

["id203", "stars", "id301"],

["id204", "stars", "id302"],

["id205", "stars", "id303"],

["id206", "stars", "id303"],

["id206", "stars", "id304"]

]

# Листинг кода SemanticNetwork

import json

class SemanticNetwork:

def \_\_init\_\_(self, fileNameEntities: str, fileNameRelations: str) -> None:

self.Entities = self.ReadJsonFileEntities(fileNameEntities)

self.Relations = self.ReadJsonFileRelations(fileNameRelations)

def ReadJsonFileEntities(self, jsonFileName: str) -> dict[str, str]:

entities = {}

with open(jsonFileName, "r", encoding='utf-8') as read\_entities:

raw\_data = json.load(read\_entities)

for key, value in raw\_data.items():

entities[key] = value

return entities

def ReadJsonFileRelations(self, jsonFileName: str) -> list[list[str]]:

relations = []

with open(jsonFileName, "r", encoding='utf-8') as read\_relations:

raw\_data = json.load(read\_relations)

for item in raw\_data:

relations += [item]

return relations

def ListRelationsWith(self, entity: str) -> None:

"""

Lists relations where entity take part in. \n

Parameter \"entity\" can be an id or a value of an entity.

"""

if self.Entities.get(entity):

id = entity

else:

id = self.\_GetEntityIdByValue(entity)

# print(f"{id} = {entity}\n") # debug, checking if id is correct

if id == None:

print("Entity with such id or name doesn't exist")

return

count = 1

for item in self.Relations:

if id in (item[0], item[2]):

print(f"{count}. {self.Entities[item[0]]} {item[1]} {self.Entities[item[2]]}")

count += 1

def ListAll(self) -> None:

count = 1

for item in self.Relations:

print(f"{count}. {self.Entities[item[0]]} {item[1]} {self.Entities[item[2]]}")

count += 1

def \_GetEntityIdByValue(self, name: str) -> str | None:

""" Returns enitity's id or None. """

for id, value in self.Entities.items():

if value == name:

return id

return None

def PrintHelp(self) -> None:

print("""Try running:

- \"python main.py -all\" to get a list of all relations

- \"python main.py id100\" to select relations with specific entity

- \"python main.py 'Robert Pattinson'\" to select relations with specific entity

""")

# Листинг кода main.py

import pathlib

import sys

from SemanticNetwork import SemanticNetwork

folder = str(pathlib.Path(\_\_file\_\_).parent.resolve())

jsonFileNameEntities = folder + "/entities.json"

jsonFileNameRelations = folder + "/relations.json"

m = SemanticNetwork(jsonFileNameEntities, jsonFileNameRelations)

print() # first blank line

if len(sys.argv) == 1:

m.PrintHelp()

exit()

if sys.argv[1] == "-all":

m.ListAll()

else:

m.ListRelationsWith(sys.argv[1])

print() # last blank line before exit