Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «ИНЖЕНЕРНЫЙ БИЗНЕС И МЕНЕДЖМЕНТ»

КАФЕДРА «ПРОМЫШЛЕННАЯ ЛОГИСТИКА» (ИБМ-3)

Рубежный Контроль №1

«Парадигмы и конструкции языков программирования»

38.03.05 «Бизнес-информатика» (бакалавриат)

Студент ИБМ3-34Б _____ Сергеева М. А.
(Подпись, дата)

2024 г.

1. Введение

В данном отчете представлена разработка программы на языке Python, которая решает задачи, связанные с обработкой данных в предметной области, отражающей связи между языками программирования и синтаксическими конструкциями. В процессе работы были созданы классы, реализующие связи "один-ко-многим" и "многие-ко-многим", а также выполнены запросы с использованием функциональных возможностей языка Python.

2. Описание предметной области и классов

В качестве примера была выбрана область "Языки программирования и их синтаксические конструкции". В рамках этой области были реализованы три класса:

- **ProgrammingLanguage** — класс, описывающий язык программирования.
  - Поля: `id` (идентификатор языка), `name` (название языка).
- **SyntaxConstruction** — класс, описывающий синтаксическую конструкцию языка программирования.
  - Поля: `id` (идентификатор конструкции), `name` (название конструкции), `language_id` (связь с языком программирования, одно значение может быть связано с несколькими конструкциями).
- **LanguageSyntax** — класс, реализующий связь "многие-ко-многим" между языками программирования и их синтаксическими конструкциями.
  - Поля: `syntax_id` (идентификатор синтаксической конструкции), `language_id` (идентификатор языка программирования).

3. Тестовые данные

Для тестирования программы были созданы следующие данные:

- Языки программирования:
  - Python
  - Java
  - C++
- Синтаксические конструкции:
  - for loop (Python)
  - if-else (Python)
  - while loop (Java)
  - switch-case (C++)
  - try-catch (Java)
- Связь между языками и синтаксическими конструкциями реализована с помощью объектов класса `LanguageSyntax`.

4. Реализация программы

Программа была разделена на несколько частей:

- **Создание объектов классов** с тестовыми данными.
- **Запросы** для извлечения и сортировки данных в соответствии с условиями задачи.

5. Запросы

Запрос 1: Список всех связанных синтаксических конструкций и языков, отсортированных по языкам

Для этого запроса были использованы языки программирования, отсортированные по алфавиту, и для каждого языка был сформирован список синтаксических конструкций, связанных с этим языком.

```
# Сортировка языков по названию
sorted_languages = sorted(languages, key=lambda lang: lang.name)

# Для каждого языка выводим все его синтаксические конструкции
for lang in sorted_languages:
    constructs = [syn.name for syn in syntax_constructions if syn.language_id
== lang.id]
    print(f"{lang.name}: {', '.join(constructs)}")
```

Результат:

```
C++: switch-case
Java: while loop, try-catch
Python: if-else, for loop
```

Запрос 2: Список языков с количеством конструкций

Здесь мы подсчитываем количество синтаксических конструкций для каждого языка и выводим их, отсортированные по количеству конструкций в порядке убывания.

```
# Подсчет количества конструкций для каждого языка
language_counts = {
    lang.name: sum(1 for syn in syntax_constructions if syn.language_id ==
lang.id)
    for lang in languages
}

# Сортировка языков по количеству конструкций
print(sorted(language_counts.items(), key=lambda x: x[1], reverse=True))
```

Результат:

```
 [('Java', 2), ('Python', 2), ('C++', 1)]
```

Запрос 3: Языки, содержащие слово "язык" и их конструкции

В этом запросе мы фильтруем языки, название которых содержит слово "язык", и выводим все их синтаксические конструкции.

```
# Фильтрация языков по слову "язык" в названии
filtered_languages = [lang for lang in languages if "язык" in
lang.name.lower()]

# Вывод синтаксических конструкций для отфильтрованных языков
for lang in filtered_languages:
    constructs = [syn.name for syn in syntax_constructions if syn.language_id
== lang.id]
    print(f"{lang.name}: {', '.join(constructs)}")
```

6. Заключение

Программа решает задачу, описанную в условии, с использованием функциональных возможностей Python, таких как сортировка, фильтрация, и использование генераторов списков. В ходе выполнения работы были созданы и протестированы классы, а также выполнены все необходимые запросы.

```
class SyntaxConstruction:
    def __init__(self, id, name, language_id):
        self.id = id
        self.name = name
        self.language_id = language_id

class ProgrammingLanguage:
    def __init__(self, id, name):
        self.id = id
        self.name = name

class LanguageSyntax:
    def __init__(self, syntax_id, language_id):
        self.syntax_id = syntax_id
        self.language_id = language_id

# Создание тестовых данных
languages = [
    ProgrammingLanguage(1, "Python"),
    ProgrammingLanguage(2, "Java"),
    ProgrammingLanguage(3, "C++")
]

syntax_constructions = [
    SyntaxConstruction(1, "if-else", 1),
    SyntaxConstruction(2, "for loop", 1),
    SyntaxConstruction(3, "while loop", 2),
    SyntaxConstruction(4, "switch-case", 3),
    SyntaxConstruction(5, "try-catch", 2)
]

language_syntax = [
    LanguageSyntax(1, 1), LanguageSyntax(2, 1),
    LanguageSyntax(3, 2), LanguageSyntax(4, 3),
    LanguageSyntax(5, 2)
]

# Запрос 1: Список всех связанных синтаксических конструкций и языков, отсортированных
по языкам
sorted_languages = sorted(languages, key=lambda lang: lang.name)
for lang in sorted_languages:
    constructs = [syn.name for syn in syntax_constructions if syn.language_id ==
lang.id]
    print(f"{lang.name}: {', '.join(constructs)}")

# Запрос 2: Список языков с количеством конструкций
language_counts = {
    lang.name: sum(1 for syn in syntax_constructions if syn.language_id == lang.id)
    for lang in languages
}
print(sorted(language_counts.items(), key=lambda x: x[1], reverse=True))

# Запрос 3: Языки, содержащие "язык" и их конструкции
filtered_languages = [lang for lang in languages if "язык" in lang.name.lower()]
for lang in filtered_languages:
    constructs = [syn.name for syn in syntax_constructions if syn.language_id ==
lang.id]
    print(f"{lang.name}: {', '.join(constructs)}")
```