

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ  
ФЕДЕРАЦИИ**  
**Федеральное государственное автономное  
образовательное учреждение высшего образования  
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций**

**Институт цифрового развития**

**ОТЧЁТ**

**по лабораторной работе №2.4**

Дисциплина: «Основы кроссплатформенного программирования»

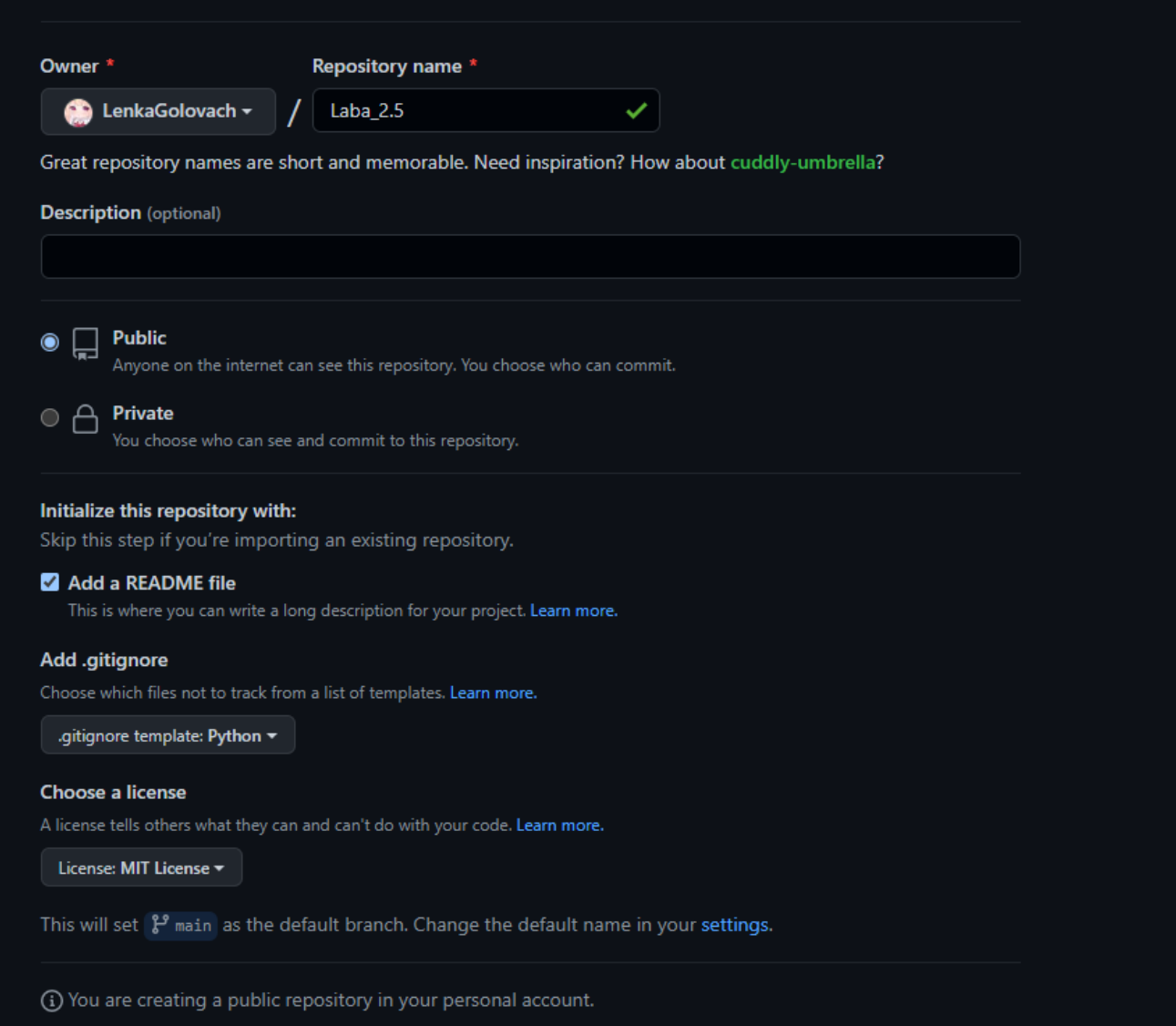
Тема: «Работа со списками в языке Python»

Выполнил: студент 1 курса  
группы ИВТ-б-о-21-1  
Криворот Владимир  
Геннадьевич

Ставрополь 2022

## Выполнение работы:

1. Создал репозиторий в GitHub «rep 2.5» в который добавил .gitignore, который дополнил правила для работы с IDE PyCharm с ЯП Python, выбрал лицензию MIT, клонировал его на лок. сервер и организовал в соответствии с моделью ветвления git-flow.



Owner \* **LenkaGolovach** / Repository name \* **Laba\_2.5** ✓

Great repository names are short and memorable. Need inspiration? How about **cuddly-umbrella**?

Description (optional)

☒ **Public**  
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**  
You choose who can see and commit to this repository.

**Initialize this repository with:**  
Skip this step if you're importing an existing repository.

☒ **Add a README file**  
This is where you can write a long description for your project. [Learn more.](#)

**Add .gitignore**  
Choose which files not to track from a list of templates. [Learn more.](#)

**.gitignore template:** Python

**Choose a license**  
A license tells others what they can and can't do with your code. [Learn more.](#)

**License:** MIT License

This will set **main** as the default branch. Change the default name in your [settings](#).

*i* You are creating a public repository in your personal account.

Рисунок 1.1 Создание репозитория

```
D:\git>git clone https://github.com/LenkaGolovach/Laba_2.5.git
Cloning into 'Laba_2.5'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
```

Рисунок 1.2 Клонирование репозитория

```

D:\git\Laba_2.5>git flow init

Which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [D:/git/Laba_2.5/.git/hooks]

```

Рисунок 1.3 Организация репозитория в соответствии с моделью ветвления git-flow

```

# PEP 582; used by e.g. github.com/David-OConnor/pyflow
__pypackages__

# Celery stuff
celerybeat-schedule
celerybeat.pid

# SageMath parsed files
*.sage.py

# Environments
.env
.venv
env/
venv/
ENV/
env.bak/
venv.bak/

# Spyder project settings
.spyderproject
.spyproject

# Rope project settings
.ropeproject

# mkdocs documentation
/site

# mypy
.mypy_cache/
.dmypy.json
dmypy.json

# Pyre type checker
.pyre/

```

ЛР.

Рисунок 1.4 Изменение .gitignore

2. Создал проект PyCharm в папке репозитория, проработал примеры

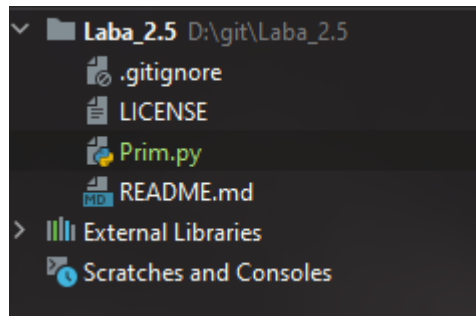


Рисунок 2.1 Создал проект в PyCharm

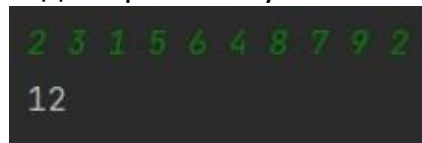
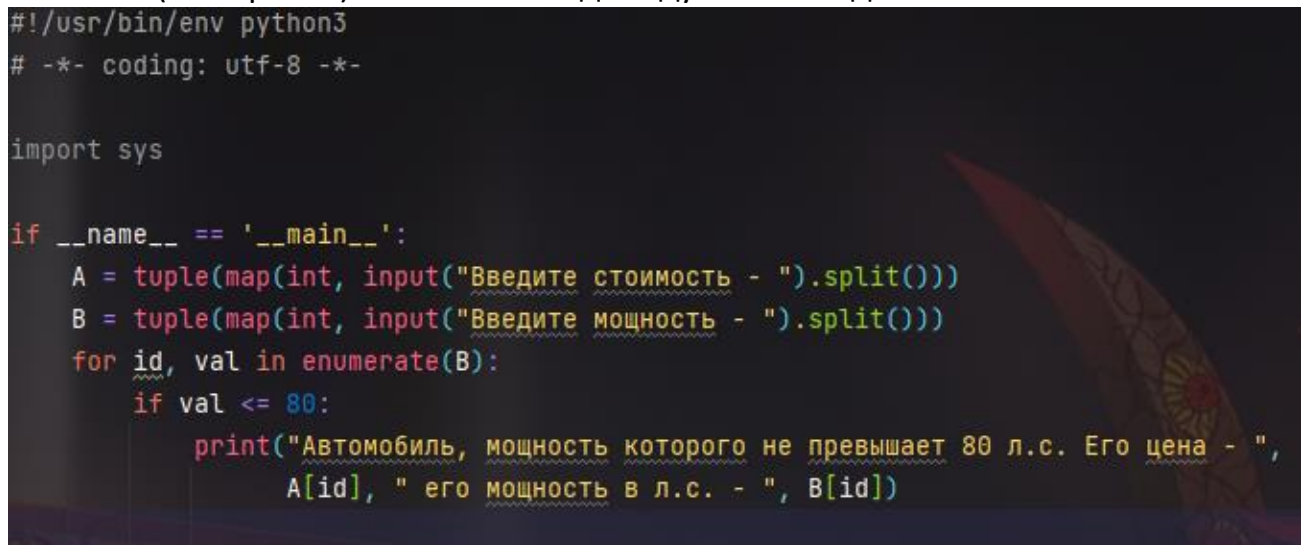


Рисунок 2.2 Рез-т выполнения программы

3. (16 вариант). Выполнил индивидуальное задание.



```

Введите стоимость - 1 2 3 4 5 6 7 8 9 10
Введите мощность - 120 10 500 300 79 20 90 60 40 29
Автомобиль, мощность которого не превышает 80 л.с. Его цена - 2 его мощность в л.с. - 10
Автомобиль, мощность которого не превышает 80 л.с. Его цена - 5 его мощность в л.с. - 79
Автомобиль, мощность которого не превышает 80 л.с. Его цена - 6 его мощность в л.с. - 20
Автомобиль, мощность которого не превышает 80 л.с. Его цена - 8 его мощность в л.с. - 60
Автомобиль, мощность которого не превышает 80 л.с. Его цена - 9 его мощность в л.с. - 40
Автомобиль, мощность которого не превышает 80 л.с. Его цена - 10 его мощность в л.с. - 29

```

Рисунок 3.1 Вывод программы индивидуального задания

4. Сделал коммит, выполнил слияние с веткой main, и запустил изменения в уд. репозиторий.

```

D:\git\Laba_2.5>git add .
D:\git\Laba_2.5>git commit -m "Выполненное задание"
[develop 1170b67] Выполненное задание
1 file changed, 12 insertions(+), 11 deletions(-)
rewrite Ind.py (62%)

```

Рисунок 4.1 коммит изменений и переход на ветку main

```

D:\git\Laba_2.5>git merge develop
Updating 1e35626..1170b67
Fast-forward
 .gitignore | 3 ++-
 Ind.py     | 12 ++++++++
 Prim.py    | 20 ++++++++
3 files changed, 34 insertions(+), 1 deletion(-)
create mode 100644 Ind.py
create mode 100644 Prim.py

```

Рисунок 4.2 Слияние ветки main с develop

```
C:\Users\adamk\OneDrive\Рабочий стол\rep_2.5>git push
info: please complete authentication in your browser...
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (7/7), 1.10 KiB | 562.00 KiB/s, done.
Total 7 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/AdamKh/rep_2.5.git
307f1de..1a66e01 main -> main
```

Рисунок 4.3 Пуш изменений на удаленный сервер








	LenkaGolovach	Выполненное задание	1170b67 2 minutes ago	 4 commits
	.gitignore	ignore		23 hours ago
	Ind.py	Выполненное задание		2 minutes ago
	LICENSE	Initial commit		23 hours ago
	Prim.py	laba		1 hour ago
	README.md	Initial commit		23 hours ago

Рисунок 4.4 Изменения на удаленном сервере

### Контр. вопросы и ответы на них:

#### 1. Что такое кортежи в языке Python?

Кортеж (tuple) – это неизменяемая структура данных, которая по своему подобию очень похожа на список.

#### 2. Каково назначение кортежей в языке Python?

Существует несколько причин, по которым стоит использовать кортежи вместо списков. Одна из них – это обезопасить данные от случайного изменения. Если мы получили откуда-то массив данных, и у нас есть желание поработать с ним, но при этом непосредственно менять данные мы не собираемся, тогда, это как раз тот случай, когда кортежи придутся как нельзя кстати. Кортежи в памяти занимают меньший объем по сравнению со списками. Кортежи работают быстрее, чем списки

#### 3. Как осуществляется создание кортежей?

a = ()

`b = tuple()`

#### **4. Как осуществляется доступ к элементам кортежа?**

Доступ к элементам кортежа осуществляется также как к элементам списка – через указание индекса.

#### **5. Зачем нужна распаковка (деструктуризация) кортежа?**

Обращение по индексу, это не самый удобный способ работы с кортежами. Дело в том, что кортежи часто содержат значения разных типов, и помнить, по какому индексу что лежит — очень непросто.

#### **6. Какую роль играют кортежи в множественном присваивании?**

Используя множественное присваивание, можно провернуть интересный трюк: обмен значениями между двумя переменными.

#### **7. Как выбрать элементы кортежа с помощью среза?**

С помощью операции взятия среза можно получить другой кортеж. Общая форма операции взятия среза для кортежа следующая

`T2 = T1[i:j]`

здесь

- `T2` – новый кортеж, который получается из кортежа `T1`;
- `T1` – исходный кортеж, для которого происходит срез;
- `i, j` – соответственно нижняя и верхняя границы среза. Фактически

берутся ко вниманию элементы, лежащие на позициях `i, i+1, ..., j-1`. Значение `j` определяет позицию за последним элементом среза.

#### **8. Как выполняется конкатенация и повторение кортежей?**

Для кортежей можно выполнять операцию конкатенации, которая обозначается символом `+`.

`T3 = T1 + T2`

#### **9. Как выполняется обход элементов кортежа?**

Элементы кортежа можно последовательно просмотреть с помощью операторов цикла `while` или `for`.



**10. Как проверить принадлежность элемента кортежу?**

Проверка вхождения элемента в кортеж - оператор `in`.

**11. Какие методы работы с кортежами Вам известны?**

`index()`, `count()`.

**12. Допустимо ли использование функций агрегации таких как `len()`, `sum()` и т. д. при работе с кортежами?**

Доступно.

**13. Как создать кортеж с помощью спискового включения.**

Так же как и список.