Адаптация кодов для исполнения на Jupyter Lite

Jupyter Lite — облегчённая версия Jupyter, работающая прямо в браузере без серверной части. Это накладывает ряд ограничений на используемые библиотеки и работу с файлами, интернетом и внешними зависимостями. Вот как адаптировать практикум по информационной безопасности для Jupyter Lite:

1. Используйте только поддерживаемые библиотеки

- В Jupyter Lite предустановлены основные библиотеки: numpy, pandas, matplotlib, sklearn и некоторые другие.
- Установка новых библиотек возможна только для чисто-питоновских пакетов (те, что имеют wheel вида py3-none-any.whl). Библиотеки с C-расширениями (например, cryptography, yara, folium) установить нельзя[2].
- Для визуализации используйте matplotlib или plotly (если доступен).

2. Работа с файлами

- Доступ к файловой системе ограничен. Можно использовать только файлы, загруженные пользователем в сессию, либо работать с данными в памяти (например, через переменные или StringIO).
- Для примеров с файлами используйте встроенные данные или предложите загрузить файл через интерфейс Jupyter Lite.

3. Криптография

- Используйте стандартную библиотеку Python: модуль hashlib для хеширования, модуль base64 для кодирования.
- Пример: простое симметричное "шифрование" (XOR) и хеширование.

python

```
# Пример простого XOR-шифрования
def xor_encrypt_decrypt(message, key):
    return ''.join(chr(ord(c) ^ key) for c in message)

msg = "секрет"
key = 42
enc = xor_encrypt_decrypt(msg, key)
dec = xor_encrypt_decrypt(enc, key)
print("Зашифровано:", enc)
print("Расшифровано:", dec)

# Пример хеширования
import hashlib

text = "секретное сообщение"
hash_object = hashlib.sha256(text.encode())
print("SHA256:", hash_object.hexdigest())
```

4. Анализ данных

- Используйте pandas для анализа и matplotlib для визуализации.
- Пример поиска номеров карт в тексте:

python

```
import re
import pandas as pd

data = ["Платеж 1234-5678-9012-3456", "Нет карты здесь", "Карта: 4111 1111 1111 1111"]
pattern = r"\b(?:\d[ -]*?){13,16}\b"

df = pd.DataFrame(data, columns=["text"])
df["cards"] = df["text"].apply(lambda x: re.findall(pattern, x))
print(df)
```

5. Визуализация

python

```
import matplotlib.pyplot as plt

# Пример визуализации количества найденных карт
counts = [1, 0, 1]
plt.bar(range(len(counts)), counts)
plt.xlabel("Строка")
plt.ylabel("Количество карт")
plt.title("Обнаружение номеров карт")
plt.show()
```

6. Импорт локальных модулей

- В Jupyter Lite не получится импортировать модули с диска, как в обычном Jupyter (sys.path.insert, %load_ext autoreload)1. Всё должно быть в одной тетради или в загруженных пользователем файлах.

7. Пример структуры практикума

- Каждый блок: Markdown с теорией → Ячейка с кодом → Ячейка для самостоятельной работы → Ячейка с визуализацией → Выводы.

8. Что не будет работать

- Установка и использование библиотек, требующих компиляции или C-расширений (cryptography, yara, folium и др.)2.
- Долгие вычисления и большие файлы (>100 МБ).
- Доступ к интернету из кода.

Вывод:

Практикум должен использовать только стандартные и поддерживаемые библиотеки, работать с небольшими данными, а все решения и визуализации должны быть реализованы средствами, доступными в Jupyter Lite24. Для криптографии и анализа угроз используйте просты