Министерство науки и высшего образования РФ

|  |  |  |
| --- | --- | --- |
| Федеральное государственное автономное образовательное учреждение высшего образования  «СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ» | | |
| Институт космических и информационных технологий | | |
| Кафедра информационных систем | | |
| **ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №8** Классы и модули | | |
| Вариант 2 | | |
| Преподаватель |  | А.Г. Фельдман |
|  |  | *подпись, дата* |
| Студент | КИ22-06Б, 032215878 | Д.А. Безпалый |
|  | *номер группы, зачетной книжки* | *подпись, дата* |
|  | Красноярск 2024 | |

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc181363608)

[ОСНОВНАЯ ЧАСТЬ 4](#_Toc181363609)

[Задание 1. 4](#_Toc181363610)

[Задание 2. Работа с бинарными файлами 6](#_Toc181363611)

[ЗАКЛЮЧЕНИЕ 8](#_Toc181363612)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 9](#_Toc181363613)

# ВВЕДЕНИЕ

Цель практической работы: изучение технологии структурирования программы с использованием классов и модулей в Python.

Основная задача практической работы: решить 1 задачу по своему варианту и 2 задачи.

# ОСНОВНАЯ ЧАСТЬ

## Задание 1. Объектно-ориентированное программирование

Опишите класс ТРЕУГОЛЬНИК, заданный длинами сторон. Включите в описание класса методы, позволяющие вывести длины сторон треугольника на экран, рассчитать периметр, площадь, высоты треугольника и свойство, позволяющее установить, существует ли треугольник с данными длинами сторон.

Код, решивший задачу отображён на рисунке 1.

class Triangle:  
 def \_\_init\_\_(self, a, b, c):  
 self.a = a  
 self.b = b  
 self.c = c  
  
 def printSides(self):  
 print(f"Стороны равны: {self.a, self.b, self.c}")  
  
 def calculatePerimeter(self):  
 return self.a + self.b + self.c  
  
 def calculateSquare(self):  
 perimeter = self.calculatePerimeter()  
 halfPerimeter = perimeter / 2  
 halfPerimeterA = halfPerimeter - self.a  
 halfPerimeterB = halfPerimeter - self.b  
 halfPerimeterC = halfPerimeter - self.c  
 square = (halfPerimeter \* halfPerimeterA \* halfPerimeterB \* halfPerimeterC) \*\* 0.5  
 return square  
  
 def calculateHeights(self):  
 square = self.calculateSquare()  
 heightA = 2 \* square / self.a  
 heightB = 2 \* square / self.b  
 heightC = 2 \* square / self.c  
 return heightA, heightB, heightC  
  
 def isTriangleExist(self):  
 return self.a + self.b > self.c and self.a + self.c > self.b and self.b + self.c > self.a

Рисунок 1 – Программный код

Инициализация и вызов функций класса изображены на рисунке 2.

import sys  
import triangle  
  
a, b, c = map(int, input().split())  
  
trangl = triangle.Triangle(a, b, c)  
  
exist = trangl.isTriangleExist()  
if exist:  
 print("Треугольник существует")  
if not exist:  
 print("Треугольник не существует")  
 sys.exit()  
  
trangl.printSides()  
  
perimeter = trangl.calculatePerimeter()  
print(f"Периметр равен: {perimeter}")  
  
square = trangl.calculateSquare()  
print(f"Площадь равна: {square}")  
  
heights = trangl.calculateHeights()  
print(f"Высоты равны: {heights}")

Рисунок 2 – Программный код

## Задание 2. Подключение модулей

Разработаем скрипт для работы с курсами валют и разместим его в

отдельном модуле под названием Currency.py. Данные по курсам валют

можно брать из открытых источников, например: <https://www.cbr-xmldaily.ru/daily_json.js>

Модуль Currency.py изображён на рисунке 3.

import requests  
from datetime import datetime, timedelta  
  
  
class Rate:  
 def \_\_init\_\_(self, format='value'):  
 self.format = format  
 self.base\_url = 'https://www.cbr-xml-daily.ru'  
  
 def exchange\_rates(self):  
 try:  
 r = requests.get(f'{self.base\_url}/daily\_json.js')  
 return r.json()['Valute']  
 except:  
 return None  
  
 def exchange\_date\_rates(self, date):  
 try:  
 date\_str = date.strftime("%Y/%m/%d")  
 r = requests.get(f'{self.base\_url}/archive/{date\_str}/daily\_json.js')  
 return r.json()['Valute']  
 except:  
 return None  
  
 def make\_format(self, currency):  
 response = self.exchange\_rates()  
 if response and currency in response:  
 if self.format == 'full':  
 return response[currency]  
 elif self.format == 'value':  
 return response[currency]['Value']  
 elif self.format == 'name':  
 return f"{response[currency]['Name']} {response[currency]['Value']} руб."  
  
 return 'Error'  
  
 def eur(self, diff=False):  
 response = self.exchange\_rates()  
 if response and 'EUR' in response:  
 if diff:  
 return response['EUR']['Value'] - response['EUR']['Previous']  
 return self.make\_format('EUR')  
 return 'Error'  
  
 def usd(self, diff=False):  
 response = self.exchange\_rates()  
 if response and 'USD' in response:  
 if diff:  
 return response['USD']['Value'] - response['USD']['Previous']  
 return self.make\_format('USD')  
 return 'Error'  
  
 def get\_currency(self, code):  
 return self.make\_format(code)  
  
 def get\_max\_currency(self):  
 response = self.exchange\_rates()  
 if response:  
 max\_currency = max(response.items(), key=lambda x: x[1]['Value'])  
 return f"{max\_currency[1]['Name']} {max\_currency[1]['Value']} руб."  
 return 'Error'  
  
 def get\_min\_currency(self):  
 response = self.exchange\_rates()  
 if response:  
 min\_currency = min(response.items(), key=lambda x: x[1]['Value'])  
 return f"{min\_currency[1]['Name']} {min\_currency[1]['Value']} руб."  
 return 'Error'  
  
 def get\_usd\_history(self, weeks=1):  
 history = {}  
 current\_date = datetime.now()  
  
 for i in range(weeks \* 7):  
 date = current\_date - timedelta(days=i)  
 rates = self.exchange\_date\_rates(date)  
  
 if rates and 'USD' in rates:  
 history[date.strftime("%Y-%m-%d")] = rates['USD']['Value']  
 else:  
 history[date.strftime("%Y-%m-%d")] = 'Нет данных'  
  
 return history  
  
 def get\_currency\_history(self, code, weeks=1):  
 history = {}  
 current\_date = datetime.now()  
  
 for i in range(weeks \* 7):  
 date = current\_date - timedelta(days=i)  
 rates = self.exchange\_date\_rates(date)  
  
 if rates and code in rates:  
 history[date.strftime("%Y-%m-%d")] = rates[code]['Value']  
 else:  
 history[date.strftime("%Y-%m-%d")] = 'Нет данных'  
  
 return history

Рисунок 3 – Модуль Currency.py

from Currency import Rate  
  
rate\_value = Rate('value')  
rate\_full = Rate('full')  
rate\_name = Rate('name')  
  
print(rate\_value.usd())  
print(rate\_name.eur())  
  
print(rate\_value.usd(diff=True))  
  
print(rate\_name.get\_currency('GBP'))  
  
print(rate\_name.get\_max\_currency())  
print(rate\_name.get\_min\_currency())  
  
usd\_history = rate\_value.get\_usd\_history(weeks=2)  
for date, value in usd\_history.items():  
 print(f"Курс доллара на {date}: {value}")  
  
eur\_history = rate\_value.get\_currency\_history('EUR', weeks=1)  
for date, value in eur\_history.items():  
 print(f"Курс евро на {date}: {value}")

Рисунок 4 – Программный код

## Задание 3. Работа с текстовыми и бинарными файлами

Для своего класса, реализованного в 1 задании, выполните следующие

операции:

1. Создайте список из 3 объектов класса и сохраните данные в текстовый

файл (самостоятельно подумайте над тем, как разделить между собой

значения);

2. Добавьте в конец файла данные еще 2 объектов;

3. Выполните чтение данных из файла 3 способами: с помощью метода

readline(), метода read() или метода readlines();

4. С помощью модуля pickle:

4.1. Сохраните список объектов, созданный в п. 1, в бинарный файл;

4.2. Выполните чтение данных из файла, созданного в предыдущем

пункте;

4.3. Сохраните каждый объект по отдельности в новый файл;

4.4. Выполните чтение данных из файла, созданного в предыдущем

пункте. Предположим, что мы не знаем кол-во объектов, данные которых мы

храним в файле. Как можно выполнить чтение данных в этом случае?

5. С помощью модуля shelve сохраните данные в бинарный файл-хранилище. Выполните операции получения, обновления и удаления данных.

Код решивший задачу приведён на рисунке 5.

import pickle  
import shelve  
from triangle import Triangle  
  
triangles = [  
 Triangle(3, 4, 5),  
 Triangle(6, 8, 10),  
 Triangle(5, 12, 13)  
]  
  
with open('triangles.txt', 'w') as file:  
 for triangle in triangles:  
 file.write(f"{triangle.a},{triangle.b},{triangle.c}\n")  
  
new\_triangles = [  
 Triangle(7, 24, 25),  
 Triangle(8, 15, 17)  
]  
  
with open('triangles.txt', 'a') as file:  
 for triangle in new\_triangles:  
 file.write(f"{triangle.a},{triangle.b},{triangle.c}\n")  
  
with open('triangles.txt', 'r') as file:  
 while True:  
 line = file.readline()  
 if not line:  
 break  
 a, b, c = map(float, line.strip().split(','))  
 triangle = Triangle(a, b, c)  
 print(f"Стороны треугольника (readline): {triangle.a}, {triangle.b}, {triangle.c}")  
  
with open('triangles.txt', 'r') as file:  
 data = file.read()  
 lines = data.split('\n')  
 for line in lines:  
 if line:  
 a, b, c = map(float, line.strip().split(','))  
 triangle = Triangle(a, b, c)  
 print(f"Стороны треугольника (read): {triangle.a}, {triangle.b}, {triangle.c}")  
  
with open('triangles.txt', 'r') as file:  
 lines = file.readlines()  
 for line in lines:  
 a, b, c = map(float, line.strip().split(','))  
 triangle = Triangle(a, b, c)  
 print(f"Стороны треугольника (readlines): {triangle.a}, {triangle.b}, {triangle.c}")  
  
with open('triangles.pkl', 'wb') as file:  
 pickle.dump(triangles, file)  
  
with open('triangles.pkl', 'rb') as file:  
 loaded\_triangles = pickle.load(file)  
 for triangle in loaded\_triangles:  
 print(f"Стороны треугольника (pickle): {triangle.a}, {triangle.b}, {triangle.c}")  
  
with open('triangles\_separate.pkl', 'wb') as file:  
 for triangle in triangles:  
 pickle.dump(triangle, file)  
  
with open('triangles\_separate.pkl', 'rb') as file:  
 while True:  
 try:  
 triangle = pickle.load(file)  
 print(f"Стороны треугольника (pickle по отдельности): {triangle.a}, {triangle.b}, {triangle.c}")  
 except EOFError:  
 break  
  
with shelve.open('triangles.db') as db:  
 for i, triangle in enumerate(triangles):  
 db[f'triangle\_{i}'] = triangle  
  
with shelve.open('triangles.db') as db:  
 for key in db:  
 triangle = db[key]  
 print(f"Стороны треугольника (shelve): {triangle.a}, {triangle.b}, {triangle.c}")  
  
with shelve.open('triangles.db') as db:  
 db['triangle\_0'] = Triangle(4, 5, 6)  
  
  
with shelve.open('triangles.db') as db:  
 del db['triangle\_0']

Рисунок 5 – Программный код

# ЗАКЛЮЧЕНИЕ

Выполняя практическую работу, были получены навыки работы с технологиями структурирования программы с использованием классов и модулей на языке программирования Python.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. СТУ 7.5–07–2021. Стандарт университета «Система менеджмента качества. Общие требования к построению, изложению и оформлению документов учебной деятельности».
2. еКурсы – Система электронного обучения СФУ : Курс: Основы программирования на Python 2024 URL: [https://e.sfu-](https://e.sfu-kras.ru/course/view.php?id=38620&section-0) [kras.ru/course/view.php?id=38620#section-0](https://e.sfu-kras.ru/course/view.php?id=38620&section-0) (дата обращения 31.10.2024).