ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ РЕСПУБЛИКИ ХАКАСИЯ  
«ХАКАССКИЙ ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ»

Специальность 09.02.07

«Информационные системы и программирование»

ТЕМА: Практическая работа №3

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Студент гр. | ИС(ТП)-31 |  |  | | | |  |  | Соломатин А.Д. Шухраев К.Н. | | |
|  |  |  | *подпись* | | | |  |  | *Фамилия, И.О.* | | |
| Руководитель |  |  |  |  |  |  | | | |  | Брюханова И.Н. |
|  | *оценка* |  | *дата* |  |  | *подпись* | | | |  | *Фамилия, И.О.* |

Абакан, 2024

# Практическая работа №\_3\_:

**Цель работы:**

**Задание №1**

1. Заказчик требует реализовать API-приложение, предоставляющее

текущую погоду в городах республики Хакасия.

2. Приложение должно быть протестировано через консоль.

3. Приложение должно предоставлять погоду для следующих

населенных пунктов: Абакан, Сорск, Усть-Абакан, Таштып, Боград, Копьёво,

Абаза, Черногорск.

4. Заказчику нужна погода в градусах в трёх единицах измерения

погоды (Цельсий, Фаренгейт, Кельвин).

5. Приложение должно иметь функцию вывода погоды в градусах в

трёх различных единицах измерения и с корректными показателями погоды

по названию города.

− входные данные: название города (например, «Абакан»);

− выходные данные: список показателей погоды в трёх

различных единицах измерения (например, [30.0C, 86.0F,

303.15K])

6. Приложение должно иметь функцию фильтрации городов по

показателям: больше нуля или ноль, меньше нуля (только по Цельсию).

− входные данные: символы ‘+’ или ‘-‘;

− выходные данные: список городов в формате: [Город, Значение

погоды, Сокращенный формат единицы измерения] (например:

`[[Абакан, 28, C], [Боград, 65, F]]`)

7. Приложение должно иметь функцию сортировки по температуре

по Цельсию как по убыванию, так и по возрастанию.

− входные данные: команда о вызове сортировки;

− выходные данные: отсортированный список городов.

Сообщения от менеджера проекта

Ваша компания решила отказаться от циклов в программном коде и

использует только встроенные функции, такие как `filter()` и `map()`.

Для единиц измерения погоды используйте перечисления (enum) со

значениями: Celsius, Fahrenheit, Kelvin.

В качестве заглушки реальной погоды – используйте генератор

случайных чисел (например randint) для назначения погоды по Цельсию.

Значения по Фаренгейту и Кельвина высчитайте с помощью формулы.

Значения погоды должны иметь формат с одним числом после запятой

(например, 36.3C). Не допускается значения по типу: 38.35323434534F.

Для сортировки используйте функцию `sorted()` как для сортировки по

возрастанию, так и по убыванию. Либо встроенную функцию `sort()`,

доступную через список.

2.Листинг программы:

from enum import Enum

from random import randint

# Перечисление единиц измерения температуры

class TemperatureUnit(Enum):

Celsius = "C"

Fahrenheit = "F"

Kelvin = "K"

# Города Республики Хакасия

CITIES = ["Абакан", "Сорск", "Усть-Абакан", "Таштып", "Боград", "Копьёво", "Абаза", "Черногорск"]

# Генерация случайной температуры в градусах Цельсия

def generate\_temperature():

return round(randint(-30, 40) + randint(0, 9) / 10, 1)

# Преобразование температуры

def convert\_temperature(temp\_celsius, unit):

if unit == TemperatureUnit.Fahrenheit:

return round(temp\_celsius \* 9/5 + 32, 1)

elif unit == TemperatureUnit.Kelvin:

return round(temp\_celsius + 273.15, 1)

return temp\_celsius

# Генерация данных по всем городам

def get\_weather\_data():

return {city: generate\_temperature() for city in CITIES}

# Получение температуры для города в трёх единицах измерения

def get\_city\_weather(city, weather\_data):

if city not in weather\_data:

return f"Город {city} не найден."

temp\_celsius = weather\_data[city]

return [

f"{temp\_celsius}{TemperatureUnit.Celsius.value}",

f"{convert\_temperature(temp\_celsius, TemperatureUnit.Fahrenheit)}{TemperatureUnit.Fahrenheit.value}",

f"{convert\_temperature(temp\_celsius, TemperatureUnit.Kelvin)}{TemperatureUnit.Kelvin.value}"

]

# Фильтрация городов по температуре

def filter\_cities(symbol, weather\_data):

if symbol == '+':

filtered = filter(lambda item: item[1] >= 0, weather\_data.items())

elif symbol == '-':

filtered = filter(lambda item: item[1] < 0, weather\_data.items())

else:

return "Некорректный символ фильтрации."

return [[city, temp, TemperatureUnit.Celsius.value] for city, temp in filtered]

# Сортировка городов по температуре

def sort\_cities(order, weather\_data):

reverse = order == "desc"

return sorted(weather\_data.items(), key=lambda item: item[1], reverse=reverse)

# Тестирование через консоль

def main():

weather\_data = get\_weather\_data()

print("Добро пожаловать в API-приложение для погоды в Хакасии!")

while True:

print("\nВыберите действие:")

print("1: Узнать погоду в городе")

print("2: Фильтрация городов по температуре")

print("3: Сортировка городов по температуре")

print("0: Выход")

choice = input("Введите номер действия: ")

if choice == "1":

city = input("Введите название города: ")

print(get\_city\_weather(city, weather\_data))

elif choice == "2":

symbol = input("Введите фильтр (‘+’ или ‘-’): ")

print(filter\_cities(symbol, weather\_data))

elif choice == "3":

order = input("Введите порядок сортировки (‘asc’ или ‘desc’): ")

print(sort\_cities(order, weather\_data))

elif choice == "0":

print("Выход из приложения.")

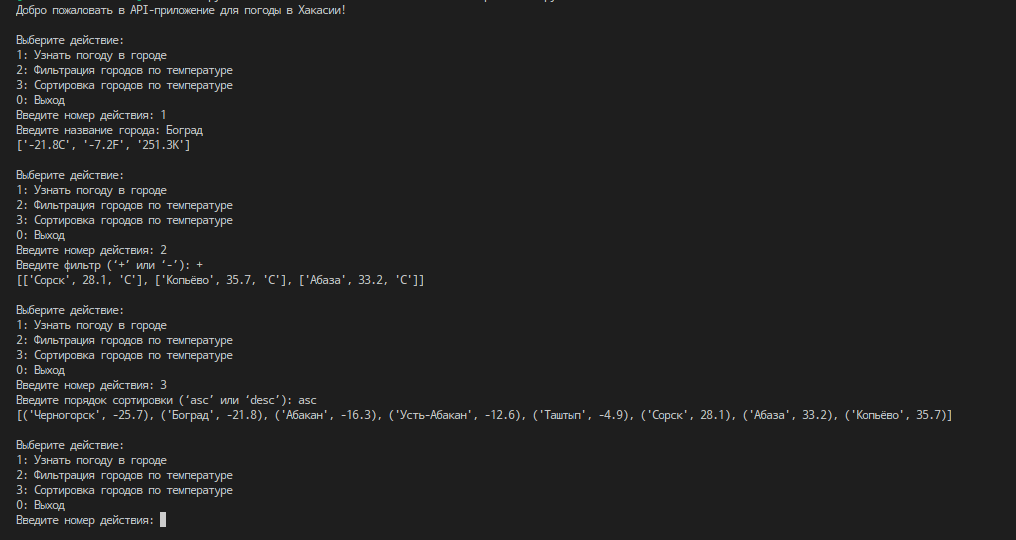
break

else:

print("Некорректный выбор.")

main()

3.Результаты

****