МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВВГУ»)

ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И АНАЛИЗА ДАННЫХ
КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СИСТЕМ

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №6

по дисциплине
«Информатика и программирование»

Студент
гр. БИН-25-2 _____ К.Ф. Кучерчук

Ассистент
преподавателя _____ М.В. Водяницкий

Владивосток 2025

Задание

Выполнить задания и оформить отчет по стандартам ВВГУ.

Основные задания

Задание 1

Написать функцию, которая конвертирует время из одной величины в другую.

На вход подается:

- число (величина времени)
- исходная единица измерения
- единица измерения, в которую нужно перевести

Функция должна вернуть конвертированное значение.

Примеры (формат ввода/вывода можно выбрать свой, если нет строгих требований):

Вход	Выход
`4h m`	`240m`
`30m h`	`0.5h`
`12s h`	`0.03h`

Задание 2

Пользователь делает вклад в банке в размере `a` рублей сроком на `n` лет. Процент по вкладу **зависит от суммы и срока**.

Зависимость от суммы:

- каждые 10 000 рублей увеличивают ставку на 0.3%
- но суммарное увеличение не может превышать 5%
- минимальный вклад — 30 000 рублей

Зависимость от срока:

- первые 3 года — 3%
- от 4 до 6 лет — 5%
- более 6 лет — 2%

Необходимо написать функцию, которая рассчитывает прибыль пользователя без учета первоначально вложенной суммы. Используется сложный процент: каждый год процент начисляется на текущую сумму вклада.

На вход подаются: сумма вклада и количество лет. Результат: сумма прибыли (не весь вклад, а только заработанные проценты).

Примеры:

Вход	Выход
30000 3	3648.67
100000 5	38920.10
200000 8	183925.42

Задание 3

Написать функцию для вывода всех простых чисел в заданном диапазоне. Нужно учитывать некорректные данные (например, начало больше конца или диапазон без простых чисел).

На вход подаются два числа: начало и конец диапазона (включительно). На выходе — список всех простых чисел или сообщение об ошибке.

Примеры:

Вход	Выход
1 10	2 3 5 7
15 120	17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 1
0 1	Error!

Задание 4

Реализовать функцию сложения двух матриц.

При сложении двух матриц получается новая матрица того же размера, где каждый элемент — это сумма элементов с тем же индексом из двух исходных матриц.

Ограничения:

- складывать можно только матрицы одинакового размера
- размер матрицы должен быть строго больше 2 (например, 3×3, 4×4 и т.д.)
- при нарушении условий нужно вывести сообщение об ошибке

На вход подаются:

1. размер матрицы `n` (для квадратной матрицы `n × n`)
2. элементы первой матрицы (по строкам, через пробел)
3. элементы второй матрицы в таком же формате

Результат — новая матрица (в том же формате), либо сообщение об ошибке.

Пример (один из возможных вариантов формата):

Вход:

```
2
```

```
2 5
5 3
5 2
4 1

```

Выход:

```
7 7
9 4

```

Пример с ошибкой (слишком маленький размер, неправильный ввод и т.п.):

Вход:

```
1
4
5

```

Выход:

```
Error!

```

Задание 5

Написать функцию, которая определяет, является ли строка палиндромом.

Палиндром — это строка, которая читается одинаково слева направо и справа налево (обычно без учета пробелов, регистра и знаков препинания — эти правила нужно явно задать в своей реализации).

На вход подается строка. На выходе:

- `Да`, если это палиндром
- `Нет`, если это не палиндром

Примеры:

Вход	Выход
`А роза упала на лапу Азора`	`Да`
`Borrow or rob`	`Да`
`Алфавитный порядок`	`Нет`

Содержание

1 Выполнение работы

1.1 Задание 1

Программа реализует функцию преобразования единиц времени (секунды, минуты, часы, дни). На вход подаётся строка вида «12s h», где число и исходная единица отделяются от целевой единицы пробелом. Функция использует словарь для хранения коэффициентов перевода в секунды.

```
#task_1

def time_convert(data: str):
    table = {
        "s" : 1,
        "m" : 60,
        "h" : 3600,
        "d" : 3600*24
    }

    source, res = data.split()
    num, char = int(''.join(source)[0:-1]), ''.join(source)[-1]
    result = num * table[char]/ table[res]
    return str(result.__round__(3)) + res


us_input = "12s h"
print(time_convert(us_input))
```

Рисунок 1 – Листинг программы для задания 1

Внутри функции строка разбивается на исходное значение и целевую единицу. Числовая часть извлекается путём удаления последнего символа (обозначения единицы), а коэффициенты из словаря используются для перевода в целевую систему. Результат округляется до трёх знаков после запятой и возвращается в виде строки с суффиксом целевой единицы.

1.2 Задание 2

Функция рассчитывает прибыль по вкладу с учётом сложного процента. Она учитывает минимальную сумму вклада (30 000 руб.), бонусную ставку (до 5% в зависимости от суммы) и основную ставку (в зависимости от срока). Прибыль вычисляется как разница между итоговой суммой и первоначальным вкладом.

```
def calculate_vklad(summa:int,srok:int) -> float:
    if summa <30000:
        return 'Минимальная сумма вклада 30 000 рублей'
    if srok < 1:
        return "Минимальный срок вклада 1 месяц"

    bonus = min(summa // 10000 * 0.003, 0.05)
    stavka = 0.03 if srok <=3 else 0.05 if 4<srok<=7 else
    0.02

    result = summa * (1+stavka+bonus) ** srok - summa
    return f'Прибыль : {result.round(2)}'



x, y = [int(i) for i in input('Введите сумму и срок : ').
    split()]
print(calculate_vklad(x, y))
```

Рисунок 2 – Листинг программы для задания 2

Сначала проверяются ограничения: сумма не менее 30 000, срок — не менее 1 года. Затем вычисляется бонус: 0.3% за каждые 10 000 руб., но не более 5%. Основная ставка определяется по диапазону срока. Формула сложного процента применяется единожды: $S = P \cdot (1 + r)^n - P$. Однако в текущей реализации допущена ошибка: условие ‘4 < srok <= 7‘ не охватывает 4 года. Это следует исправить на ‘srok <= 3‘, ‘4 <= srok <= 6‘, иначе ‘else‘.

1.3 Задание 3

Функция принимает два целых числа — границы диапазона — и возвращает все простые числа в нём. Если простых чисел нет, возвращается строка «Error!». Диапазон всегда начинается с `max(2, a)`, так как числа меньше 2 не могут быть простыми.

```
def primes(a,b):
    primes_list = []
    for num in range(max(2, a), b + 1):
        if num > 1:
            for i in range (2, int(num**0.5) + 1):
                if num % i == 0:
                    break
            else:
                primes_list.append(num)

    return primes_list if primes_list else "Error!"

a = int(input("Начало диапазона:"))
b = int(input("Конец диапазона:"))
print(*primes(a, b))

primes(a,b)
```

Рисунок 3 – Листинг программы для задания 3

Для каждого числа от `max(2, a)` до `b` проверяется делимость на все целые от 2 до $\sqrt{n}$. Если делитель найден — число составное. Иначе — простое и добавляется в список. Это классический метод проверки простоты «перебором до корня».

1.4 Задание 4

Программа складывает две квадратные матрицы одинакового размера. Перед началом проверяется, что размер матрицы строго больше 2. Если нет — выводится «Error!». При любых ошибках ввода (неверное число элементов, нечисловые значения и т.п.) также выводится сообщение об ошибке.

```
def add_matrices():
    try:
        n = int(input("Введите размер матрицы: "))
        if n <= 2:
            print("Error!")
            return

        print("Введите первую матрицу:")
        matrix1 = [list(map(int, input().split())) for _ in range(n)]

        print("Введите вторую матрицу:")
        matrix2 = [list(map(int, input().split())) for _ in range(n)]

        print("Результат:")
        for i in range(n):
            print(' '.join(str(matrix1[i][j] + matrix2[i][j]) for j in range(n)))

    except:
        print("Error!")

add_matrices()
```

Рисунок 4 – Листинг программы для задания 4

Чтение матриц осуществляется построчно с помощью генератора списков и `map(int, input().split())`. Сложение выполняется поэлементно в цикле. Обработка ошибок реализована через универсальный блок `except`, что соответствует требованиям задания.

1.5 Задание 5

Программа определяет, является ли введённая строка палиндромом. Для этого удаляются все неалфавитно-цифровые символы, строка приводится к нижнему регистру, и сравнивается с обратной копией.

```
s = input().strip()
cleaned = ''.join(c.lower() for c in s if c.isalnum())
print("Да" if cleaned == cleaned[::-1] else "Нет")
```

Рисунок 5 – Листинг программы для задания 5

Очистка строки реализована с помощью генератора: ‘c.isalnum()‘ отбирает только буквы и цифры. Метод ‘[::-1]‘ создаёт реверс строки. Сравнение даёт булев результат, который преобразуется в «Да» или «Нет» через тернарный оператор. Такой подход корректно обрабатывает фразы на русском и английском, игнорируя пробелы и пунктуацию.