

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

Н.Д. Ананко

Ассистент

преподавателя

М.В. Водяницкий

## Задание

Выполнить задания на Python и оформить отчет по стандартам ВВГУ.

**Задание 1.** Написать функцию, которая конвертирует время из одной величины в другую.

- 1) На вход подается:
- 2) число (величина времени)
- 3) исходная единица измерения
- 4) единица измерения, в которую нужно перевести

Функция должна вернуть конвертированное значение

**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:

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

Зависимость от срока:

- 1) первые 3 года - 3%
- 2) от 4 до 6 лет - 5%
- 3) 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)

**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

**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:

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

На вход подаются:

- 1) размер матрицы  $n$  (для квадратной матрицы  $n \times n$ )
- 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 **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:

- 1) Да, если это палиндром
- 2) Нет, если это не палиндром

## Содержание

1	Выполнение работы .....	3
1.1	Задание 1 .....	3
1.2	Задание 2 .....	3
1.3	Задание 3 .....	4
1.4	Задание 4 .....	5
1.5	Задание 5 .....	6

## 1 Выполнение работы

### 1.1 Задание 1

В данном задании требуется написать функцию для конвертирования времени. На рисунке 1 представлен код полученной программы.

```

1 TimeUnits = {
2     "s": 1,
3     "m": 60,
4     "h": 3600,
5     "d": 86400
6 }
7
8 def ConvertTime(time_str, target_unit):
9     for unit in TimeUnits:
10         if time_str.endswith(unit):
11             value = int(time_str[:-len(unit)])
12             seconds = value * TimeUnits[unit]
13             result = seconds / TimeUnits[target_unit]
14             return f"{int(result)}{target_unit}"
15     return "Error!"
16
17 inp = input().split()
18 print(ConvertTime(inp[0], inp[1]))

```

Рисунок 1 – Листинг программы для задания 1

Пояснение работы программы:

- 1) Объявляется словарь TimeUnits с соотношениями единиц времени в секундах
- 2) Определяется функция ConvertTime, принимающая время и целевую единицу
- 3) В функции из входной строки убирается единица, число переводится в секунды
- 4) Результат делится на значение целевой единицы и возвращается с её обозначением
- 5) Программа запрашивает у пользователя ввод (время и целевая единица)
- 6) Вызывает функцию и выводит результат на экран

После выполнения программы в консоль выводится сконвертированное значение времени в указанной единице.

### 1.2 Задание 2

В данном задании требуется написать функцию для расчёта прибыли по банковскому вкладу с учётом сложных процентов и зависимой ставки. На рисунке 2 представлен код полученной программы.

```

1 def calculate_profit(deposit, years):
2     if deposit < 30000:
3         return 0.0
4     if years <= 3:
5         base_rate = 0.03
6     elif years <= 6:
7         base_rate = 0.05
8     else:
9         base_rate = 0.02
10    extra = min(deposit // 10000 * 0.003, 0.05 - base_rate)
11    rate = min(base_rate + extra, 0.05)
12    profit = deposit * ((1 + rate) ** years - 1)
13    return round(profit, 2)
14
15 d = int(input())
16 y = int(input())
17 print(calculate_profit(d, y))

```

Рисунок 2 – Листинг программы для задания 2

- 1) Проверяется, что сумма вклада не меньше 30 000 рублей
- 2) Определяется базовая ставка в зависимости от срока: 3% — до 3 лет, 5% — 4–6 лет, 2% — более 6 лет
- 3) Рассчитывается дополнительная ставка: 0.3% за каждые 10 000 рублей, но не более 5% суммарно
- 4) Итоговая ставка — минимум из (базовая + доп.) и 5%
- 5) Прибыль рассчитывается по формуле сложного процента: вклад  $\cdot (1 + \text{ставка})^{\text{лет}}$
- 6) Результат округляется до двух знаков и возвращается

После выполнения программы в консоль выводится сумма заработанных процентов (прибыль) за указанный срок.

### 1.3 Задание 3

В данном задании требуется написать функцию для вывода всех простых чисел в заданном диапазоне с обработкой некорректных данных. На рисунке 3 представлен код полученной программы.

```

1 def is_prime(n):
2     if n < 2:
3         return False
4     i = 2
5     while i * i <= n:
6         if n % i == 0:
7             return False
8         i += 1
9     return True
10
11 def print_primes(a, b):
12     if a < 0 or a > b:
13         return "Error!"
14     primes = []
15     for num in range(a, b + 1):
16         if is_prime(num):
17             primes.append(str(num))
18     if not primes:
19         return "Error!"
20     return " ".join(primes)
21
22 a = int(input())
23 b = int(input())
24 print(print_primes(a, b))

```

Рисунок 3 – Листинг программы для задания 3

Пояснение работы программы:

- 1) Определяется функция IsPrime, проверяющая, является ли число простым (до корня из числа)
- 2) Определяется функция PrintAllPrimes, принимающая начало и конец диапазона
- 3) Проверяются входные данные: начало  $\geq 0$  и начало  $\leq$  конец — иначе выводится "Error!"
- 4) Перебираются все числа в диапазоне, для каждого вызывается IsPrime
- 5) Простые числа добавляются в строку через пробел
- 6) Если ни одного простого числа не найдено — выводится "Error! иначе — строка с числами

После выполнения программы в консоль выводится строка с простыми числами в указанном диапазоне или сообщение "Error!" при некорректном вводе или отсутствии простых чисел.

#### 1.4 Задание 4

В данном задании требуется написать функцию для сложения двух квадратных матриц одинакового размера. На рисунке 4 представлен код полученной программы.

```

1 def add_matrices(n, m1, m2):
2     if n <= 2:
3         return "Error!"
4     res = []
5     for i in range(n):
6         row = []
7         for j in range(n):
8             row.append(m1[i][j] + m2[i][j])
9         res.append(row)
10    return res
11
12 n = int(input())
13 if n <= 2:
14     print("Error!")
15 else:
16     m1 = [list(map(int, input().split())) for _ in range(n)]
17     m2 = [list(map(int, input().split())) for _ in range(n)]
18     result = add_matrices(n, m1, m2)
19     if result != "Error!":
20         for r in result:
21             print(" ".join(map(str, r)))
22     else:
23         print("Error!")

```

Рисунок 4 – Листинг программы для задания 4

Пояснение работы программы:

- 1) Проверяется, что размер матрицы  $n$  строго больше 2 — иначе возвращается "Error!"
- 2) Создаётся пустой результирующий список Result
- 3) Внешний цикл по строкам, внутренний — по столбцам: складываются элементы с одинаковыми индексами из обеих матриц
- 4) Суммы добавляются в строки результата, строки — в общую матрицу
- 5) Функция возвращает итоговую матрицу или "Error!" при нарушении условий
- 6) Программа считывает размер и две матрицы, вызывает функцию и выводит результат построчно

После выполнения программы в консоль выводится результирующая матрица (по строкам через пробел) или сообщение "Error!" при неверном размере или формате.

### 1.5 Задание 5

В данном задании требуется написать функцию для проверки, является ли строка палиндромом (без учёта пробелов, регистра и знаков препинания). На рисунке 5 представлен код полученной программы.



```

1 import string
2
3 def is_palindrome(s):
4     clean = ""
5     for ch in s.lower():
6         if ch.isalnum():
7             clean += ch
8     return "Да" if clean == clean[::-1] else "Нет"
9
10 s = input()
11 print(is_palindrome(s))

```

Рисунок 5 – Листинг программы для задания 5

Пояснение работы программы:

- 1) Импортируется модуль string для работы со знаками препинания
- 2) Строка приводится к нижнему регистру
- 3) Удаляются все пробелы и знаки препинания из строки
- 4) Сравнивается полученная строка с её обратной версией
- 5) Если совпадает — возвращается "Да иначе — "Нет"

После выполнения программы в консоль выводится "Да если строка — палиндром, или "Нет"— если нет.