

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

# ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №6

По дисциплине  
«Информатика и программирование»

Студент		
гр. Бин-25-2	_____	И.А. Маклаков
Ассистент		
Преподавателя	_____	М.В. Водяницкий

## Задание

Выполнить задания и написать отчёт по стандартам ВВГУ.

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

На вход подается:

- 1) число (величина времени)
- 2) исходная единица измерения
- 3) 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:

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 учёта 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 97 101 103 107 109 113
0 1	Error!

(Формат вывода списка простых чисел может быть любым удобным: в строку через пробел, в несколько строк и т.п.)

**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, 3×3, 4×4 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

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.):

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 учёта пробелов, регистра и знаков препинания - эти правила нужно явно задать в своей реализации)

На вход подаётся строка. На выходе:

- 1) Да, если это палиндром
- 2) 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 .....	3
1.1	Задание 1 .....	3
1.2	Задание 2 .....	4
1.3	Задание 3 .....	5
1.4	Задание 4 .....	6
1.5	Задание 5 .....	8

# 1 Выполнение работы

## 1.1 Задание 1

В данном задании нужно написать код, который будет преобразовывать из одной единицы времени в другую. Программа выполняющая это изображена на Рисунке 1.

```

1  # -*- coding: cp1251 -*-
2  print('Works with ms, sec, min, h (milliseconds, seconds, minutes, hours)')
3  forms = ('ms', 'sec', 'min', 'h')
4  ms = 'ms'
5  sec = 'sec'
6  m = 'min'
7  h = 'h'
8  def trans(a, b, c):
9      if b == ms:
10         a = a
11     elif b == sec:
12         a = a * 1000
13     elif b == m:
14         a = a * 60 * 1000
15     elif b == h:
16         a = a * 60 * 60 * 1000
17
18     if c == ms:
19         a = a
20     elif c == sec:
21         a = a / 1000
22     elif c == m:
23         a = a / 1000 / 60
24     elif c == h:
25         a = a / 1000 / 60 / 60
26     return a
27 try:
28     a = float(input('Enter Value:'))
29     if a < 0:
30         print('Time can\'t be negative')
31         exit()
32     elif a == 0:
33         print('0 is the same in any other time')
34         exit()
35 except ValueError:
36     print('Invalid Input(Allowed - 15, 13.5, etc. )')
37     exit()
38
39
40 b, c = map(str, input('Enter time transform(like |sec h|, |h min|, etc.): ').split())
41 if b not in forms or c not in forms:
42     print('Invalid Input(Allowed - ms, sec, min, h)')
43     exit()
44
45 print("\033[H\033[J", end="") # \033[H перемещает курсор в верхний левый угол экрана,
46 print(f"""Entered data:
47 {a} {b}""")
48
49 print(f"""Output data:
50 {trans(a, b, c):.3f} {c}""")

```

Рисунок 1 - Код для задания 1

Пояснение работы программы:

- 1) Командой print() выводим с чем работает программа
- 2) Вводим кортеж с обозначениями и для каждой переменной её тип в str(), для минут я взял m = 'min', чтобы не запускалась операция min()
- 3) Ввожу функцию trans(a, b, c), для внутренней переменной a будут значения плавающей точки(тип данных float()), внутренняя переменная b это входная единица времени, внутренняя переменная c это единица времени на выходе

4) Работа функции `trans(a, b, c)` состоит в том, чтобы переводить входную единицу времени в миллисекунды, а на выход переводить в нужную

5) Программа обрабатывает ошибки по типу: числа меньше 0, введено значение 0, неверный ввод входного значения

6) `\033[H` - переводит курсор в левых верхний угол, `\033[J` - очищает всё что находится ниже курсора, сделано для чистоты экрана(бесполезный функционал в данной программе)

7) Командой `print()` выводим исходные данные, а после повторной командой `print()` выводим выходные данные из функции `trans(a, b, c)` с форматом вызова через f-строку, значение выводится плавающей точкой с тремя знаками после запятой

Программа корректно выполняет перевод времени.

## 1.2 Задание 2

В данном задании нужно написать код, который будет выводить получаемую прибыль от вклада под сложным процентом через  $n$  лет. Программа изображена на Рисунке 2.

```

1  print('Minimal deposit 30k(30000)')
2  def calcu(a, b):
3      if a // 10000 > 16:
4          k1 = 5
5      else:
6          k1 = 0.3 * (a // 10000)
7      if 1 <= b <= 3:
8          k2 = 3
9      elif 4 <= b <= 6:
10         k2 = 5
11     elif b > 6:
12         k2 = 2
13     ks = k1 + k2
14     c = 0
15     y = 0
16     ksf = ks / 100
17     a = a
18     while y != b:
19         c += a * ksf
20         y += 1
21         a += c
22     c = round(c)
23     return {
24         'cof1': k1,
25         'cof2': k2,
26         'income': c
27     }
28     try:
29         a, b = map(int, input('Enter your deposit and length in integer(value years): ').split())
30         if a < 30000:
31             print('Less than minimal deposit')
32             exit()
33         elif b < 1:
34             print('Deposit lenght can\'t be less 1 year')
35             exit()
36         res = calcu(a, b)
37         print(f"""
38         coefficient 1 - {res['cof1']}
39         coefficient 2 - {res['cof2']}
40         income - {res['income']}
41         """)
42     except ValueError:
43         print('Not allowed any other symbols except integers')
44 
```

Рисунок 2 - Код для задания 2

Пояснение работы программы:

- 1) Командой print() выводим минимальную сумму вклада
- 2) Вводим функцию calcu(a, b), где внутренняя переменная a это сумма вклада, а внутренняя переменная b это количество лет
- 3) В функции calcu(a, b) вычитывается k1(первый коэффициент по формуле a делить на 10.000, максимум 5%), k2(второй коэффициент связанный с количеством лет, где 1-3 года это 3%, 4-6 лет это 5%, больше 6 лет это 2%) и прибыль со вклада через цикл while, конечная прибыль округляется функцией round()
- 4) От пользователя требуется ввод через input(), сумма вклада и количество лет через пробел
- 5) Программа обрабатывает ошибки по типу: Сумма вклада меньше минимума, Длительность вклада меньше 1(меньше одного года), неправильно введены входные данные
- 6) Через команду print() с f-строкой выводится первый и второй коэффициент, прибыль со вклада, без первоначальной суммы

Программа верно рассчитывает коэффициенты и отображает прибыль по сложному проценту.

### 1.3 Задание 3

В данном задании нужно написать код, который выводит все простые числа в указанном диапазоне. Программа изображена на Рисунке 3.

```

1  def find_all_common(start, end):
2      if start > end:
3          return 'Start can\'t be more than end'
4      elif start < 1 or end < 1:
5          return 'Start or end can\'t be less than 1'
6      elif end == 1:
7          return 'End can\'t be 1'
8
9      def common(c):
10         if c < 2:
11             return False
12         if c == 2:
13             return True
14         if c % 2 == 0:
15             return False
16
17         for i in range(3, int(c**0.5) + 1, 2):
18             if c % i == 0:
19                 return False
20         return True
21
22     commons = []
23     for com in range(start, end + 1):
24         if common(com):
25             commons.append(com)
26
27     if commons:
28         return " ".join(map(str, commons))
29     else:
30         return 'Error'
31
32 try:
33     a, b = map(int, input('Enter radius(3 5, 1 17, 8 90, etc.): ').split())
34     print(find_all_common(a, b))
35 except ValueError:
36     print('Start and end should be integers')

```

Рисунок 3 - Код для задания 3



Пояснение работы программы:

- 1) Вводим функцию `find_all_common(start, end)`
  - 2) В ней обрабатываем ошибки по типу: начала больше конца, начало или конец меньше 1, конец равен 1
  - 3) Вводим внутри функцию ещё одну функцию `common(c)`
  - 4) Функция `common(c)` находит простые числа по алгоритму:  $c < 2 \Rightarrow \text{False}$  (число не вернётся),  $c = 2 \Rightarrow \text{True}$  (Возвращает число), если остаток деления  $c$  на 2 будет 0  $\Rightarrow \text{False}$ ; Цикл `for i in range(3, int(c**0.5) + 1, 2)` проверяет делимость от 3 до квадратного корня  $c$ , с шагом 2. Если остаток от деления будет 0, то вернётся `False`, иначе `True`
  - 5) Затем в функцию `find_all_common(start, end)` добавляется список `commons`, запускается цикл `for com in range(start, end + 1)`, то есть от начала до конца включительно, вызывается `common(com)`, если при переборе числа из `com` в `common` вернётся `True`, то она добавляется в этот список
  - 6) Затем список возвращается во внешнюю среду из функции, преобразованный в строку функцией `"".join` предварительно преобразованный в тип строки `str()`
  - 7) Если выйдет пустой список, то вернётся во внешнюю строка `Error`
  - 8) В блоке `try-except` выполняем основной код
  - 9) Запрашивает у пользователя 2 числа разделённые пробелом
  - 10) Потом командой `print()` вызывает функцию `find_all_common(a, b)` к входным данным
  - 11) В случае неверного ввода выдаст ошибку
- Программа успешно находит все простые числа по заданному диапазону, корректно обрабатывает ошибки.

#### 1.4 Задание 4

В данном задании нужно написать код, который будет складывать 2 матрицы одинаковой размерности, но не меньше 3x3. Данная программа корректно выполняет все условия чтобы правильно складывать 2 матрицы одной размерности. Программа изображена на Рисунке 4.

```

1     def matrixsum(n, mat1, mat2):
2         res = []
3         for i in range(n):
4             row = []
5             for j in range(n):
6                 row.append(mat1[i][j] + mat2[i][j])
7             res.append(row)
8         return res
9
10    def matmade(n):
11        matrix = []
12        for l in range(n):
13            try:
14                row = list(map(int, input().split()))
15                if len(row) != n:
16                    print('Row has incorrect number of elements')
17                    exit()
18                matrix.append(row)
19            except ValueError:
20                print('Matrix allow only integers')
21                return None
22        return matrix
23
24    def matprint(matrix):
25        for row in matrix:
26            print(" ".join(map(str, row)))
27
28    try:
29        n = int(input('Enter matrix size nxn(n > 2): '))
30        if n < 3:
31            print('Matrix can\'t be less than 3x3')
32            exit()
33
34        print('Enter Matrix 1, elements that n rows with integers separated by space')
35        matrix1 = matmade(n)
36        if matrix1 == None:
37            exit()
38
39        print('Enter Matrix 2, elements that n rows with integers separated by space')
40        matrix2 = matmade(n)
41        if matrix2 == None:
42            exit()
43
44        result = matrixsum(n, matrix1, matrix2)
45
46        print("Result matrix:")
47        matprint(result)
48
49    except ValueError:
50        print('Invalid input: allow only integer')
51        exit()

```

Рисунок 4 - Код для задания 4

Пояснение работы программы:

- 1) Вводим функцию matrixsum(n, mat1, mat2), где n это размер матриц, mat1 это первая матрица, mat2 это вторая матрица
- 2) В ней создаём список res, находим сумму матриц по формуле  $C[i][j] = mat1[i][j] + mat2[i][j]$  и возвращаем значение res
- 3) Вводим функцию matmade(n), где n это размер матрицы
- 4) В ней создаём список matrix, создаём цикл for l in range(n) и выполняем «создание» матрицы в блоке try-except. Если введено не 3 целых числа выйдет ошибка и программа завершится, если введено не целое число, выйдет ошибка и возвращаем None. Если всё хорошо, то возвращаем значение списка matrix
- 5) Вводим функцию matprint(matrix)

6) Функция `matprint(matrix)` преобразует элементы в строки через `map(str, row)`, и разделяет их пробелами через `' '.join()`

7) В блоке `try-except` выполняем основной код, запрашиваем у пользователя `n` с условием `n > 2`, иначе ошибка и завершение программы

8) Через `print()` просим ввести первую матрицу `n` строк с `n` количеством элементов и создаём переменную `matrix1` со значением вызванной функции `matmade(n)`, если вернётся `None`, то программа завершится

9) Через `print()` просим ввести вторую матрицу `n` строк с `n` количеством элементов и создаём переменную `matrix2` со значением вызванной функции `matmade(n)`, если вернётся `None`, то программа завершится

10) Создаём переменную `result` и даём ей значение функции `matrixsum(n, matrix1, matrix2)`

11) Через `print()` выводим «Результативная матрица:» и выполняем функцию `matprint(result)`

Программа корректно обрабатывает ошибки и неверный ввод, находит и отображает результативную матрицу.

## 1.5 Задание 5

В данном задании нужно написать код, который сможет определить является ли предложение палиндромом или нет. Программа изображена на Рисунке 5.

```

1  def check_for_palindrom(a):
2      clear = ''.join(char.lower() for char in a if char.isalnum())
3      if clear == clear[::-1]:
4          return 'Yes, it\'s palindrom'
5      else:
6          return 'No, it\'s not palindrom'
7  user = str(input('Enter your sentence: '))
8  print(check_for_palindrom(user))

```

Рисунок 5 - Код для задания 5

Пояснение работы программы:

- 1) Вводим функцию `check_for_palindrom(a)`
  - 2) Создаём переменную `clear` которая получает значение `a`, преобразовав её в нижний регистр(`char.lower()`), также оставив только буквы и цифры(`char.isalnum()`)
  - 3) Условие: Если переменная `clear` равна себе зеркальной(`[::-1]`) то возвращает «Да, это палиндром», иначе «Нет, это не палиндром»
  - 4) Запрашивается ввод пользователя в переменную `user`
  - 5) Команда `print()` вызывает функцию `check_for_palindrom(a)` и отображает ответ
- Данная программа справляется со своей задачей.