

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 _____ С.Е. Казюта
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.

На вход подается:

* число (величина времени) * исходная единица измерения * 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:

* 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)

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

(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:

* 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 ‘ $n \times n$ ’) 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:

““txt 2 2 5 5 3 5 2 4 1 ““

Выход:

““txt 7 7 9 4 ““

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

““txt 1 4 5 ““

Выход:

““txt 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

Содержание

1 Выполнение работы.....	4
1.1 Задание 1	4
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

Задача данной программы это конвертация времени из одной величины в другую.

Алгоритм работы:

- 1) Пользователь вводит время, и из какой в какую величину он хочет его конвертировать.
- 2) Пользователь конвертирует время.
- 3) Программа возвращает конвертированное время.

```

1 ino = input().split(' ',1)
2 nume = int(ino[0][:-1])
3 frome = ino[0][-1]
4 toe = ino[1]
5 times = ['s','m','h']
6 diff = times.index(toe) - times.index(frome)
7
8 while diff != 0:
9     if diff > 0:
10         diff -= 1
11         nume /= 60
12     elif diff < 0:
13         diff += 1
14         nume *= 60
15     else:
16         break
17
18 print(str(round(nume, 3))+toe)

```

Рисунок 1 – Листинг программы для задания 1

Ключевые элементы кода:

- 1) ino = input().split(' ',1) - разделяет ввод пользователя на информацию с которой можно работать.
- 2) nume = int(ino[0][:-1]) - выделяет число.
- 3) frome = ino[0][-1] toe = ino[1] - выделяют из какой ед. измерения в какую нужно перевести время.
- 4) times = ['s','m','h'] - список ед. измерений на котором держится логика программы.
- 5) diff = times.index(toe) - times.index(frome) - определяет сколько раз нужно разделить или умножить время на 60 чтобы конвертировать его время из одной величины в другую.
- 6) while diff != 0: - цикл отвечающий за деление или умножение времени на 60.
- 7) print(str(round(nume, 3))+toe) - выводит результат.

1.2 Задание 2

Задача данной программы это рассчёт прибыли пользователя без учета первоначально вложенной суммы.

Алгоритм работы:

- 1) Пользователь сумму и время.
- 2) Программа делает сложные расчёты.
- 3) Программа выводит чистую прибыль.

```

1 while 1:
2     ino = input().split(' ',1)
3     input_money = int(ino[0])
4     time_left = int(ino[1])
5     stavka = 1
6     if input_money < 30000:
7         print('not enough money, try again (need at least
8             30000)')
9         continue
10    else:
11        break
12
13 stavka += input_money/10000*0.003
14 if stavka > 1.05:
15     stavka = 1.05
16 money = input_money
17
18 for time_unit in range(time_left):
19     if time_unit+1 <= 3:
20         money *= stavka + 0.03
21     elif time_unit+1 <= 6:
22         money *= stavka + 0.05
23     elif time_unit+1 > 6:
24         money *= stavka + 0.02
25
26 print(money-input_money)

```

Рисунок 2 – Листинг программы для задания 2

Ключевые элементы кода:

- 1) while 1 - эта часть кода нужно чтобы при ошибке ввода программа просила пользователя ввести данные ещё раз.
- 2) ino = input().split(' ',1) - разделяет ввод пользователя на информацию с которой можно работать.
- 3) input-money = int(ino[0]) - определяет ко-во вложенных денег.
- 4) time-left = int(ino[1]) - определяет кол-во времени.
- 5) if input-money < 30000: - перезапускает цикл при неверном вводе.
- 6) stavka += input-money/10000*0.003 - отвечает за расчёт процентной ставки.
- 7) for time-unit in range(time-left): - расчитывает рост прибыли учитывая процентную ставку.
- 8) print(money-input-money) - выводит результат.

1.3 Задание 3

Задача данной программы это вывести все простые числа в заданном диапазоне.

Алгоритм работы:

- 1) Пользователь вводит диапазон чисел.
- 2) Программа анализирует каждое число в диапазоне.
- 3) Программа выводит все простые числа в заданном диапазоне.

```

1 ino = input().split(' ', 1)
2
3 if int(ino[1]) < int(ino[0]):
4     print('Error!')
5
6 cnt = 0
7 for num in range(int(ino[0]),int(ino[1])):
8     devs = 0
9     for i in range(1,int(num**0.5)+1):
10        if num % i == 0:
11            devs += 1
12        if devs == 1:
13            if num == 1:
14                continue
15            print(num, end = ' ')
16            cnt += 1
17
18 if cnt == 0:
19     print('Error!')

```

Рисунок 3 – Листинг программы для задания 3

Ключевые элементы кода:

- 1) ino = input().split(' ', 1) - пользователь вводит диапазон.
- 2) if int(ino[1]) < int(ino[0]): - программа проверяет пользователя на ошибку.
- 3) for num in range(int(ino[0]),int(ino[1])): - цикл выводящий каждое число в заданном диапазоне.
- 4) for i in range(1,int(num**0.5)+1): - цикл проверяющий сколько раз число делится на числа до своего корня.
- 5) if devs == 1: - если кол-во делителей равно 1 то число простое.
- 6) print(num, end = ' ') - выводит простое число
- 7) if cnt == 0: print('Error!') - выводит ошибку если нет ни одного простого числа.

1.4 Задание 4

Задача данной программы это реализовать функцию сложения двух матриц.

Алгоритм работы:

- 1) Пользователь вводит размер матрицы.
- 2) Пользователь вводит значения первой матрицы.
- 3) Пользователь вводит значения второй матрицы.

- 4) Программа складывает матрицы.
- 5) Программа выводит результат сложения двух матриц.

```

1 def matrix_master(x, merge = False, mt1 = '', mt2 = ''):
2     matrix = []
3     for i in range(x):
4         while 1:
5             if merge:
6                 matrix_input = [int(mt1[i][mt_element]) +
7                     int(mt2[i][mt_element]) for mt_element in range(x)]
8                 matrix.extend([matrix_input])
9             else:
10                 matrix_input = input().split(' ')
11                 matrix.extend([matrix_input])
12                 if len(matrix_input) > x:
13                     print('Error!')
14                     continue
15                 break
16             return matrix
17
18     while 1:
19         ino = int(input())
20         if ino < 2:
21             print('Error!')
22             continue
23         break
24
25     matrix_1 = matrix_master(ino)
26     matrix_2 = matrix_master(ino)
27     matrix_3 = matrix_master(ino, merge = True, mt1 = matrix_1,
28                               mt2 = matrix_2)
29
30     print()
31     for i in matrix_3:
32         print(*i)

```

Рисунок 4 – Листинг программы для задания 4

Ключевые элементы кода:

- 1) def matrix-master(x, merge = False, mt1 = "", mt2 = ""): - функция отвечающая за создание и слияние матриц.
- 2) if merge: matrix-input = [int(mt1[i][mt-element]) + int(mt2[i][mt-element]) for mt-element in range(x)] - складывает элементы матрицы при merge = True.
- 3) else: matrix-input = input().split(' ') - создаёт элементы матрицы на основе ввода пользователя при merge = False.
- 4) if len(matrix-input) > x: print('Error!') - выводит сообщение об ошибке при неправильном вводе.
- 5) while 1: ino = int(input()) - принимает ввод пользователя до тех пор пока он не будет правильным.
- 6) if ino < 2: print('Error!') - выводит сообщение об ошибке при неправильном вводе.
- 7) matrix-1 = matrix-master(ino) - создаёт первую матрицу.

- 8) matrix-2 = matrix-master(ino) - создаёт вторую матрицу.
- 9) matrix-3 = matrix-master(ino, merge = True, mt1 = matrix-1, mt2 = matrix-2) - складывает первую и вторую матрицы.
- 10) for i in matrix-3: print(*i) - выводит результат сложения двух матриц.

1.5 Задание 5

Задача данной программы это определить, является ли строка палиндромом.

Алгоритм работы:

- 1) Пользователь вводит строку.
- 2) Программа проверяет является ли введённая пользователем строка палиндромом.
- 3) Программа выводит положительный или отрицательный ответ.

```

1 ino = input().lower().replace(' ', '')
2 half_len = int(len(ino) / 2)
3 if len(ino) % 2 != 0:
4     ino = ino[:half_len] + ino[half_len + 1:]
5 if ino[:half_len] == ino[half_len:][::-1]:
6     print('Да')
7 else:
8     print('Нет')

```

Рисунок 5 – Листинг программы для задания 5

Ключевые элементы кода:

- 1) ino = input().lower().replace(' ', '') - программа обрабатывает введённую пользователем строку.
- 2) half-len = int(len(ino) / 2) - программа определяет длину половины строки с округлением в меньшую сторону.
- 3) if len(ino) %. 2 != 0: ino = ino[:half-len] + ino[half-len + 1:] - если строка не делится без остатка на свою половину то программа убирает лишний символ не влияющий на результат работы программы.
- 4) if ino[:half-len] == ino[half-len:][::-1]: print('Да') - программа выводит положительный ответ если строка является палиндромом.
- 5) else: print('Нет') - программа выводит отрицательный ответ если строка не является палиндромом.