

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 №5
по дисциплине
«Информатика и программирование»

Студент		
гр. БИН-25-3	_____	М.В. Кирийчук
Ассистент		
преподавателя	_____	М.В. Водяницкий

Задание

Выполнение основных заданий (задания 1-10) Написание отчета по стандартам оформления ВВГУ

Задание 1. Дан список из 10 различных целых чисел. Необходимо найти в нем число 3 и заменить на 30.

Задание 2. Дан список из 5 целых чисел. Необходимо превратить его в список квадратов этих чисел.

Задание 3. Имеется список различных целых чисел. Программа должна найти наибольшее из чисел списка и разделить его на длину списка.

Задание 4. Имеется кортеж из нескольких произвольных элементов. Необходимо этот кортеж отсортировать. Если хотя бы один элемент не является числом, то кортеж остается неизменным.

Задание 5. Имеется словарь товаров в магазине. Необходимо найти товар с минимальной и максимальной ценой.

Задание 6. Имеется список произвольных элементов. Необходимо на основе этого списка создать словарь, где каждый элемент списка будет и ключом, и значением.

Задание 7. Имеется словарь перевода английских слов на русский, где ключ английского слова, значение - русского. Необходимо реализовать программу которая получает на ввод русское слово и результатом выдает перевод на английский.

Задание 8. Реализовать игру Камень-Ножницы-Бумага-Ящерица-Спок. Программа должна запрашивать у пользователя ввод одного из вариантов. Второй вариант случайно генерирует сама программа и возвращает победителя.

- Ножницы режут бумагу - Бумага покрывает камень - Камень давит ящерицу - Ящерица отравляет Спока - Спок ломает ножницы - Ножницы обезглавливают ящерицу - Ящерица съедает бумагу - Бумага подставляет Спока - Спок испаряет камень - Камень разбивает ножницы

Задание 9. Дан список слов, например:
"яблоко" "груша" "банан" "киви" "апельсин" "ананас"

Необходимо создать новый словарь, где:

- Ключом будет первая буква слова - Значением - список всех слов, начинающихся с этой буквы

Пример результата:

```
{ 'я': 'яблоко', 'г': 'груша', 'б': 'банан', 'к': 'киви', 'а': 'апельсин',  
'ананас' }
```

Задание 10. Дан список кортежей, где каждый кортеж содержит имя студента и его оценки, например:

```
{[("Анна [5, 4, 5]), ("Иван [3, 4, 4]), ("Мария [5, 5, 5])]}
```

Содержание

1	Выполнение работы	3
1.1	Задание 1	3
1.2	Задание 2	3
1.3	Задание 3	3
1.4	Задание 4	3
1.5	Задание 5	4
1.6	Задание 6	4
1.7	Задание 7	5
1.8	Задание 8	5
1.9	Задание 9	6
1.10	Задание 10	7

1 Выполнение работы

1.1 Задание 1

Создается список из 10 чисел, `len(numbers)` возвращает длину списка, `range` создает последовательность 0-9. Цикл выполнится 10 раз, `i` будет принимать значения от 0 до 9. При истинном условии выполняется замена числа 3 на 30. На рисунке 1 представлен код программы.

```
1 print("task_1")
2
3 numbers = [2, 6, 4, 3, 11, 88, 908, 55, 6, 97]
4 print("Исходный список:", numbers)
5
6 for i in range(len(numbers)):
7     if numbers[i] == 3:
8         numbers[i] = 30
9 print("Измененный список:", numbers)
```

Рисунок 1 – Листинг программы для задания 1

1.2 Задание 2

Создается список из 5 чисел, `map` - применяет последовательно к каждому элементу функцию `lambda`, которая принимает число `x` и возвращает его квадрат. На рисунке 2 представлен код программы.

```
1 print("task_2")
2
3 numbers = [2, 7, 5, 3, 8]
4
5 print(*map(lambda x: x**2, numbers))
```

Рисунок 2 – Листинг программы для задания 2

1.3 Задание 3

Создается список чисел. Функция `max` находит максимальный элемент в последовательности, функция `len` возвращает количество элементов в последовательности. Затем происходит деление. С помощью `f`-строки выводим результат. `f`-строка позволяет вставить переменные прямо в строку. На рисунке 3 представлен код программы.

```
1 print("task_3")
2
3 numbers = [0, 9, 54, 444, 88, 13, 66, 99]
4 result = max(numbers)/len(numbers)
5
6 print(f"Результат {result}")
```

Рисунок 3 – Листинг программы для задания 3

1.4 Задание 4

Объявляется функция `sort tuple` с параметром функции `t`, который применяет кортеж на вывод. Итерация `for x in t` последовательно перебирает все элементы кортежа, `isinstance` проверяет на тип данных, возвращает `True` и `False`. Функция `sorted` сортирует все элементы по возрастанию, `tuple` преобразует список в кортеж, `return` завершает выполнение функции, возвращает результат вызывающему коду. На рисунке 4 представлен код решения.

```

1 print("task_4")
2
3 def sort_tuple(t):
4
5     numbers = [x for x in t if isinstance(x, (int, float))]
6     return tuple(sorted(numbers))
7
8
9 x = [9, 0.66, "indifference", "apps", 99, 14, None, 'True']
10 result = sort_tuple(tuple(x))
11 print(result)

```

Рисунок 4 – Листинг программы для задания 4

1.5 Задание 5

Создаем функцию `help`, функция принимает один параметр, в нашем случае - словарь. Создается словарь, где ключами являются названия товаров, а цены значениями. Функция `min` берет каждый ключ и передает его в `products.get`, возвращает значение этого ключа. На рисунке 5 представлен код программы.

```

1 print("task_5")
2
3 def help(products):
4     min1=min(products, key=products.get)
5     max1=max(products, key=products.get)
6     return min1, max1
7 products={"milk":120,"bread":50,"pie":220,"beer":90, "eggs":150}
8 print(help(products))

```

Рисунок 5 – Листинг программы для задания 5

1.6 Задание 6

Создается список произвольных элементов разных типов данных, создается пустой словарь. Происходит поочередная итерация каждого элемента из списка. Если элемент не хешируемый т.е список, словарь, массив - то он просто игнорируется благодаря механизму `try-except`. На рисунке 6 представлен код программы.

```

1 print("task_6")
2
3 list = [22, 'tv', 2.11, 'wow', True, 33, [1,2], None]
4 fool = dict()
5 for i in list:
6     try:
7         fool[i] = i
8     except:
9         pass
10 print(fool)

```

Рисунок 6 – Листинг программы для задания 6

1.7 Задание 7

Создается словарь eng to rus в котором ключами являются английские слова, а значениями - русские переводы. Следом создается функция Find english translation с параметром russian word. Метод items возвращает ключ и значение из словаря, метод lower делает поиск регистронезависимым. На рисунке 7 представлен код программы.

```

1 print("task_7")
2
3 eng_to_rus = {
4     "sea": "море",
5     "air": "воздух",
6     "sky": "небо",
7     "sun": "солнце",
8     "moon": "луна",
9     "rain": "дождь"
10 }
11
12 def find_english_translation(russian_word):
13
14     for english, russian in eng_to_rus.items():
15         if russian == russian_word:
16             return english
17     return "Перевод не найден"
18
19 word = input("Введите русское слово: ")
20 result = find_english_translation(word.lower())
21 print(f"Английский перевод: {result}")

```

Рисунок 7 – Листинг программы для задания 7

1.8 Задание 8

Используем рандомайзер для генерации случайных чисел. Создаем функцию, которая содержит всю логику игры. Добавляем список всех вариантов, добавляем словарь, который определяет правила. Ключ это выбор игрока, значение это список вариантов, которые этот выбор побеждает. Цикл while true продолжается до команды выхода. Input получает ввод от пользователя, lower преобразует ввод в нижний регистр. Random.choice случайным образом выбирает вариант для компьютера. На рисунке 8 представлен код программы.

```

1 print("task_8")
2
3 import random
4
5 def simple_game():
6     choices = ["камень", "ножницы", "бумага", "ящерица", "спок"]
7
8     rules = {
9         "камень": ["ножницы", "ящерица"],
10        "ножницы": ["бумага", "ящерица"],
11        "бумага": ["камень", "спок"],
12        "ящерица": ["бумага", "спок"],
13        "спок": ["камень", "ножницы"]
14    }
15
16    while True:
17        user = input("Ваш выбор камень(, ножницыбумага,, ящерица, спок, выход): ").lower()
18
19        if user == "выход":
20            break
21        if user not in choices:
22            print("Неверный выбор!")
23            continue
24
25        computer = random.choice(choices)
26        print(f"Компьютер выбирает: {computer}")
27
28        if user == computer:
29            print("Ничья!")
30        elif computer in rules[user]:
31            print("Вы выиграли!")
32        else:
33            print("Компьютер выиграл!")
34
35    simple_game()

```

Рисунок 8 – Листинг программы для задания 8

1.9 Задание 9

Создается список слов и пустой словарь для хранения результата. Цикл перебирает каждое слово из списка, `f letter` извлекает первую букву слова, т.к. индексация идет с 0. If `f letter not in word dict` проверяет есть ли такая буква, как ключ в словаре. `Word dict` создает пустой список для буквы. `append word` добавляет текущее слово в список для этой буквы. На рисунке 9 представлен код программы.

```

1 print("task_9")
2
3 words = ["яблоко", "груша", "банан", "киви", "апельсин", "
  ананас"]
4
5 word_dict = {}
6 for word in words:
7     f_letter = word[0]
8     if f_letter not in word_dict:
9         word_dict[f_letter] = []
10    word_dict[f_letter].append(word)
11
12 print("Словарь по первой букве:")
13 print(word_dict)

```

Рисунок 9 – Листинг программы для задания 9

1.10 Задание 10

Имеется список кортежей. Создается генератор словаря. Best name переменная для хранения имени лучшего студента, best avg хранит наивысший средний балл, avg grades items возвращает ключ и значение из словаря. 2f форматирование до 2 знаков после запятой. На рисунке 10 представлен код программы.

```

1 print("task_10")
2
3 students = [("Анна", [5, 4, 5]), ("Иван", [3, 4, 4]), ("
  Мария", [5, 5, 5])]
4
5 avg_grades = {name: sum(grades)/len(grades) for name, grades
  in students}
6
7 best_name = ""
8 best_avg = 0
9 for name, avg in avg_grades.items():
10    if avg > best_avg:
11        best_avg = avg
12        best_name = name
13
14 print(f"Лучший студент: {best_name} со средним баллом {
  best_avg:.2f}")

```

Рисунок 10 – Листинг программы для задания 10