

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 №4
по дисциплине
«Информатика и программирование»

Студент		
гр. БИН-25-2	_____	К.Ф. Кучерчук
Ассистент		
преподавателя	_____	М.В. Водяницкий

Задание

Выполнить задания и оформить отчет по стандартам ВВГУ.

Задание 1. Написать программу, которая определяет, как будет вести себя кондиционер. Если температура в помещении 20 градусов и выше, то кондиционер выключается, если меньше - включается. Температура должна вводиться пользователем с консоли.

Пример:

Введите температуру: 18

Кондиционер включен

Задание 2. Год делится на четыре сезона: зима, весна, лето и осень. Написать программу, которая запрашивает у пользователя номер месяца и выводит к какому сезону этот месяц относится.

Пример:

Введите номер месяца: 4

Это весна

Задание 3. Считается, что один год, прожитый собакой, эквивалентен семи человеческим годам. При этом зачастую не учитывается, что собаки становятся абсолютно взрослыми уже к двум годам. Таким образом, многие предпочитают каждый из первых двух лет жизни собаки приравнивать к 10.5 годам человеческой жизни, а все последующие к 4.

Написать программу, которая будет переводить собачий возраст в человеческий. Программа должна корректно обрабатывать входные данные и выводить соответствующие сообщения об ошибках:

- Если вводится не число
- Если вводится число меньше 1
- Если вводится число большее 22

Пример:

Введите возраст собаки (в годах): 5

Возраст собаки в человеческих годах: 33.0

Пример:

Введите возраст собаки (в годах): 0

Ошибка: возраст должен быть не меньше 1

Задание 4. Число делиться на 6 только в случае соблюдения двух условий:

- Последняя цифра четная

- Сумма всех цифр делиться на 3

Написать программу, которая выведет делиться ли введенное число на 6 или нет.

Задание 5. Написать программу, которая будет проверять пароль на надежность. Пароль считается надежным, если его длина не менее 8 символов и если он содержит:

- Заглавные буквы латиницы
- Строчные буквы латиницы
- Числа
- Специальные знаки

В случае, если пароль не проходит по одному из условий, необходимо сообщить пользователю каким именно условиям он не удовлетворяет.

Пример:

Введите пароль: qwerty

Пароль ненадежный: отсутствуют заглавные буквы, числа и специальные символы

Задание 6. Написать программу, которая определяет, является ли введенный пользователем год високосным. Год считается високосным, если он делится на 4, но не делится на 100, либо если он делится на 400.

Пример:

Введите год: 2024

2024 - високосный год

Задание 7. Написать программу, которая запрашивает у пользователя три числа и выводит на экран наименьшее из них. При решении нельзя использовать встроенные функции `min()` и `max()`.

Пример:

Введите три числа: 8 3 5

Наименьшее число: 3

Задание 8. В магазине проводится акция. Акция работает по следующим правилам:

- Сумма $< 1000 \Rightarrow$ скидка - 0%
- Сумма $< 5000 \Rightarrow$ скидка - 5%
- Сумма $< 10000 \Rightarrow$ скидка - 10%
- Сумма $> 10000 \Rightarrow$ скидка - 15%

Напишите программу, которая запрашивает сумму покупки и выводит размер скидки и итоговую сумму к оплате.

Пример:

Введите сумму покупки: 7500

Ваша скидка: 10К оплате : 6750.0

Задание 9. Написать программу, которая определяет время суток по введенному часу (целое число от 0 до 23).

- С 0 до 5 часов - ночь
- С 6 до 11 часов - утро
- С 12 до 17 часов - день
- С 18 до 23 часов - вечер

Пример:

Введите час (0–23): 20

Сейчас вечер

Задание 10. Написать программу, которая определяет, является ли введенное число простым. Число называется простым, если оно больше 1 и делится только на 1 и само себя. Программа должна корректно обрабатывать некорректный ввод и выводить соответствующие сообщения об ошибках.

Пример:

Введите число: 17

17 - простое число

Содержание

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

1 Выполнение работы

1.1 Задание 1

В переменную `temp` при помощи функции `input()` вносим вводимые пользователем данные, сразу переводя их в целочисленный тип данных функцией `int()`. После при помощи условной конструкции `if else` проверяем переменную `temp` на соответствие условиям: если `temp` больше или равно 20 (при помощи оператора сравнения больше равно `>=`), то мы при помощи функции `print()` выводим пользователю сообщение 'Кондиционер выключается', в противном же случае при помощи той же функции выводится сообщение 'Кондиционер включается'. На рисунке 1 представлен код программы.

```
1 # Task_1
2
3 temp = int(input('Введите температуру: '))
4 if temp >= 20:
5     print('Кондиционер выключается...')
6 else:
7     print('Кондиционер включается...')
```

Рисунок 1 – Листинг программы для задания 1

1.2 Задание 2

Запрашиваем у пользователя на ввод строку (функция `input()`) и переводим её в число (функция `int()`). После при помощи условной конструкции `if else` проверяем, к какому диапазону значений принадлежит переменная `num`, и выводим соответствующее сообщение. На рисунке 2 представлен код программы.

```
1 # Task_2
2
3 num = int(input('Введите номер месяца: '))
4 if num in [1,2,12]:
5     print('Сейчас зима')
6 elif num in [3,4,5]:
7     print('Сейчас весна')
8 elif num in [6,7,8]:
9     print('Сейчас лето')
10 elif num in [9,10,11]:
11     print('Сейчас осень')
12 else:
13     print("Введён неверный номер месяца")
```

Рисунок 2 – Листинг программы для задания 2

1.3 Задание 3

Создаём переменную `dog-age` и вносим её значение, вводимое пользователем с консоли при помощи функции `input()`. После для обработки ошибок используем конструкцию `try expect`. Внутри блока `try` пробуем перевести переменную `dog-age` к целочисленному типу данных. Если у нас получается, то код продолжает работать, а если же мы не можем

привести её к этому типу данных, обработчик ошибок перемещает нас из блока `try` в блок `except`, внутри которого мы при помощи функции `print()` выводим сообщение о соответствующей ошибке. Далее внутри блока `try` мы создаём новую переменную `current-age` и присваиваем ей значение переменной `dog-age`, и создаём ещё одну переменную `human-age`, чтобы сохранять в неё возраст человека. После мы блоком `if else` проверяем входит ли значение переменной `dog-age` в область допустимых значений, и если нет то выводим соответствующее сообщение об ошибке. Если переменная удовлетворяет ОДС, то мы при помощи цикла `while` производим основные вычисления. Счётчиком будет являться переменная `current-age`, которую мы будем уменьшать в конце каждого цикла. Строками 13 и 14 мы показываем, что первые 2 года собаки считаются по 10.5 человеческих лет, остальные же по 4. И после завершения цикла `while`, а именно когда переменная `current-age` приравняется к нулю, мы при помощи функции `print()` выводим результат, записанный в переменную `human-age`. На рисунке 3 представлен код программы.

```

1 # Task_3
2
3 dog_age = input()
4
5 try:
6     dog_age = int(dog_age)
7     current_age = dog_age
8     human_age = 0
9     if dog_age > 22 or dog_age < 1:
10        print('Введён некорректный возраст')
11    else:
12        while current_age != 0:
13            if dog_age - current_age < 2:
14                human_age += 10.5
15            else:
16                human_age += 4
17                current_age -= 1
18            print(human_age)
19 except:
20    print('введено не число')

```

Рисунок 3 – Листинг программы для задания 3

1.4 Задание 4

Создаё переменную `num`, в которую при помощи функции `input()` заносим переменную, и при помощи функции `int()` сразу приводим её к целочисленному типу данных. Далее определяем функцию `div-by-six`, которая принимает 1 аргумент целочисленного типа данных, и возвращает булево значение. Далее создаём переменную `last-is-even`, содержащую тернарный оператор, который возвращает `True` если выполняется указанное условие и `False` если нет. Контекст этого условия следующий: мы должны проверить число на чётность, для этого достаточно чтобы последняя цифра числа делилась на два без остатка. Для этого мы приводим переменную `num` к строке функцией `str()`, затем при

помощи индекса[-1] получаем последний символ и функцией `int()` приводим его обратно к строке и после этого сравниваем его остаток от деления на два с нулём. В следующий строке мы создаём ещё одну переменную `summ-div-3`, содержащую тернарный оператор, который возвращает `True` если выполняется указанное условие и `False` если нет. Контекст условия: мы проверяем делится ли число на 3. Для этого необходимо чтобы сумма всех его чисел тоже делилась на три. Для получения суммы мы используем функцию `sum()`, которая возвращает число сумму, полученное сложение переданных в неё аргументов целочисленного типа данных. Таким образом мы помещаем в неё генератор `int(x) for x in list(str(num))`. Он работает следующим образом: мы берём `x` из списка, который мы получили путём приведения переменная `num` к строке функцией `str()` и последующим приведением её к списку функцией `list()`, после чего все `x` приводятся к целым числам и попадают в аргументы функции `sum()`. Потом сравниваем остаток от деления на три возвращённого ею значения с нулём. В конце функции возвращаем истину если переменные `last-is-even` и `summ-div-3` имеют значение `True` и ложь в противном случае. В конце мы выводим сообщение пользователю при помощи `f` строки. На рисунке 4 представлен код решения. На рисунке 4 представлен код решения.

```

1 #Task 4
2
3 num = int(input())
4
5 def div_by_six(num: int) -> bool:
6     last_is_even = True if int(str(num)[-1]) % 2 == 0 else
       False
7     summ_div_3 = True if sum(int(x) for x in list(str(num)))
       % 3 == 0 else False
8     return True if (last_is_even and summ_div_3) else False
9
10 print(f'Результат того делиться ли число {num} на 6: да{" " if
       div_by_six(num) else нет""}')

```

Рисунок 4 – Листинг программы для задания 4

1.5 Задание 5

Запрашиваем ввод пароля от пользователя и записываем в переменную `password`. Создаем словарь с условиями проверки пароля: Пароль не менее 8 символов; наличие заглавных букв латиницы (`any()` возвращает `True`, если хотя бы один символ удовлетворяет условию, `s.isupper()` проверяет, является ли символ заглавной буквой); наличие строчных букв латиницы; наличие цифр (`s.isdigit()` проверяет, является ли символ цифрой); проверка наличия специальных символов (`not s.isalnum()` проверяет, НЕ является ли символ буквой или цифрой). Проверяем каждое условие с помощью цикла (`items()` возвращает пары

(ключ, значение) из словаря). Если хотя бы одно условие не выполнено, общий результат становится False – пароль ненадежный. На рисунке 5 представлен код программы.

```

1 #Task 5
2
3 def check_password(password):
4     length_ok = len(password) >= 8
5
6     has_upper = any(char.isupper() for char in password)
7
8     has_lower = any(char.islower() for char in password)
9
10    has_digit = any(char.isdigit() for char in password)
11
12    special_chars = "!@#$%^&*()_+={}|;:,<.>?/~`"
13    has_special = any(char in special_chars for char in
14                      password)
15
16    conditions = {
17        "длина": length_ok,
18        "заглавные буквы": has_upper,
19        "строчные буквы": has_lower,
20        "цифры": has_digit,
21        "специальные символы": has_special
22    }
23
24    if all(conditions.values()):
25        return "Пароль надежный!"
26    else:
27        failed_conditions = [key for key, value in
28                             conditions.items() if not value]
29        return f"Пароль не соответствует требованиям:\n" + "\n"
30               .join(f"- Нет {condition}" for condition in
31                     failed_conditions)
32
33 password = input("Введите пароль для проверки: ")
34 result = check_password(password)
35 print(result)

```

Рисунок 5 – Листинг программы для задания 5

1.6 Задание 6

Год является високосным, если делится на 4 без остатка и не делится на 100 или делится на 400. Проверяем делимость. Создаём функцию visikos с одним аргументом god. После после при помощи операторов сравнения и логики проверяем переменную на удовлетворение условиям, и f строкой выводим результат. На рисунке 6 представлен код программы.

```

1 #Task 6
2
3 def visikos(god):
4     print(f"{god} - високосный{' год' if ((god % 4 == 0 and
5     god % 100 != 0) or god % 400 == 0) else не ' високосный
6     год'}")
7
8 num = int(input('Введите год :'))
9 visikos(num)

```

Рисунок 6 – Листинг программы для задания 6

1.7 Задание 7

Заводим список `a`, который наполняем числами, полученные путём перевода строки, разделённой методом `split()` и по частям переведённой в числа функцией `int()`, оборачивая всё это генератором. В следующей строке задаём минимальное значение по умолчанию, равное первому элементу списка. После при помощи цикла `for` проходимся по списку, начиная со второго элемента при помощи среза `[1:]`. После проверяем тернарным оператором больше ли `i` нашего минимального значения, если меньше то переприсваиваем переменную `minimal`, в ином случае оставляем её неизменной. На рисунке 7 представлен код программы.

```

1 # Task 7
2
3 a = [int(i) for i in input('Введите 3 числа :').split()]
4 minimal = a[0]
5 for i in a[1:]:
6     minimal = int(i) if int(i) < minimal else minimal
7 print(minimal)

```

Рисунок 7 – Листинг программы для задания 7

1.8 Задание 8

Для проверки ошибки открываем блок `try expect`. В блоке `try` создаём переменную `num` и пробуем привести её к целочисленному типу данных, в случае ошибки на этом этапе нас выбрасывает в блок `expect` и выводится сообщение о соответствующей ошибке. Дальше внутри блока `try` заводим переменную `discount` для занесения в неё размера скидки, дальше проверяем принадлежность переменной `num` диапазонам значений при помощи условных операторов, при значении `num` меньше 1000 присваиваем переменной `discount` значение 0, при `num` больше или равно тысячи и меньше пяти тысяч присваиваем переменной скидки значение 5, при `num` больше или равно десяти тысячам делаем `discount` равным 10, при `num` больше десяти тысяч делаем скидку равной 15 процентам, и последний диапазон при `num` меньше или равно нулю, в таком случае мы вызываем ошибку вручную и снова переходим в блок `expect`. Дальше выводим на экран нашу скидку `print()` `om` и `f` строкой. На следующей строке таким же образом рассчитываем по формуле `num*((100-discount)/100)` финальную стоимость и выводим её в нужном формате. На рисунке 8 представлен код программы.

```

1 #Task 8
2
3 try:
4     num = int(input('Введите сумму покупки: '))
5     discount = 0
6     if num <=0:
7         raise
8     if num < 1000:
9         discount = 0
10    elif 1000<=num<5000:
11        discount = 5
12    elif 5000<=num<10000:
13        discount = 10
14    elif num <= 10000:
15        discount = 15
16    print(f'Ваша скидка: {discount}%')
17    print(f'К оплате: {num*((100-discount)/100)}')
18 except:
19    print('Неверный ввод')

```

Рисунок 8 – Листинг программы для задания 8

1.9 Задание 9

Создаём переменную `time` и заносим в неё значение, вводимое пользователем и переведённое в число, выведя при этом сообщение о вводе. Далее через условные операторы проверяем переменную `time` на принадлежность диапазонам, создаваемых каждым своим генератором. Если `time` выходит за границы диапазона времени поднимаем ошибку, приводящее нас в блок `except` и сообщением об ошибке ввода. Далее в зависимости от принадлежности мы выводим сообщение: при `time` принадлежащем `int(i) for i in range(6)` `print('Сейчас ночь')` при `time` принадлежащем `int(i) for i in range(6,12)` `print('Сейчас утро')` при `time` принадлежащем `int(i) for i in range(12,18)` `print('Сейчас день')` при `time` принадлежащем `int(i) for i in range(18,24)` `print('Сейчас вечер')`. На рисунке 9 представлен код программы.

```

1 #Task 9
2 try:
3
4     time = int(input('Введите час -(023): '))
5     if time not in [int(i) for i in range(24)]: raise
6     if time in [int(i) for i in range(6)]:
7         print('Сейчас ночь')
8     elif time in [int(i) for i in range(6,12)]:
9         print('Сейчас утро')
10    elif time in [int(i) for i in range(12,18)]:
11        print('Сейчас день')
12    elif time in [int(i) for i in range(18,24)]:
13        print('Сейчас вечер')
14
15 except:
16    print('неверный ввод')

```

Рисунок 9 – Листинг программы для задания 9

1.10 Задание 10

Сначала определяется функция `is_prime(number)`, которая проверяет, является ли число простым. В первой строке функции проверяется базовый случай: если число равно 0 или 1, оно не является простым, и функция возвращает `False`. Далее в цикле `for` перебираются все числа от 2 до `number-1`, и если находится делитель без остатка, функция немедленно возвращает `False`, указывая на составное число. Если по завершении цикла делителей не найдено, функция возвращает `True`, подтверждая простоту числа. Основная часть программы начинается с блока `try`, где запрашивается ввод пользователя, который преобразуется в целое число. Полученное число передается в функцию `is_prime`, и результат сохраняется в переменной `res`. Затем с помощью `f`-строки выводится форматированный результат, где тернарный оператор выбирает между вариантами "простое число" и "составное число" в зависимости от значения `res`. Если пользователь вводит нечисловые данные, срабатывает блок `except ValueError`, выводящий сообщение "Неверный ввод". На рисунке 10 представлен код программы.

```

1 #Task 10
2
3 def is_prime(number):
4     if number < 2:
5         return False
6     for i in range(2, int(number**0.5) + 1):
7         if number % i == 0:
8             return False
9     return True
10
11 try:
12     num = int(input('Введите число: '))
13     res = is_prime(num)
14     print(f'{num} - простое{" число" if res else составное'
15           ' число}')
16 except ValueError:
17     print('Неверный ввод')

```

Рисунок 10 – Листинг программы для задания 10

Спасибо за понимание !