**Министерство образования и науки Российской Федерации**

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**(ФГБОУ ВО «АмГУ»)**

Факультет математики и информатики

Кафедра информационных и управляющих систем

Направление подготовки 09.03.04 - Программная инженерия

Направленность (профиль) образовательной программы: Программная инженерия

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2**

на тему:Работа с циклами

|  |  |  |
| --- | --- | --- |
| Исполнитель  студент группы 357-об | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | Д.Е. Буханов |
| Проверил | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | Е.В. Дегтярёв |

Благовещенск 2023

**1 ОБЩИЕ ЗАДАНИЯ**

**1 Задание.**

Последовательно вводится ненулевые числа. Определить сумму положительных и сумму отрицательных чисел. Закончить ввод чисел при вводе 0. Для перевода из строки в целое число, использовать класс int().

sum\_number\_pos = 0 # Переменные для хранения значения

sum\_number\_neg = 0 # Переменные для хранения значения

while True:

    number = int(input("Введите число: "))

    if number == 0:

        break

    if number > 0:

        sum\_number\_pos += number

    elif number < 0:

        sum\_number\_neg += number

print(f"Сумма положительных: {sum\_number\_pos}")

print(f"Сумма отрицательных: {sum\_number\_neg}")

Рисунок 1 – Листинг рабочей программы «file\_1.py»

Таблица 1

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 2  -2  -5  6  0 | Сумма положительных: 8  Сумма отрицательных: -7 |
| -3  -4  5  6  0 | Сумма положительных: 11  Сумма отрицательных: -7 |
| 0 | Сумма положительных: 0  Сумма отрицательных: 0 |
| 3  -7  0 | Сумма положительных: 3  Сумма отрицательных: -7 |

**2 Задание.**

При помощи цикла распечатать ряд Фибоначчи: 1 1 2 3 5 8 13 21

seq\_fib = (1, 1, 2, 3, 5, 8, 13, 21) # Ряд Фибоначчи

print("Вывод ряда Фибоначчи")

for i in seq\_fib:

    print(i)

Рисунок 2 – Листинг рабочей программы «file\_2.py»

Таблица 2

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| Кортеж из чисел | Вывод ряда Фибоначчи  1  1  2  3  5  8  13  21 |

**3 Задание.**

Запрашивается 10 чисел (целые значения от 0 до 1000). Опишите алгоритм, позволяющий найти и вывести минимальное значение среди введенных чисел, которые имеют четное значение и не делятся на три.

lst = () # Список для хранения будущих данных

for i in range(10):

    lst += int(input("Введите число: ")),  # Добавляем введенное число в список

lst\_sorted = ()

for i in range(10): # Сортируем список от меньшего к большему

    lst\_sorted += min(lst),

    lst = lst[:lst.index(min(lst))] + lst[lst.index(min(lst)) + 1:]

for i in lst\_sorted:

    if i % 2 == 0 and i % 3 != 0:

        print(f"Вот это число: {i}")

        break # Прекращаем цикл так как было найдено число

else:

    print("Такое число не было введено")

Рисунок 3 – Листинг рабочей программы «file\_3.py»

Таблица 3

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 3  4  5  6  7  8  9  11  22  33 | Вот это число: 4 |
| 0  0  0  0  0  0  0  0  0  0 | Такое число не было введено |

**4 Задание.**

Составить программу для вычисления среднеарифметического N произвольных вводимых чисел.

N = 10

lst\_sum = 0 # Список для хранения будущих данных

for i in range(N):

    lst\_sum += int(input("Введите значение цифры: "))

print(f"Среднее арифметическое: {lst\_sum / N}")

Рисунок 4 – Листинг рабочей программы «file\_4.py»

Таблица 4

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 1  2  3  4  5  6  7  8  9  10 | Среднее арифметическое: 5.5 |

**5 Задание.**

Исправить предыдущее задание (2\_4) для работы со случайными числами.

import random

N = 10 # количество итераций цикла

lst\_sum = 0 # Список для хранения будущих данных

for i in range(N):

    lst\_sum += random.randint(0, 1\_000)

print(f"Среднее арифметическое случайных чисел: {lst\_sum / N}")

Рисунок 5 – Листинг рабочей программы «file\_5.py»

Таблица 5

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| Набор случайных значений | Среднее арифметическое случайных чисел: 431.4 |

**6 Задание.**

Найдите все трехзначные и четырехзначные числа Армстронга.

Числом Армстронга считается натуральное число, сумма цифр которого, возведенных в N-ную степень (N – количество цифр в числе) равна самому числу. Например, 153 = 13+53+33

for i in range(100, 10\_000):

    i = str(i) # Переводим i в строку

    lst\_i = 0

    for ii in i:

        # Возводим каждое число в степень длины строки и суммируем полученные результаты

        lst\_i += int(ii) \*\* len(i)

    if lst\_i == int(i):

        print(f"{i} — это число Армстронга")

Рисунок 6 – Листинг рабочей программы «file\_6.py»

Таблица 6

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
|  | 153 - это число Армстронга  370 - это число Армстронга  371 - это число Армстронга  407 - это число Армстронга  1634 - это число Армстронга  8208 - это число Армстронга  9474 - это число Армстронга |

**7 Задание.**

Напишите программу, которая запрашивает натуральное число N и выводит на экран все автоморфные числа, не превосходящие N.

Автоморфным называется натуральное число, если оно равно последним цифрам своего квадрата. Например, 252 = 625

number\_N = int(input("Введите натуральное число N: "))

if number\_N > 0:

    for i in range(number\_N+1):

        i\_sqr = i \*\* 2 # Квадрат числа i

        if str(i\_sqr).endswith(str(i)): # Проверка на автоморфность

            print(f"Число {i} - автоморфное число. Его квадрат равен {i\_sqr}")

else: # Если введено не верное значение

    # Подымаем исключение о неверном значении

    print("Было введено не натуральное число. Натуральные числа - те числа которые принадлежат диапазону [0; ∞)")

Рисунок 7 – Листинг рабочей программы «file\_7.py»

Таблица 7

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 6 | Число 0 - автоморфное число. Его квадрат равен 0  Число 1 - автоморфное число. Его квадрат равен 1  Число 5 - автоморфное число. Его квадрат равен 25  Число 6 - автоморфное число. Его квадрат равен 36 |

**8 Задание.**

Распечатать дни недели с их порядковыми номерами. Кроме того, рядом выводить выходной ли — это день или рабочий

name\_day = (

    "Понедельник", "Вторник", "Среда",

    "Четверг", "Пятница", "Суббота", "Воскресенье",

) # Название дней

for i in range(7):

    print(f"№ дня: {i + 1}. Название дня: {name\_day[i]}.  \t Это {'выходной день.' if i + 1 > 5 else 'рабочий день.'}")

    # \t - Tab (для более приятного вывода)

Рисунок 8 – Листинг рабочей программы «file\_8.py»

Таблица 8

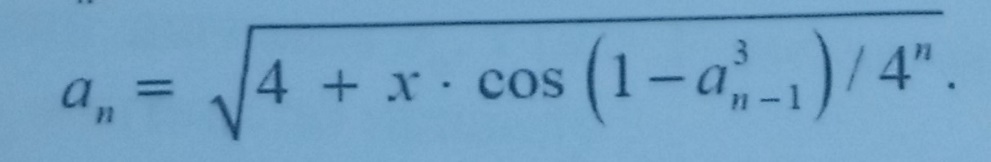
|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| Кортеж из названия дней недели | № дня: 1. Название дня: Понедельник. Это рабочий день.  № дня: 2. Название дня: Вторник. Это рабочий день.  № дня: 3. Название дня: Среда. Это рабочий день.  № дня: 4. Название дня: Четверг. Это рабочий день.  № дня: 5. Название дня: Пятница. Это рабочий день.  № дня: 6. Название дня: Суббота. Это выходной день.  № дня: 7. Название дня: Воскресенье. Это выходной день. |

**2 ИНДИВИДУЛЬНОЕ ЗАДАНИЕ**

**2.4 Задания.**

Вычислить предел последовательности an с точностью 0.5 \* 10-6, если значения a0,x (x > 0) вводятся.

Значения an считать по правилу



import math

x = float(input('Введите x: '))

if x <= 1:

    print("Были введены не верные данные")

else:

    n = 0 # Количество итераций цикла

    # Точность предела последовательности

    e =  0.0000005

    # Прошлый элемент последовательности

    a\_past = float(input('Введите a0: '))

    while True:

        n += 1

        # Новый элемент последовательности

        a\_next = 1 + math.sin(1 + (a\_past \*\* 2)) / x \*\* n

        if abs(a\_past - a\_next) < e:

            print(f"Найденный предел последовательности: {a\_next}")

            break

        a\_past = a\_next

Рисунок 9 – Листинг рабочей программы «main.py»

Таблица 9

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 2  5 | Найденный предел последовательности: 1.0000004335864567 |
| 4  9 | Найденный предел последовательности: 1.0000000541983394 |



Блок схема 1

**2.5 Задание.**

Дано натуральное число. Вычислить количество цифр, кратных 4.

number = input("Введите число: ")

flag = 0

for i in number:

    if int(i) % 4 == 0:

        flag += 1

print(f"Количество цифр кратных 4: {flag}")

Рисунок 10 – Листинг рабочей программы «main\_2.py»

Таблица 10

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| 1234567890 | Количество цифр кратных 4: 3 |



Блок схема 2

**2.6 Задание.**

Составить блок-схему и программу для запыления таблицы значений функции y = f(x) на отрезке с указанным шагом измерения аргумента. Значение функции выводить с точностью до тысячных долей.

import math

print("-" \* 60)

print("!{: ^28}!".format('X'), "{: ^28}!".format('y = f(x)'))

print("-" \* 60)

for i in range(-500, 1000, 55):

    i /= 100

    if i < -1:

        print(

            "!{: ^28}!".format(i),

            "{: ^29}!".format(

                "{: .3f}".format(i \*\* 2)

            )

        )

    if 0 <= i <= 12.5:

        print(

            "!{: ^28}!".format(i),

            "{: ^29}!".format(

                "{: .3f}".format((math.e \*\* i) + 5 + math.cos(0.001 \* i))

            )

        )

Рисунок 11 – Листинг рабочей программы «main\_3.py»

Таблица 10

|  |  |
| --- | --- |
| Данные: | |
| Входные (ввод) | Выходные (вывод) |
| Константы |  |



Блок схема 3