**Лабораторная работа №5  
Тема: Циклический алгоритм**

### **Задача 1. Условие.**

Программа для нахождения максимального значения функции Cos(x)/Sin(x) на отрезке a,ba,ba,b с шагом h.

**Таблица данных.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс** | **Имя** | **Смысл** | **Тип, структура** | **Ограничения на значения** |
| Входные данные | a, b | Начало и конец отрезка | вещественные | a < b |
| Входные данные | h | Шаг изменения x | вещественное | h > 0 |
| Промежуточные данные | x | Текущее значение аргумента | вещественное |  |
| Промежуточные данные | value | Значение функции при текущем x | вещественное |  |
| Выходные данные | max\_value | Максимальное значение функции на заданном отрезке | вещественное |  |

**Тесты.**

|  |  |  |
| --- | --- | --- |
| **Класс** | **Имя** | **Значение** |
| Входные данные | a, b | 0, π |
| Входные данные | h | 0.1 |
| Выходные данные | max\_value | 1.00 |

**Листинг программы:**

a = float(input("Введите начало отрезка a: "))  
b = float(input("Введите конец отрезка b: "))  
h = float(input("Введите шаг h: "))  
max\_value = -math.inf  
x = a

while x <= b:  
value = math.cos(x) / math.sin(x)  
if value > max\_value:  
max\_value = value  
x += h

print(f"Максимальное значение функции Cos(x)/Sin(x) на отрезке [{a}, {b}] с шагом {h}: {max\_value:.2f}")

### **Задача 2. Условие.**

Программа для нахождения количества положительных чисел в последовательности.

**Таблица данных.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс** | **Имя** | **Смысл** | **Тип, структура** | **Ограничения на значения** |
| Входные данные | N | Количество элементов последовательности | целое | N > 0 |
| Промежуточные данные | number | Текущее число последовательности | вещественное |  |
| Выходные данные | count\_positive | Количество положительных чисел | целое |  |

**Тесты.**

|  |  |  |
| --- | --- | --- |
| **Класс** | **Имя** | **Значение** |
| Входные данные | N | 5 |
| Входные данные | Числа | [-1, 2, -3, 4, 5] |
| Выходные данные | count\_positive | 3 |

**Листинг программы:**

N = int(input("Введите количество элементов последовательности: "))  
count\_positive = 0  
i = 1

while i <= N:  
number = float(input(f"Введите число {i}: "))  
if number > 0:  
count\_positive += 1  
i += 1

print(f"Количество положительных чисел: {count\_positive}")

### **Задача 3. Условие.**

Программа для нахождения количества положительных и отрицательных чисел в последовательности.

**Таблица данных.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс** | **Имя** | **Смысл** | **Тип, структура** | **Ограничения на значения** |
| Входные данные | N | Количество элементов последовательности | целое | N > 0 |
| Промежуточные данные | number | Текущее число последовательности | вещественное |  |
| Выходные данные | positive\_count | Количество положительных чисел | целое |  |
| Выходные данные | negative\_count | Количество отрицательных чисел | целое |  |

**Тесты.**

|  |  |  |
| --- | --- | --- |
| **Класс** | **Имя** | **Значение** |
| Входные данные | N | 6 |
| Входные данные | Числа | [1, -2, 3, -4, 5, -6] |
| Выходные данные | positive\_count | 3 |
| Выходные данные | negative\_count | 3 |

**Листинг программы:**

N = int(input("Введите количество элементов последовательности: "))  
positive\_count = 0  
negative\_count = 0  
i = 1

while i <= N:  
number = float(input(f"Введите число {i}: "))  
if number > 0:  
positive\_count += 1  
elif number < 0:  
negative\_count += 1  
i += 1

if positive\_count > negative\_count:  
print("Положительных чисел больше")  
elif negative\_count > positive\_count:  
print("Отрицательных чисел больше")  
else:  
print("Количество положительных и отрицательных чисел равно")

### **Задача 4. Условие.**

Программа для нахождения суммы положительных и отрицательных чисел в последовательности.

**Таблица данных.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс** | **Имя** | **Смысл** | **Тип, структура** | **Ограничения на значения** |
| Входные данные | N | Количество элементов последовательности | целое | N > 0 |
| Промежуточные данные | number | Текущее число последовательности | вещественное |  |
| Выходные данные | positive\_sum | Сумма положительных чисел | вещественное |  |
| Выходные данные | negative\_sum | Сумма отрицательных чисел | вещественное |  |

**Тесты.**

|  |  |  |
| --- | --- | --- |
| **Класс** | **Имя** | **Значение** |
| Входные данные | N | 4 |
| Входные данные | Числа | [-1, 2, -3, 4] |
| Выходные данные | positive\_sum | 6 |
| Выходные данные | negative\_sum | -4 |

**Листинг программы:**

N = int(input("Введите количество элементов последовательности: "))  
positive\_sum = 0  
negative\_sum = 0  
i = 1

while i <= N:  
number = float(input(f"Введите число {i}: "))  
if number > 0:  
positive\_sum += number  
elif number < 0:  
negative\_sum += number  
i += 1

print(f"Сумма положительных чисел: {positive\_sum}")  
print(f"Сумма отрицательных чисел: {negative\_sum}")

### **Задача 5. Условие.**

Программа для вычисления суммы положительных чисел до первого отрицательного.

**Таблица данных.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс** | **Имя** | **Смысл** | **Тип, структура** | **Ограничения на значения** |
| Входные данные | number | Введенное число | целое |  |
| Выходные данные | sum\_positive | Сумма положительных чисел | целое |  |

**Тесты.**

|  |  |  |
| --- | --- | --- |
| **Класс** | **Имя** | **Значение** |
| Входные данные | Числа | [1, 2, 3, -1] |
| Выходные данные | sum\_positive | 6 |

**Листинг программы:**

sum\_positive = 0

while True:  
number = int(input("Введите целое число: "))  
if number < 0:  
break  
sum\_positive += number

print(f"Сумма положительных чисел до первого отрицательного: {sum\_positive}")

### **Задача 6. Условие.**

Программа для вычисления суммы всех ненулевых чисел.

**Таблица данных.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс** | **Имя** | **Смысл** | **Тип, структура** | **Ограничения на значения** |
| Входные данные | number | Введенное число | целое |  |
| Выходные данные | sum\_non\_zero | Сумма всех ненулевых чисел | целое |  |

**Тесты.**

|  |  |  |
| --- | --- | --- |
| **Класс** | **Имя** | **Значение** |
| Входные данные | Числа | [1, 2, -3, 4, 0] |
| Выходные данные | sum\_non\_zero | 4 |

**Листинг программы:**

sum\_non\_zero = 0

while True:  
number = int(input("Введите целое число: "))  
if number == 0:  
break  
sum\_non\_zero += number

print(f"Сумма всех ненулевых чисел: {sum\_non\_zero}")

### **Задача 7. Условие.**

Программа для вычисления количества чисел, которые меньше заданного.

**Таблица данных.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс** | **Имя** | **Смысл** | **Тип, структура** | **Ограничения на значения** |
| Входные данные | threshold | Пороговое значение | целое |  |
| Входные данные | number | Введенное число | целое |  |
| Выходные данные | count | Количество чисел, меньших порогового | целое |  |

**Тесты.**

|  |  |  |
| --- | --- | --- |
| **Класс** | **Имя** | **Значение** |
| Входные данные | threshold | 5 |
| Входные данные | Числа | [1, 2, 6, 4, 7] |
| Выходные данные | count | 3 |

**Листинг программы:**

threshold = int(input("Введите пороговое значение: "))  
count = 0

while True:  
number = int(input("Введите число (0 для выхода): "))  
if number == 0:  
break  
if number < threshold:  
count += 1

print(f"Количество чисел, меньше {threshold}: {count}")

### **Задача 8. Условие.**

Программа для расчета срока, за который вклад увеличится вдвое при заданной процентной ставке.

**Таблица данных.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс** | **Имя** | **Смысл** | **Тип, структура** | **Ограничения на значения** |
| Входные данные | initial\_deposit | Начальный вклад | вещественное | initial\_deposit > 0 |
| Входные данные | rate | Процентная ставка | вещественное | rate > 0 |
| Промежуточные данные | current\_deposit | Текущий размер вклада | вещественное |  |
| Выходные данные | years | Количество лет до удвоения вклада | целое |  |

**Тесты.**

|  |  |  |
| --- | --- | --- |
| **Класс** | **Имя** | **Значение** |
| Входные данные | initial\_deposit | 1000 |
| Входные данные | rate | 5 |
| Выходные данные | years | 15 |

**Листинг программы:**

initial\_deposit = float(input("Введите начальный вклад: "))  
rate = float(input("Введите процентную ставку: "))  
years = 0  
current\_deposit = initial\_deposit

while current\_deposit < 2 \* initial\_deposit:  
current\_deposit += current\_deposit \* rate / 100  
years += 1

print(f"Вклад удвоится через {years} лет при ставке {rate}%")

### **Задача 9. Условие.**

Программа для вычисления суммы чисел на отрезке от 10 до 20.

**Таблица данных.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс** | **Имя** | **Смысл** | **Тип, структура** | **Ограничения на значения** |
| Входные данные | number | Число в заданном диапазоне | целое |  |
| Выходные данные | sum\_segment | Сумма чисел в диапазоне | целое |  |

**Тесты.**

|  |  |  |
| --- | --- | --- |
| **Класс** | **Имя** | **Значение** |
| Входные данные | Диапазон | [10, 20] |
| Выходные данные | sum\_segment | 165 |

**Листинг программы:**

sum\_segment = 0

for number in range(10, 21):  
sum\_segment += number

print(f"Сумма чисел на отрезке от 10 до 20: {sum\_segment}")

### **Задача 10. Условие.**

Программа для расчета идеального веса для взрослого человека по формуле: Идеальный вес = Рост - 100.

**Таблица данных.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс** | **Имя** | **Смысл** | **Тип, структура** | **Ограничения на значения** |
| Входные данные | height | Рост человека в сантиметрах | целое |  |
| Выходные данные | ideal\_weight | Идеальный вес | целое |  |

**Тесты.**

|  |  |  |
| --- | --- | --- |
| **Класс** | **Имя** | **Значение** |
| Входные данные | height | 170 |
| Выходные данные | ideal\_weight | 70 |

**Листинг программы:**

height = int(input("Введите рост (в см): "))  
ideal\_weight = height - 100

while height != 250:  
height += 1

print(f"Идеальный вес: {ideal\_weight} кг")