

Студент группы ИС-26 Байрамов Д.Ю.

Практическое занятие № 5

Тема: Составление программ циклической структуры в IDE PyCharm Community.

Цель: Закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составления программ циклической структуры в IDE PyCharm Community.

Постановка задачи 1:

1. **Входные данные:** Два целых числа AA и BB , где $A < B$ и $A < B$.
2. **Выходные данные:** Последовательность чисел от $B-1$ до $A+1$ в порядке убывания, а также количество таких чисел MN .
3. **Тип алгоритма:** Циклический.

Текст программы:

```
def descending_numbers(a, b):  
    """  
    Вывод чисел от B-1 до A+1 в порядке убывания и их количества.  
    :param a: int, меньшее число  
    :param b: int, большее число  
    :return: tuple (list, int), список чисел и их количество  
    """  
    try:  
        # Проверяем корректность входных данных  
        if not isinstance(a, int) or not isinstance(b, int):  
            raise ValueError("Оба числа должны быть целыми.")  
        if a >= b:  
            raise ValueError("Число A должно быть меньше числа B.")  
        # Генерация списка чисел в порядке убывания  
        numbers = list(range(b - 1, a, -1))  
        return numbers, len(numbers)  
    except Exception as e:  
        print(f"Ошибка: {e}")  
        return [], 0  
  
if __name__ == "__main__":
```

```

try:
    a = int(input("Введите число A: "))
    b = int(input("Введите число B: "))
    numbers, count = descending_numbers(a, b)
    print(f"Числа: {numbers}")
    print(f"Количество чисел: {count}")
except ValueError:
    print("Ошибка: Введите целые числа.")

```

Протокол работы программы:

Пример 1

Вход: $A=2, B=8$ $A=2, B=8$

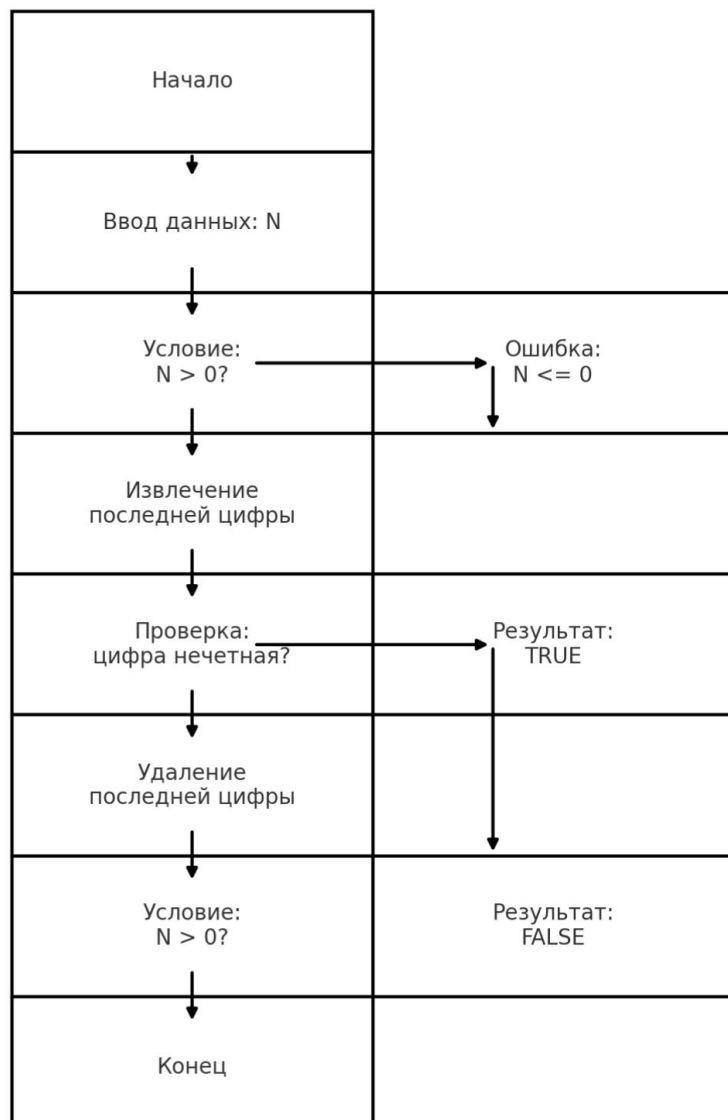
Выход: Числа: [7, 6, 5, 4, 3]

Количество чисел: 5

Постановка задачи 2:

1. **Входные данные:** Целое число $N > 0$.
2. **Выходные данные:** Логическое значение:
 - a. True, если в записи числа N есть хотя бы одна нечетная цифра.
 - b. False, если все цифры четные.
3. **Тип алгоритма:** Линейный.

Блок-схема алгоритма:



Текст программы:

```
def has_odd_digit(n):
```

```
    """
```

Проверка на наличие нечетных цифр в числе.

:param n: int, положительное число

:return: bool, True, если есть нечетные цифры, иначе False

```
    """
```

```
    try:
```

```
        # Проверяем корректность входных данных
```

```
        if not isinstance(n, int) or n <= 0:
```

```
            raise ValueError("Число должно быть целым и положительным.")
```

```
        # Проверка наличия нечетных цифр
```

```
        while n > 0:
```

```
            digit = n % 10 # Последняя цифра числа
```

```
            if digit % 2 != 0:
```

```

        return True
    n //= 10 # Удаляем последнюю цифру
    return False
except Exception as e:
    print(f"Ошибка: {e}")
    return False

if __name__ == "__main__":
    try:
        n = int(input("Введите положительное число N: "))
        result = has_odd_digit(n)
        print(f"Результат: {'TRUE' if result else 'FALSE'}")
    except ValueError:
        print("Ошибка: Введите целое положительное число.")

```

Протокол работы программы:

Пример 1

Вход: $N=2468$ $N=2468$

Выход: Результат: FALSE

Пример 2

Вход: $N=1234$ $N=1234$

Выход: Результат: TRUE

Вывод:

В ходе выполнения практического занятия были успешно решены задачи с использованием циклов, условных операторов и обработки исключений. Были разработаны программы для:

1. Генерации последовательности чисел в порядке убывания между заданными границами и подсчета их количества.
2. Определения наличия нечетных цифр в записи заданного числа.

Обе программы соответствуют требованиям ПЕР 8, включают проверку корректности входных данных и обработку исключений. Также были выполнены отладка и тестирование программного кода.

Код и отчет, содержащий постановку задач, текст программ, протокол работы и примеры выполнения, подготовлены для проверки.