Лекционное задание №1

Задача 7b. Обоснуйте, а потом напишите, алгоритм добавление к числу в троичной записи тройки.

$3_{10} = 10_3$

Примеры:

1) 111 + 10 = 121
2) 10202 + 10 = 10212
3) 120 + 10 = 200
4) 222 + 10 = 1002
5) 21 + 10 = 101

Как видно из примеров, при прибавлении 3 в троичной записи к числу, мы оставляем неизменным младший разряд, к следующему после него прибавляем 1. Как видно из примеров 3-5, если во втором разряде числа 2, при прибавлении 1 к нему мы получим 3, то есть 10. Таким образом, на место этого разряда нам следует записать 0, и прибавить 1 к следующему (старшему) разряду. Повторять такое действие с каждым старшим разрядом нам следует до тех пор, пока в нем не окажется цифра меньшая 2, или пока не кончатся разряды (из примеров 4-5).

Сформулируем алгоритм. Пусть мы храним число a в виде вектора (массива) цифр, обозначающих разряды числа начиная с младшего.

То есть число $a = \overline{a_n a_{n-1} \dots a_1 a_0}$ мы сохраним в виде массива $[a_0, a_1, \dots, a_n]$. Так как число записано в троичной записи, для каждой цифры $a_i \in \{0, 1, 2\}$.

```
k := 1
ПОКА (k < n) И (a[k] == 2):
     a[k] := 0
     k := k + 1

ЕСЛИ k >= n:
     добавить в конец a[k] := 1
ИНАЧЕ:
     a[k] := a[k] + 1
```

Инвариант цикла:

Разряды $a[1]$, ..., $a[k-1]$ обнулены (изначально были равны 2), и осталось прибавить 1 к разряду с индексом k.

k — индекс разряда, к которому нужно прибавить 1

$a[1..k-1]$ — уже обработаны (обнулены из-за переноса)

$a[k..n-1]$ — ещё не изменены

Пусть A_0 — исходное значение числа. После обработки разрядов $1..k-1$:

$$A_i = A_0 + 3 - 3^k$$

То есть мы "должны" ещё 3^k, что эквивалентно прибавлению 1 к разряду k.

Доказательство корректности:

(по индукции)

<u>База (k = 1)</u>: Инвариант выполняется: ничего не изменено, нужно прибавить 3 к числу, начиная с разряда 1.

<u>Шаг индукции</u>: Пусть инвариант верен для k. Если $a[k] = 2$, устанавливаем $a[k] := 0$. Это уменьшает число на $2 \cdot 3^k$, но мы должны были прибавить 3^k $-2 \cdot 3^k + 3^k = -3^k$, значит теперь должны прибавить 3^{k+1} (перенос)

$k := k + 1$, инвариант сохраняется.

Цикл завершается когда $k >= n$ или $a[k] \neq 2$:

Если $a[k] < 2$: выполняем $a[k] := a[k] + 1$, прибавляем недостающие 3^k

Если $k >= n$: добавляем новый разряд $a[n] := 1$, что эквивалентно $+3^n$

Реализация алгоритма на ЯП Python:

```
def add_three(a: list[int]) -> list[int]:
    n = len(a)
    if n == 1:
        a.append(0)
        n = 2
```

```
    k = 1
    while k < n and a[k] == 2:
        a[k] = 0
        k += 1
    if k >= n:
        a.append(1)
    else:
        a[k] = a[k] + 1
    return a

if __name__ == "__main__":
    tests = [
        "111",       # Общий случай
        "10202",     # Общий случай
        "120",       # Один перенос
        "222",       # Каскадный перенос
        "21",        # Перенос с расширением
        "0",         # Минимальное число
        "2",         # Однозначное число
    ]

    all_passed = True
    for input_str in tests:
        a = [int(c) for c in reversed(input_str)]
        add_three(a)
        result = ''.join(map(str, reversed(a)))
        print(f"{input_str} + 10 = {result}")
```

Результаты тестирования:

```
111 + 10 = 121
10202 + 10 = 10212
120 + 10 = 200
222 + 10 = 1002
21 + 10 = 101
0 + 10 = 10
2 + 10 = 12
```

Результаты совпали с ожидаемыми. Алгоритм корректен.