**Методы реализации хранения и обработки больших чисел**

1. Массив фиксированного основания (лимбов)

Описание: Число разбивается на "лимбы" — блоки фиксированной длины (обычно 32 или 64 бита), хранящиеся в массиве.

Структура:

typedef uint64\_t mp\_limb\_t;

typedef struct {

int \_mp\_alloc; - количество выделенных лимбов

int \_mp\_size; - количество используемых лимбов

mp\_limb\_t \*\_mp\_d; - указатель на массив лимбов

} \_\_mpz\_struct;

Пример. Число 12345678901234567890 в 64-битных лимбах:

mp\_limb\_t limbs[2] = {

0xEB1F0AD2,

0xAB54A98C

};

Арифметика

1. Массив десятичных цифр

Описание: Хранит каждую десятичную цифру в массиве (uint8\_t, char)

Пример (C):

char digits[] = {1, 2, 3, 4, 5};

Арифметика:

1. BCD (Binary-Coded Decimal)

Описание: Каждая цифра кодируется 4 битами (0–9 → 0000–1001)

Пример (C):

uint8\_t bcd = 0x94; // число 94: 1001 0100

Арифметика:

Требует особых правил (корректировка после сложения)

1. Связный список/дерево

Описание: Каждая часть числа — отдельный узел списка или дерева

Пример:

typedef struct Node {

int chunk;

struct Node\* next;

} Node;

Арифметика:

1. Полиномиальное представление

Описание: Представление числа как многочлена (часто в двоичной форме):

1011 = x^3 + x + 1

Пример (C):

uint32\_t poly = 0b1011; // x^3 + x + 1

Арифметика:

1. Мантисса + экспонента (BigFloat)

Описание: Число в виде mantissa × base^exponent

Пример (структура):

typedef struct {

BigInt mantissa;

int exponent;

} BigFloat;

Арифметика: