Типы данных

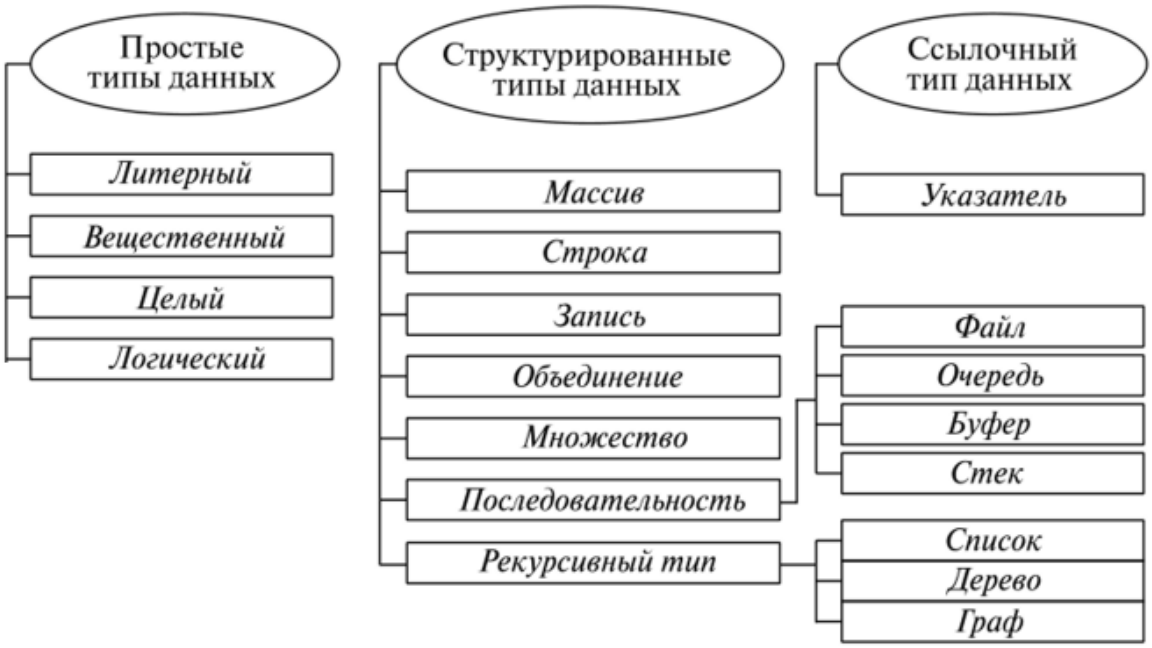
Тип данных — это классификация, определяющая множество допустимых значений, которые может принимать объект (переменная, константа, выражение), а также набор операций, применимых к этим значениям. Типы данных обеспечивают формализацию представления информации, контроль корректности операций и эффективное использование памяти.

К основным категориям типов данных относятся:

Простые (скалярные) типы — содержат единственное значение в каждый момент времени.

- Литерный (символьный) — для представления отдельных символов.
- Вещественный (дробный) — для чисел с плавающей точкой.
- Целый — для целочисленных значений.
- Логический — для значений истинности (true/false).

Структурированные (составные) типы данных — организуют несколько значений (возможно, различных типов) в единую логическую структуру. Они позволяют управлять группами данных как целым, обеспечивая иерархическую организацию информации.



Простые типы данных

- Простые (базовые) типы данных
- Целый (integer): принимает целые значения, например, -3, 0, 25.
- Вещественный (real, float): поддерживает дробные числа, например, 3.14, -0.001.
- Логический (boolean): два значения: True (истина), False (ложь).
- Символьный (char): одиночный символ, например, 'a', '5', '@'.
- Строковый (string): последовательность символов, например, «Привет», «2024».
- Составные (структурные) типы данных

- Массив (array): набор элементов одного типа (например, список оценок).
- Список (list): аналог массива, может изменяться по длине.
- Кортеж (tuple): неизменяемая последовательность элементов.
- Словарь (dict, map): набор пар «ключ–значение».

Переменная имеет определенный тип. И этот тип определяет, какие значения может иметь переменная и сколько байт в памяти она будет занимать. Определены следующие базовые типы данных:

char: представляет один символ. Занимает в памяти 1 байт (8 бит).

unsigned char: представляет один символ. Занимает в памяти 1 байт (8 бит). Может хранить любой значение из диапазона от 0 до 255

signed char: представляет один символ. Занимает в памяти 1 байт (8 бит). Может хранить любое значение из диапазона от -128 до 127

short: представляет целое число в диапазоне от –32768 до 32767. Занимает в памяти 2 байта (16 бит).

Имеет псевдонимы **short int, signed short и signed short int.**

unsigned short: представляет целое число в диапазоне от 0 до 65535. Занимает в памяти 2 байта (16 бит).

Имеет псевдоним **unsigned short int.**

int: представляет целое число. В зависимости от архитектуры процессора может занимать 2 байта (16 бит) или 4 байта (32 бита). Если брать основные платформы - 64-разрядные Windows, Linux (вместе с Android) и MacOS, то размер int составляет 4 байта. Диапазон предельных значений соответственно также может варьироваться от –32768 до 32767 (при 2 байтах) или от −2 147 483 648 до 2 147 483 647 (при 4 байтах) и выше.

Имеет псевдонимы **signed int и signed**

unsigned int: представляет положительное целое число. В зависимости от архитектуры процессора может занимать 2 байта (16 бит) или 4 байта (32 бита), и из-за этого диапазон предельных значений может меняться: от 0 до 65535 (для 2 байт), либо от 0 до 4 294 967 295 (для 4 байт).

Имеет псевдоним **unsigned:** то же самое, что и **unsigned int**

long: представляет целое число и занимает в памяти 4 байта (32 бита) или 8 байт (64 бита). В зависимости от размера может находиться в в диапазоне от -2 147 483 648 до 2 147 483 647 (4 байта), либо в диапазоне от -9223372036854775807 до +9 223 372 036 854 775 807 (8 байт).

Если брать распространенные платформы, то на 64-разрядном Windows long занимает 4 байта, а на 64-разрядных Linux/MacOS - 8 байт.

Имеет псевдонимы **long int, signed long int и signed long.**

unsigned long: представляет целое число и занимает в памяти 4 байта (32 бита) или 8 байт (64 бита). В зависимости от размера может находиться в в диапазоне от 0 до 4 294 967 295 (4 байта) или в диапазоне от 0 до 18 446 744 073 709 551 615 (8 байт).

Имеет псевдоним **unsigned long int.**

long long: представляет целое число в диапазоне от -9223372036854775807 до +9 223 372 036 854 775 807. Занимает в памяти, как правило, 8 байт (64 бита).

Имеет псевдонимы **long long int, signed long long int и signed long long.**

unsigned long long: представляет целое число в диапазоне от 0 до 18 446 744 073 709 551 615. Занимает в памяти, как правило, 8 байт (64 бита).

Имеет псевдоним **unsigned long long int.**

float: представляет вещественное число одинарной точности с плавающей точкой в диапазоне +/- 3.4E-38 до 3.4E+38. В памяти занимает 4 байта (32 бита)

double: представляет вещественное число двойной точности с плавающей точкой в диапазоне +/- 1.7E-308 до 1.7E+308. В памяти занимает 8 байт (64 бита)

long double: представляет вещественное число двойной точности с плавающей точкой в диапазоне +/- 3.4E-4932 до 1.1E+4932. В памяти занимает 10 байт (80 бит). На некоторых системах может занимать 96 и 128 бит.

void: тип без значения

Пример:

```
int age = 38;

signed int number = 2;

signed temps = -3;
```

Структурированные типы данных

Структурированный тип данных — это тип, объединяющий несколько элементов (компонентов), каждый из которых может иметь собственный тип. Элементы могут быть как простыми, так и другими структурированными типами. Основные цели использования.

Примеры структурированных типов:

- Массив — упорядоченная коллекция элементов одного типа, доступ к которым осуществляется по индексу.
- Строка — последовательность символов, часто рассматриваемая как массив литерного типа.
- Запись (структура) — совокупность именованных полей различных типов.
- Объединение (вариантная запись) — структура, содержащая несколько полей, но в каждый момент времени используется только одно из них.
- Множество — неупорядоченная коллекция уникальных элементов определённого базового типа.
- Последовательность (список, поток) — линейная структура с динамическим размером, часто реализуемая через указатели.
- Рекурсивный тип — тип, содержащий в своём определении ссылку на себя (например, узлы деревьев, списков).

  Нередко возникает необходимость работы не с одиночными данными, а с наборами данных. И для этого в языке применяются массивы. Массив представляет набор однотипных значений. Объявление массива выглядит следующим образом:

тип_переменной название_массива [длина_массива]

Язык C#

```
string[] stringarr;
stringarr = new string[3] {"Element 1", "Element 2", "Element 3"};
```

Язык C++

```
int numbers[4]{1,2,3,4};
int first = numbers[0];     // получаем первый элемент
```

Язык C

```
int numbers[4];

numbers[0] = 1;
numbers[1] = 2;
numbers[2] = 3;
numbers[3] = 4;
```

Структурированные типы данных являются фундаментом для построения сложных информационных моделей в программировании. Они позволяют эффективно организовывать, хранить и обрабатывать данные, обеспечивая гибкость и масштабируемость программных систем. Правильный выбор структуры данных напрямую влияет на производительность и корректность алгоритмов.

Пример программы с использованием простых и структурированных типов на языке C

```c
int main()
{
    // Простая переменная
    int number = 10;

    // Простой массив
    int numbers[5] = {1, 2, 3, 4, 5};

    // Выводим значение переменной
    printf("Значение переменной: %d\n", number);

    // Выводим все элементы массива
    printf("Элементы массива: ");
    for(int i = 0; i < 5; i++)
    {
        printf("%d ", numbers[i]);
    }
    printf("\n");

    // Меняем значение переменной
    number = 20;
    printf("Новое значение переменной: %d\n", number);

    // Меняем один элемент массива
    numbers[2] = 99;
    printf("Массив после изменения: ");
    for(int i = 0; i < 5; i++)
    {
        printf("%d ", numbers[i]);
    }
    printf("\n");

    return 0;
}
```