В.В. Подбельский

Иллюстрации к курсу лекций по дисциплине «Программирование на C#»

02

Данные и Типы

Благодарность Д.Д. Сагалову за помощь в переработке презентаций

Программа

Программа – это запись алгоритма на языке исполнителя.

Алгоритмы + Структуры Данных = Программы

Niklaus Wirth (1976)

Определение ISO/IEC 2382:2015:
Данные – поддающееся многократной интерпретации представление информации в формализованном виде, пригодном для передачи, связи или обработки.

Разработчики Теории Типов

Математики:

Рассел

Тьюринг

Черч

. . . .

Программисты:

Хоар

Брогстол

Дейкстра

. . . .

Тип Данных. Определения

IEEE Std 1320.2-1998:

Тип данных (тип) – множество значений и операций над этими значениями.

ISO/IEC/IEEE 24765-2010:

Тип данных – класс данных, характеризуемый членами класса и операциями, которые могут быть к ним применены.

Важно: тип данных характеризуется *одновременно*:

- Множеством допустимых значений, принимаемых данными этого типа;
- Набор операций, которые можно осуществлять над данными, принадлежащими к этому типу.

О типах данных на Википедии: https://ru.wikipedia.org/wiki/Тип_данных

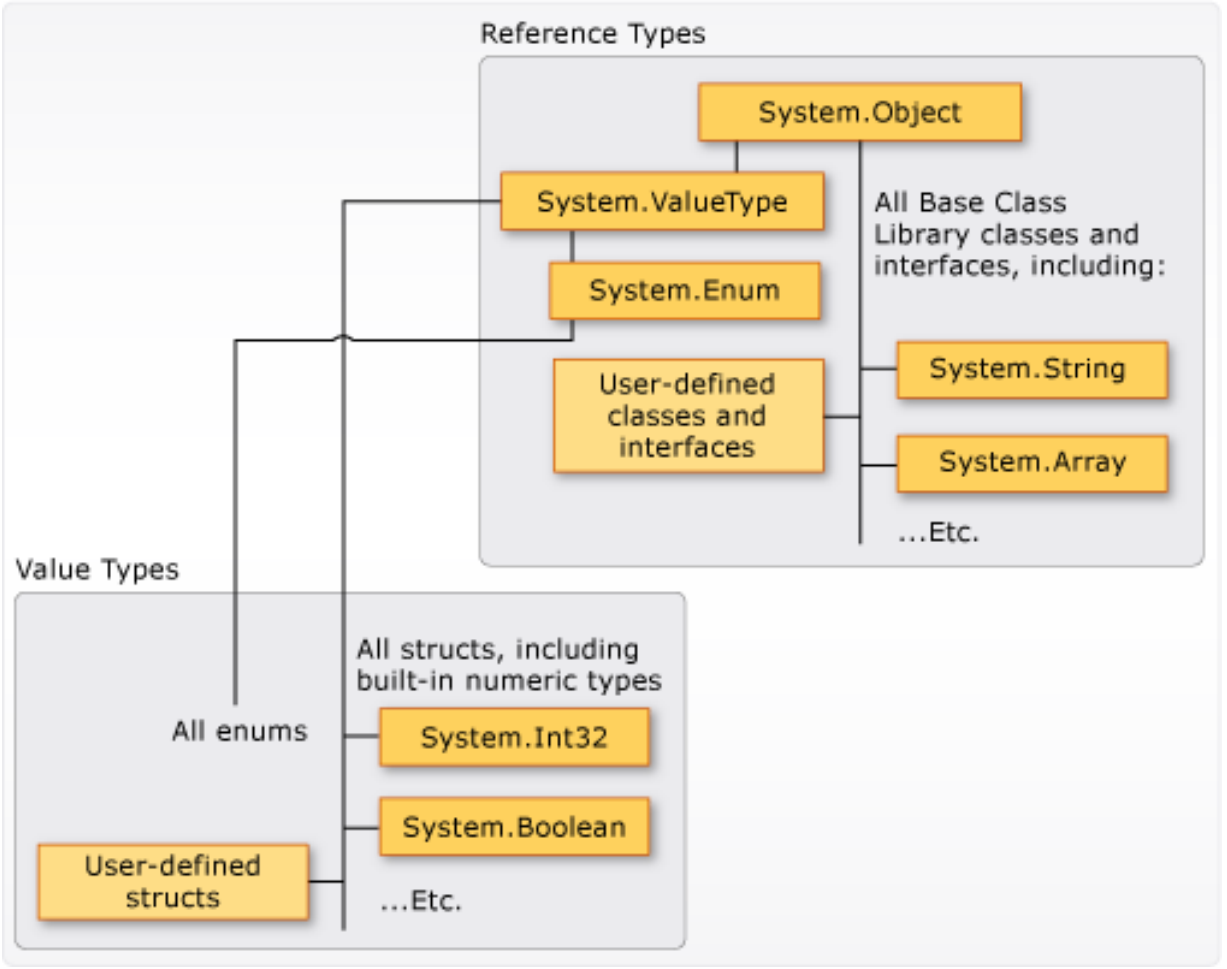
Типы Данных в Языках Программирования

Базисные (примитивные) типы:

1) Целочисленные;
2) Вещественные;
3) Логический;
4) Символьный.

Структурированные типы:

1) Массив;
2) Класс/Запись;
3) Структура;
4) Перечисление.

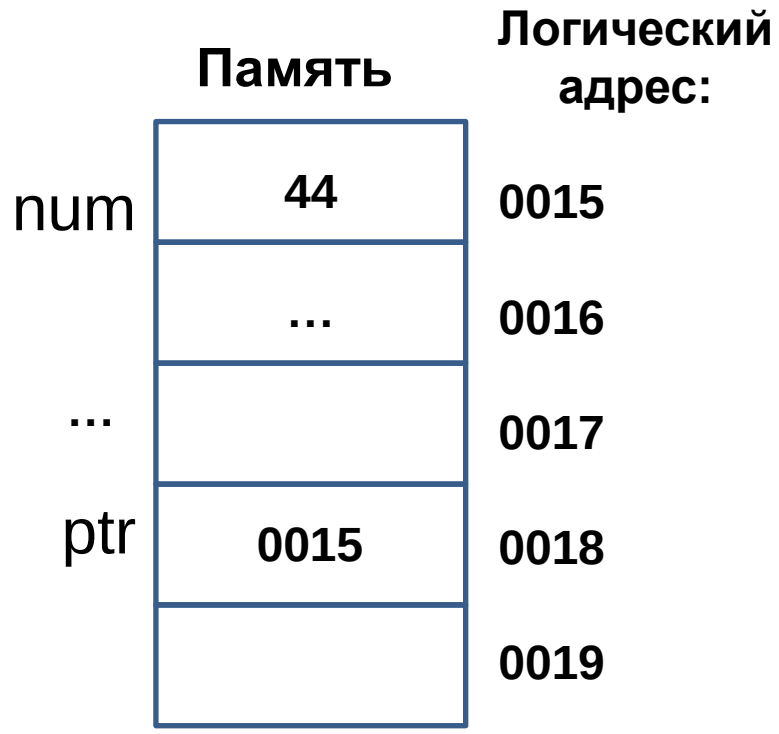


Адреса и Указатели

```
using System;
// Указатели допустимы только в небезопасном
// контексте с флагом компиляции /unsafe
unsafe
{
    int num = 44;
    // ptr хранит адрес переменной num.
    int* ptr = &num;
    // Выводится адрес num в памяти (CE)
    Console.Write(ptr + " ");

    Console.Write(*ptr);
}
```

Результат: 0015 44



О Безымянных Типах

Типы массивов:

int [ ] integers; // Тип **int** [ ]

double [ ] reals; // Тип **double**[ ]

Свойства Типа

1) Определяет множество значений;
2) Любое значение принадлежит только одному типу;
3) Тип выражения определяется без вычислений;
4) Для операции типы операндов и результата фиксированы;
5) Операция может быть применима к разным типам;
6) Операции и свойства значений определены аксиоматично.

Пример к свойству 5:

- 5 + 2 равно 7 // Сумма целых чисел.
- "5" + "2" равно "52" // Конкатенация строк.

Примеры к свойству 6:

- 5 / 2 равно 2 // Результат деления целых чисел – целое.
- 5.0 / 2.0 равно 2.5 // Результат деления вещественных – вещественное.

Роль Типов на Этапе Компиляции

Статический анализ:

```
int a = 4.3;         // Ошибка компиляции – недопустимое неявное преобразование.
sbyte sb = -12;      // 8 бит, константа этапа компиляции, вмещается в тип.
short sh = sb*24; // Ошибка компиляции – sb*24 имеет тип int.
```

Обратите внимание: по умолчанию при целочисленных вычислениях результат имеет минимальный по объёму памяти размер, а его тип является одним из следующих (по убыванию приоритетов):
int → uint → long → ulong.

Именно по этой причине 3 строка примера выдаёт ошибку.

Важнейшие Операции

Простейшие операции:

- Сравнение значений: >, >=, <, <=;
- Присваивание: =, +=, -=, *= и т. д.

Операции преобразования для структурированных типов:

- Конструирование объекта из простых элементов. Пример:

```
DateTime date = new DateTime(2021, 9, 1);
```

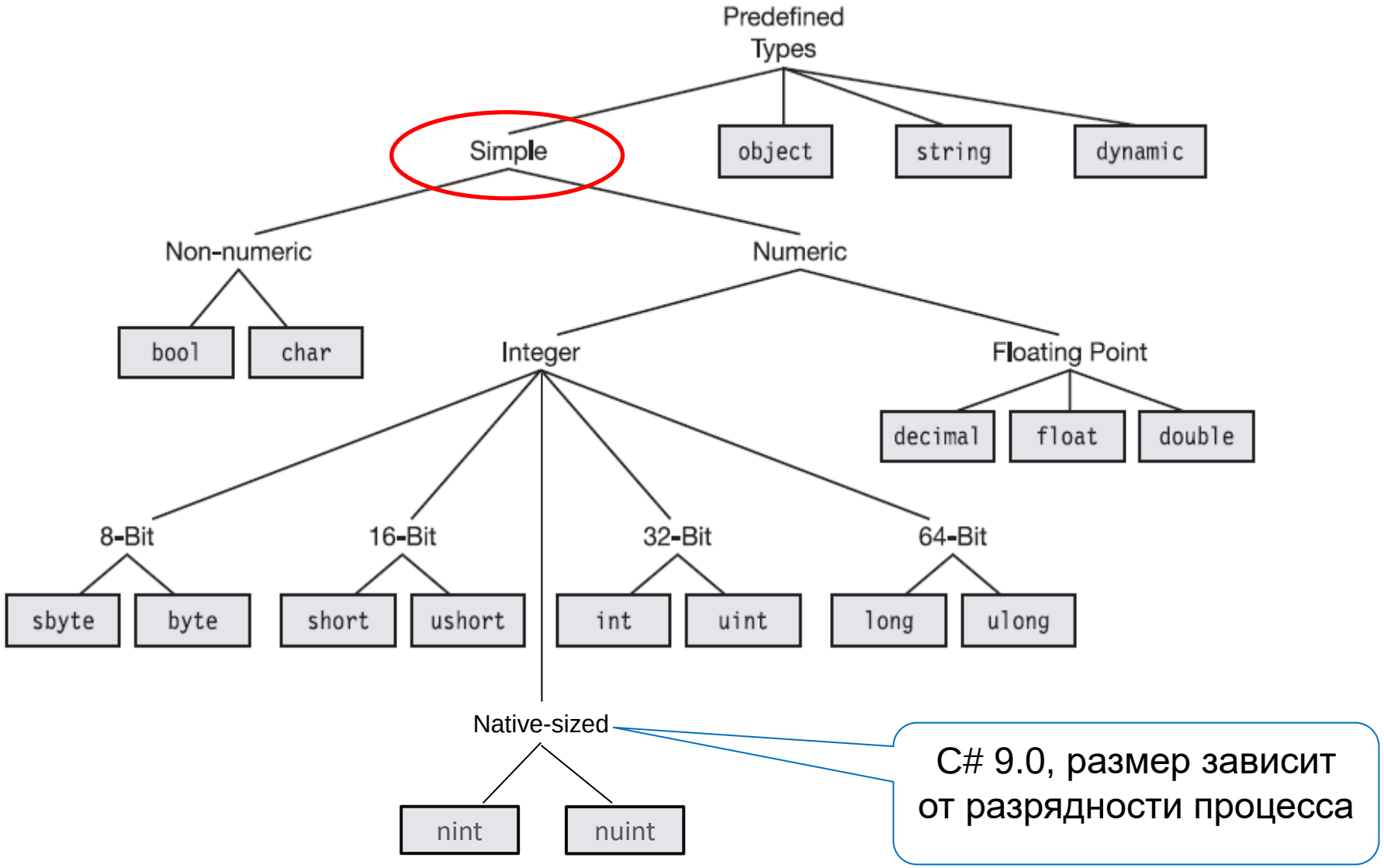
- Извлечение элементов из объекта. Пример:

```
int currentYear = DateTime.Now.Year;
```

Классификация Типов C#

	Типы значений	Типы ссылок
Предопределенные (Элементарные – Primitive Types)	sbyte byte short ushort int uint nint nuint long ulong char float double bool decimal	object string dynamic
Определяемые программистом и библиотечные	Структуры Перечисления	Классы Интерфейсы Делегаты Массивы

Встроенные Типы C#



C# 9.0, размер зависит от разрядности процесса

Виды Переменных

Название	Принадлежность типу или объекту	Описание (Назначение)
Локальная переменная метода	Нет	Хранит временные данные в пределах метода.
Поле	Да	Хранит данные, ассоциированные с типом или его экземплярами.
Параметр метода	Нет	Временная переменная, используемая для передачи данных из одного метода в другой.
Элемент массива	Да	Применяется как временная переменная или как данные, ассоциированные с типом.

Объявление и Инициализация Переменных в C#

Объявление (создание) переменной типа Type:

Type имя_переменной ;

Присваивание существующей переменной значения:

имя_переменной = выражение

Объявление и инициализация:

Type имя_переменной = выражение;

Пример: `double zero = 3.4;`

```
using System;

namespace IntroductionToDataTypes
{
    Ссылок: 0
    class Program {
        // Это объявление целочисленного значения с именем "a"
        static int a = 1;
        Ссылок: 0
        static void Main() {
            // Это объявление целочисленного значения с
            // именем "b" с инициализацией
            int b = 10;
            // Объявление "c" без инициализации
            int c;
            // Последующая инициализация
            c = 100;
            // Выведется 111
            Console.WriteLine(a + b + c);
        }
    }
}
```

Создание Переменной и Объекта по Правилам CLR

Объявление (создание) переменной типа Type:

Type имя_переменной ;

Присваивание переменной значения:

имя_переменной = **new** *Type*(*аргументы_конструктора*);

Объявление и инициализация переменной:

Type имя_переменной = **new** Type(*аргументы_конструктора*);

C# 9.0, тип объекта можно опустить, если он статически известен компилятору

Принцип работы операции new:

1) Выделить память для объекта;
2) Инициализировать специальные члены объекта;
3) Вызвать конструктор;
4) Вернуть ссылку на объект.

«Простые» предопределённые типы в BCL

Тип BCL	Разряды (бит)	Диапазон значений	Значение по умолчанию	Соответвие CLS
System.SByte	8	-128; 127	0	Нет
System.Byte	8	0; 255	0	
System.Int16	16	-32768; 32767	0	
System.UInt16	16	0; 65535	0	Нет
System.Int32	32	-2147483648; 2147483647	0	
System.UInt32	32	0; 4294967295	0	Нет
System.IntPtr	32 или 64	Соответствует либо Int32, либо Int64	0	
System.UIntPtr	32 или 64	Соответствует либо UInt32, либо UInt64	0	
System.Int64	64	-9223372036854775808; 9223372036854775807	0	
System.UInt64	64	0;18446744073709551615	0	Нет
System.Char	16	U+0000; U+ffff	\x00	
System.Single	32	$\pm 1.5*10^{-45}$; $\pm 3.4*10^{38}$	0.0f	
System.Double	64	$\pm 5*10^{-324}$; $\pm 1.7*10^{308}$	0.0d	
System.Boolean		false; true	false	
System.Decimal	128	$-7.9*10^{28}$;$7.9*10^{28}$	0m	

«НЕ Простые» Предопределённые Типы в BCL

System.String (тип **string** в C#) – объект этого типа представляет неизменяемая последовательность символов Unicode.

System.Object (тип **object** в C#) – единый базовый класс для всех типов (и библиотечных и определяемых программистом).

Явные Обращения к Типам BCL в C#

```
using System;
//  Вычислить квадрат целого числа:
System.Int32 x = new System.Int32();
Console.Write("Введите x = ");
System.String input = Console.ReadLine();
x = System.Int32.Parse(input);
Console.WriteLine("x^2 = " + x * x);
```

```
static void Main() {
    // Полные имена для типа int - с пространством
    // имён и без него.
    Int32 int1 = 1;
    System.Int32 int2 = 10;
    // System.Single - псевдоним для float
    Single float1 = int1 + int2;
    // System.Boolean эквивалентно bool. Т. к.
    // float1 > 10 - логическое выражение, такая
    // инициализация является корректной.
    Boolean bool1 = float1 > 10;
}
```

Результат выполнения:
Введите x = 6 <Enter>
x^2 = 36

Обратите внимание: обращения к типам по полным именам для встроенных типов C# полностью эквивалентно обращению к ним по псевдонимам.

Типы C# и BCL

Тип BCL	Тип в C#	Тип BCL	Тип в C#
System.SByte	sbyte	System.Char	char
System.Byte	byte	System.Single	float
System.Int16	short	System.Double	double
System.UInt16	ushort	System.Boolean	bool
System.Int32	int	System.Decimal	decimal
System.UInt32	uint	System.String	string
System.Int64	long	System.Object	dynamic
System.UInt64	ulong	System.Object	object

Варианты Объявления и Инициализации Переменной

```
System.Int32 x1 = new System.Int32();
System.Int32 x2;          // Без инициализатора.
int x3 = new int();       // Явная инициализация - конструктор.
int x4 = new();           // C# 9.0 - тип выводится неявно.
int x5;                   // Без инициализатора.
int x6 = 34;              // Инициализация константой.
```

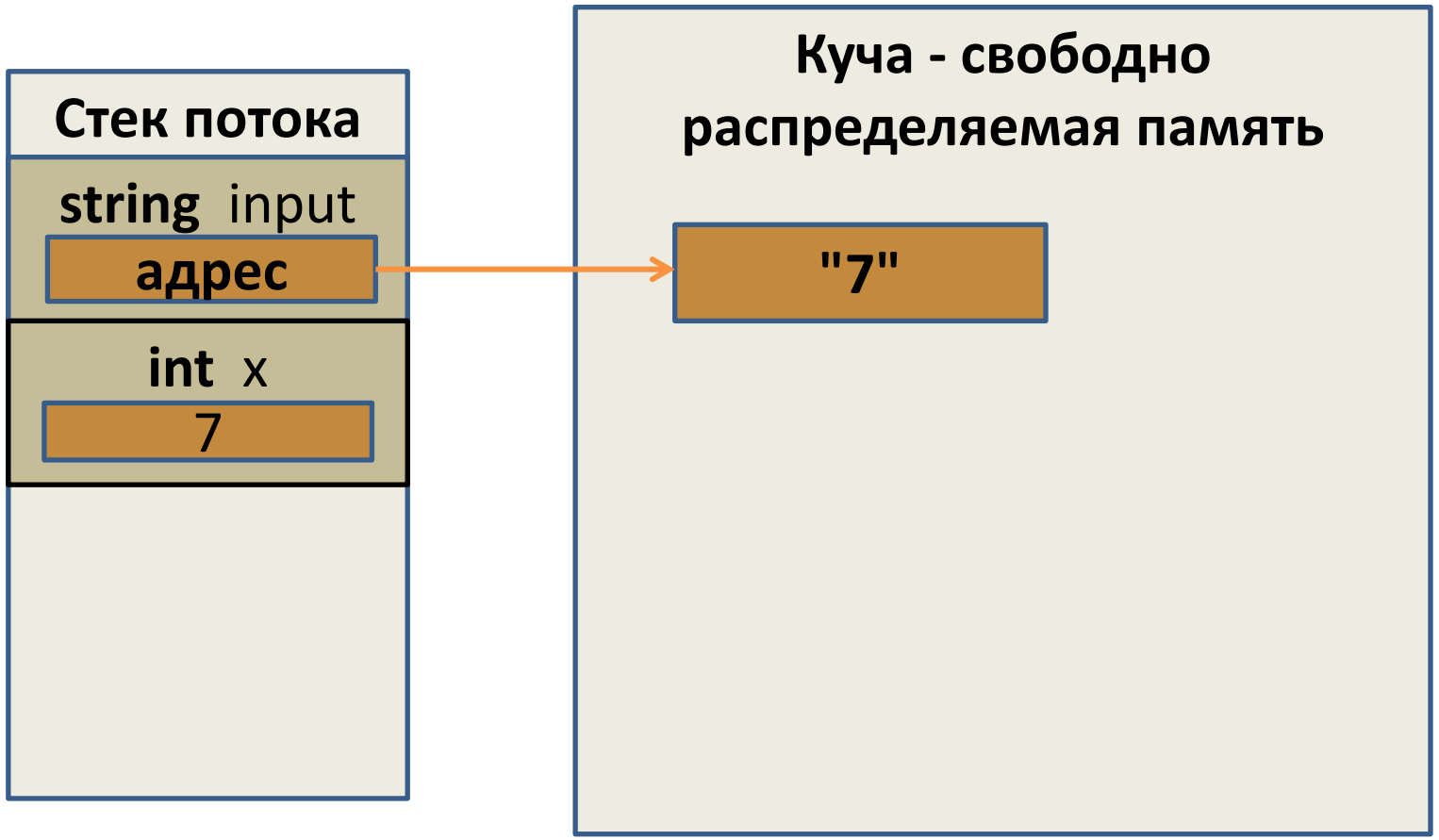
Код на C# с Упрощённой Нотацией Типов

```
using System;  // Оператор в начале файла с кодом.

//  Вычислить квадрат целого числа:
int x;       // Отсутствует инициализатор.
Console.Write("Введите x = ");
string input = Console.ReadLine();
x = int.Parse(input);
Console.WriteLine("x^2 = " + x * x);
```

Результат выполнения:
Введите x = 7<Ввод Enter>
x^2 = 49

Память После Ввода с Клавиатуры Значения x



Примеры Статических Методов Классов BCL

System.Console.Write() – статический метод класса Console.

System.Int32.Parse() - статический метод класса Int32.

System.Console.ReadLine() - статический метод класса Console.

System.Double.TryParse() - статический метод класса Double.

System.Math.Round() - статический метод класса Math.

Уточненное имя статического члена:

Пространство_имён.имя_класса.имя_члена

Нестатические Методы Класса Object

Открытые методы:

Equals() – сравнение экземпляров;

GetHashCode() – вычисление хэш-кода;

GetType() – получение типа объекта;

ToString() – получение строкового представления объекта.

Обращение к нестатическому методу:

ссылка_на_объект.имя_метода(аргументы);

Применение Методов Object – Тип Значения

```
using System;

int x = 7;
Console.WriteLine(x.Equals(4));
Console.WriteLine(x.GetHashCode());
Console.WriteLine(x.GetType());
Console.WriteLine(x.ToString());
```

Результат выполнения:

False
7
System.Int32
7

Применение Методов Object – Ссылочный Тип

```
using System;



string s1 = "Begin";
string s2 = "End";
Console.WriteLine(s1.Equals(s2));
Console.WriteLine(s1.GetHashCode());
Console.WriteLine(s2.GetHashCode());
Console.WriteLine(s1.GetType());
```

Результат выполнения:

```
False
-1088135314
650240692
System.String
```

MemberwiseClone()

- Защищенный метод класса Object:

```
protected object MemberwiseClone()
```

- Создает поверхностную копию (shallow copy) объекта.

Тип Decimal. Основные сведения

- Структурный тип
- Значения хранятся в форме с фиксированной точкой.
- 128-bit
- Диапазон: от $\pm 7{,}9 * 10^{-28}$ до $\pm 7{,}9 * 10^{28}$
- Можно представить числа, имеющие до 28-ми десятичных разрядов.

```
decimal x = 0.999m;
```

https://docs.microsoft.com/ru-ru/dotnet/api/system.decimal?view=net-5.0

Тип Decimal. Пример Программы

```
using System;

decimal x = 0.999m;
decimal y = 9999999999999999999999999999m;
Console.WriteLine("Amount x = {0:C}", x);
Console.WriteLine("Amount y = {0:C}", y);
```

Форматирование значения валюты

Результаты выполнения:

Amount x = $1.00

Amount y = $9,999,999,999,999,999,999,999,999,999.00

Тип dynamic

- хранит внутри ссылку типа object;
- пропускает статическую проверку типов (во время компиляции);
- использует Dynamic Language Runtime (DLR) для связывания во время исполнения;
- используется при работе с языками с динамической типизацией (Python, etc…), гибкость при работе с COM.

Dynamic – динамический тип

```
using System;

class Program {
    static void Main() {
        dynamic value;
        for (int demo = 0; demo < 2; demo++) {
            value = (demo == 0) ? (dynamic)5 : (dynamic)"A";
            value = value + value;
            M(value);
        }
    }
    static void M(int n) {
        Console.WriteLine("M(int): " + n);
    }
    static void M(string s) {
        Console.WriteLine("M(string): " + s);
    }
}
```

Результаты выполнения:

M(Int32): 10

M(String): AA

https://docs.microsoft.com/ru-ru/dotnet/csharp/programming-guide/types/using-type-dynamic