Объектно-ориентированное программирование с использованием Java и C#

Статические члены классов

```
Console.WriteLine("Hello world");

public static class Console
{
  ...
  public static void WriteLine();
}
```

```
System.out.println("Hello world");
```

```
public final class System {
  private System() { }
  ...

  public final static InputStream in;
  ...
}
```

Наследование и переопределение (C#)

```
  Stack<string> stack = new Stack<string>();
  Console.WriteLine(stack);
```

```
algocs.Stack`1[System.String]
```

```
public class Object
{
  public virtual string ToString();
}
```

```
public class Stack<T> : object, IEnumerable<T>
{
  public override string ToString()
  {
    return string.Format("Stack<{0}> ({1} elements)", typeof(T), count);
  }
}
```

```
Stack<System.String> (5 elements)
```

Наследование и переопределение (Java)

```
public class Object {
  public String toString() { ... }
}
```

```
public class Stack3<T> extends Object implements Iterable<T> {

    public String toString() {
        return "Stack<?> (" + count + " elements)";
    }
}
```

final (class) – запрещает наследование от класса
final (method) – запрещает переопределение метода
final (field) – запрещает изменение поля (const)

C#: синтаксический сахар (var)

```
Stack<string> stack = new Stack<string>();

var stack = new Stack<string>();
```

```
class Creator
{
  public Form CreateForm() { ... }
}

var creator = new Creator();
var form = creator.CreateForm();
```

Особенно полезен при использовании LINQ

Свойства (C#)

Свойства предоставляют гибкий доступ к private-полям.

```
class Time
{
  private int hours, minutes, seconds;
  public int Hours
  {
    get { return hours; }
    set
    {
      if (value < 0 || value > 23)
        throw new ArgumentException("Hours should be in 0-23 range");
      hours = value;
    }
  }
  ...

  public int TotalSeconds
  {
    get { return hours * 60 * 60 + minutes * 60 + seconds; }
  }
}
```

Свойства (Java): эмуляция

```
private int x;

public int getX() {
  return x;
}

public void setX(int value) {
  x = value;
}
```

```
private int hours;

public int getHours() { return hours; }

public setHours(int value) {
      if (value < 0 || value > 23)
        throw new RuntimeException("Hours should be in 0-23 range");
      hours = value;
}
```

Свойства (C#): синтаксический сахар

```
private int x;

public int X
{
  get { return x; }
  set { x = value; }
}
```

```
public int X { get; set; }
```

```
public int X { get; protected set; }
```

C#: структуры

Ссылочные типы (reference types): классы
Типы-значения (value types): базовые типы (int, ...), структуры

```
class Time { ... }

struct TimeStruct { ... }
```

Примеры стуктур в .NET Framework:

`DateTime, Point, Rectangle`

Осторожно! При передаче value-типов в метод или возврате из метода значение *копируется*.

```
static void ChangeTime(Time time) { time.Minutes += 2; }

static void ChangeTimeStruct(TimeStruct time) { time.Minutes += 2; }
```

```
class Event
{
  public TimeStruct Moment { get; set; }
}

static void TestStruct2()
{
  var ev = new Event();
  ev.Moment = new TimeStruct(10, 5, 34);
  var t1 = ev.Moment;
  t1.Hours = 9;
  var t2 = ev.Moment;
  Console.WriteLine(t1);
  Console.WriteLine(t2);
}
```