

# Язык интерпретатора cmd.exe

- Интерактивный и пакетный режим
- Только текст, графики нет
- **Операторы** = команды операционной системы:
  - внутренние (set, if, for, copy, dir, ...)
  - внешние утилиты (find, help, xcopy, ...)
- Простые **переменные** (символьные и числовые)
- **Условия** (if ...) и **циклы** (for ...)
- **Подпрограммы** с параметрами

Расширение функциональности – через утилиты, внешние объекты не поддерживаются.

# Инструментарий

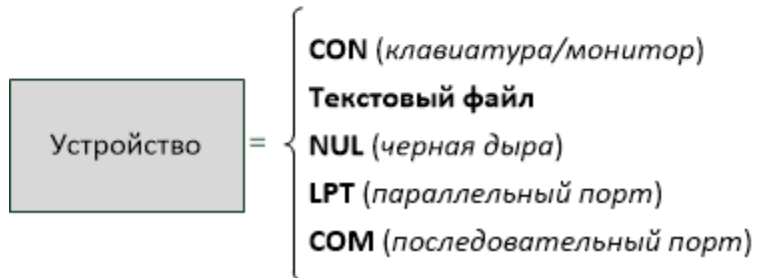
## **Минимум**

- Любой текстовый редактор

## **Дополнительно**

- Специальный редактор (notepad++, ...)
- Файловый менеджер (Total Commander, FAR, ...)

# Потоки ввода/вывода



echo Привет! > copy.txt

xcopy /? >> xcopy\_help.txt

xcopy D:\1.txt C: > copy.txt 2>&1

date < date.txt

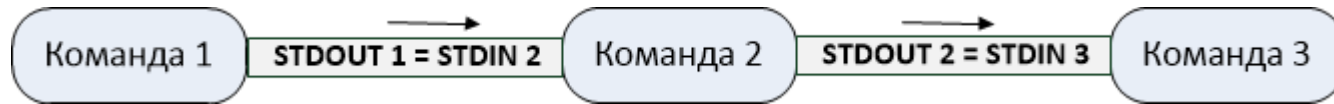
# Вывод в файл (перезапись)

# Добавление в файл

# Дублирование ошибок в STDOUT

# Ввод из файла

# Конвейеризация команд



*Команда 1 | Команда 2 | ...*

`dir c:\ /s /b | find "LOG" | more`

# Структура команд-утилит

*имя параметры ключи*

`dir z:/*.* /s /b`

`ipconfig /all`

Команда = процедура во внешнем файле

Ключи определяют функциональность. Одна команда – несколько функций.

*имя /?* – встроенная помощь к команде

## Параметры командных файлов

%1, %2, ..., %9 – параметры

%0 – путь к запущенному файлу

Команда `shift` – сдвиг параметров влево

Можно выделять из параметров элементы пути:

%~Fn, %~Dn, %~Pn, %~Nn, %~Xn

# Переменные

Список всех переменных: `set`

Стандартные:

`%date%`, `%time%`, `%random%`, `%cd%`, ...

Присваивание

`set имя=значение`    # строка

`set /a имя=значение`    # число

Получение значения

`%имя%`

`!имя!`    (сначала `setlocal enabledelayedexpansion`)

Локальные изменения

`setlocal`

...

`endlocal`

## Символьные операции

**Конкатенация** – без особого знака

set c=%a%%b%

**Выделение подстроки**

%имя:~n1,n2%

**Замена подстроки**

%имя:s1=s2%

Длина строки, поиск подстроки – стандартных нет.



# Математические операции

set a=1

set b=2

set **/a** c=%a%+%b%

+, -, \*, / (целочисленное), смена знака, битовые операции (AND, OR, XOR, сдвиг)

## Операторы условия

if [not] *строка1*==*строка2* *команда1* [else *команда2*]

if [/i] *строка1* *оператор* *строка2* *команда*

*оператор*={EQL, NEQ, LSS, LEQ, GTR, GEQ}

Для чисел также корректно работает

if [not] exist **файл** *команда1* [else *команда2*]

if defined **переменная** *команда1* [else *команда2*]

if [not] **errorlevel** *число* *команда1* [else *команда2*]

Истина, если код возврата последней команды **больше  
либо равен** *числу*

Переменная %errorlevel%

**Условное выполнение команд:**

*команда1* && *команда2*      (в случае успеха)

*команда1* || *команда2*      (в случае неудачи)

# Циклы

## **Арифметический**

for /l %переменная in (начало, шаг, конец) do команда

## **Перебор множества строк или файлов**

for %переменная in (набор) do команда

## **Перебор каталогов**

for /d %переменная in (набор) do команда

## **Цикл для каталога и всех его подкаталогов**

for /r [путь] %переменная in (набор) DO команда

# Цикл для разбора потока текста

for /f ["ключи"] %переменная in (набор) do команда

EOL=c – символ комментария

SKIP=n – пропуск первых строк

DELIMS=xxx – набор разделителей

TOKENS=X,Y,M-N – номера выделяемых подстрок

Можно обрабатывать:

- один или несколько файлов
- одну строку
- STDOUT определенной команды

# Подпрограммы

## **В отдельном файле**

*call имя\_файла параметры*

## **В том же файле**

*call :метка параметры*

*...*

*:метка*

*команды*

*exit /b*

## Недостатки cmd.exe

- Непривычный синтаксис языка
- Нет массивов, структур, логических операций в условиях, удобных процедур и функций
- Приходится анализировать текстовый вывод команд
- Несогласованный синтаксис внешних команд
- Нельзя работать с объектами в операционной системе