

Информационные технологии

**Лекция 4. Кодировка символов. Символьный
тип. Строковые функции. Примеры.**

Кодировка символов



Кодировка символов (часто называемая также кодовой страницей) – это набор числовых значений, которые ставятся в соответствие группе алфавитно-цифровых символов, знаков пунктуации и специальных символов.

Для кодировки символов в Windows используется таблица ASCII (American Standard Code for Interchange of Information).

В ASCII первые 128 символов всех кодовых страниц состоят из базовой таблицы символов. Первые 32 кода базовой таблицы, начиная с нулевого, размещают управляющие коды.

Служебные Кодировка ANSI символы



Код	Описание	Код	Описание	Код	Описание	Код	Описание
0	NUL Нуль	8	Забой (шаг назад) Backspace	16	Ключ связи данных	24	Отказ
1	Начало заголовков	9	Горизонтальная табуляция Tab	17	Управление устройством 1	25	Конец среды
2	Начало текстов	10	Перевод строки	18	Управление устройством 2	26	Замена
3	Конец текста	11	Вертикальная табуляция	19	Управление устройством 3	27	Ключ
4	Конец передачи	12	Новая страница	20	Управление устройством 4	28	Разделитель файлов
5	Запрос	13	Возврат каретки	21	Отрицательное подтверждение	29	Разделитель группы
6	Подтверждение	14	Выключить сдвиг	22	Синхронизация	30	Разделитель записей
7	Сигнал (звонок)	15	Включить сдвиг	23	Конец передаваемого блока	31	Разделитель молулей



Alt+Код

Кодировка СИМВОЛОВ - ANSI

32 Пробел	44 ,	56 8	68 D	80 P	92 \	104 h	116 t
33 !	45 -	57 9	69 E	81 Q	93]	105 i	117 u
34 “	46 .	58 :	70 F	82 R	94 ^	106 j	118 v
35 #	47 /	59 ;	71 G	83 S	95 _	107 k	119 w
36 \$	48 0	60 <	72 H	84 T	96 `	108 l	120 x
37 %	49 1	61 =	73 I	85 U	97 a	109 m	121 y
38 &	50 2	62 >	74 J	86 V	98 b	110 n	122 z
39 ‘	51 3	63 ?	75 K	87 W	99 c	111 o	123 {
40 (52 4	64 @	76 L	88 X	100 d	112 p	124
41)	53 5	65 A	77 M	89 Y	101 e	113 q	125 }
42 *	54 6	66 B	78 N	90 Z	102 f	114 r	126 ~
43 +	55 7	67 C	79 O	91 [103 g	115 s	127 ␣

Кодировка СИМВОЛОВ –



Alt+Код

расширенная

128 А	144 Р	160 а	176 ▤	192 ⌞	208 ≡	224 р	240 ≡Ë
129 Б	145 С	161 б	177 ▥	193 ⊥	209 ≡	225 с	241 ±ë
130 В	146 Т	162 в	178 ▦	194 ⊥	210 ≡	226 т	242 ≥
131 Г	147 У	163 г	179	195 ⊥	211 ≡	227 у	243 ≤
132 Д	148 Ф	164 д	180 ⊥	196 —	212 ≡	228 ф	244 ∫
133 Е	149 Х	165 е	181 =	197 ⊥	213 ≡	229 х	245 ∫
134 Ж	150 Ц	166 ж	182 ≡	198 ≡	214 ≡	230 ц	246 ∫
135 З	151 Ч	167 з	183 ≡	199 ≡	215 ≡	231 ч	247 »
136 И	152 Ш	168 и	184 ≡	200 ≡	216 ≡	232 ш	248 °
137 Й	153 Щ	169 й	185 ≡	201 ≡	217 ≡	233 щ	249 ·
138 К	154 Ъ	170 к	186 ≡	202 ≡	218 ≡	234 ъ	250 ·
139 Л	155 Ы	171 л	187 ≡	203 ≡	219 ≡	235 ы	251 √
140 М	156 Ь	172 м	188 ≡	204 ≡	220 ≡	236 ь	252 °
141 Н	157 Э	173 н	189 ≡	205 =	221 ≡	237 э	253 ²
142 О	158 Ю	174 о	190 ≡	206 ≡	222 ≡	238 ю	254 ■
143 П	159 Я	175 п	191 ⊥	207 ≡	223 ≡	239 я	255

1251 – кодовая страница Windows



128 Ъ	144 Ъ	160	176 °	192 А	208 Р	224 а	240 р
129 Ѓ	145 ‘	161 Ў	177 ±	193 Б	209 С	225 б	241 с
130 ,	146 ’	162 ў	178 l	194 В	210 Т	226 в	242 т
131 ı	147 “	163 Ј	179 i	195 Г	211 У	227 г	243 у
132 „	148 ”	164 ҕ	180 ґ	196 Д	212 Ф	228 д	244 ф
133 ...	149 •	165 Ѓ	181 μ	197 Е	213 Х	229 е	245 х
134 †	150 –	166 †	182 ¶	198 Ж	214 Ц	230 ж	246 ц
135 ‡	151 —	167 §	183 ·	199 З	215 Ч	231 з	247 ч
136 €	152 □	168 Ё	184 ё	200 И	216 Ш	232 и	248 ш
137 ‰	153 ™	169 ©	185 №	201 Ў	217 Щ	233 й	249 щ
138 Љ	154 љ	170 €	186 €	202 К	218 Ъ	234 к	250 ъ
139 <	155 >	171 «	187 »	203 Л	219 Ы	235 л	251 ы
140 Њ	156 њ	172 ¬	188 j	204 М	220 Ь	236 м	252 ь
141 Ќ	157 ќ	173	189 S	205 Н	221 Э	237 н	253 э
142 Ћ	158 ћ	174 ®	190 s	206 О	222 Ю	238 о	254 ю
143 Ў	159 ў	175 ĩ	191 ĩ	207 П	223 Я	239 п	255 я

Кодировки символов

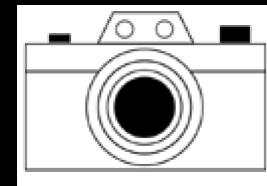


8 бит – 1 байт

EBCDIC (англ. Extended Binary Coded Decimal Interchange Code — расширенный двоично-десятичный код обмена информацией; произносится «и-би-си-дик») — стандартный восьмибитный код, разработанный корпорацией IBM для использования на мэйнфреймах собственного производства и совместимых с ними. Российским аналогом EBCDIC является код **ДКОИ-8**, в который добавлена кодировка кириллицы.

OEM расшифровывается как "Original Equipment Manufacturer". Применительно к компьютерным кодировкам изначально это обозначало набор символов, аппаратнопрошитый в знакогенераторе. А когда знакогенераторы видеокарт перестали аппаратнопрошивать, это стало обозначать кодировку **DOS**.

Кодировки символов



8 бит – 1 байт

Windows-1251 — набор символов и кодировка, являющаяся стандартной 8-битной кодировкой для русских версий Microsoft Windows до 10-й версии.

В современных приложениях отдается предпочтение Юникоду (**UTF-8**).

ISO 8859 — (International Organization for Standardization)
Международная организация по стандартизации

КОИ-8 (код обмена информацией, 8 битов), **KOI8**—
восьмибитовый стандарт кодирования символов в информатике. Разработан для кодирования букв кириллических алфавитов.

Кодировки



СИМВОЛОВ

2 байта – 16

Юнико́д или **Унико́д** (англ. *Unicode*) — стандарт кодирования символов, позволяющий представить знаки практически всех письменных языков.

Стандарт предложен в 1991 году некоммерческой организацией «Консорциум Юникода» (англ. *Unicode Consortium, Unicode Inc.*).

Unicode символов “А” –

0410

“Я” – 042F

Строковый тип данных



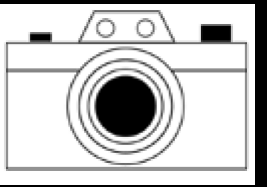
Любые текстовые данные, сохраняемые в VBA, называются строками.

Строки всегда заключаются в двойные кавычки.

Строка может содержать любые символы, начиная с кода 32 кодовых таблиц.

“Пример строковой константы”

Строковый тип данных



Строка переменной длины

Dim s1 as string ‘ Размер 10 байт + от 0 символов до ~ 2 млрд символов

Строка фиксированной длины

Dim s2 as string*N ‘где N-размер строки от 1 символа до 65 400 символов

Операторы над



Конкатенация строками (лат. *concatenatio* «присоединение цепями; сцепление») — операция склеивания объектов линейной структуры, обычно строк.

`sText = "Привет" & " мир!"` □ `sText = " Привет мир!"`

`sText = "Привет" + " мир!"` □ `sText = " Привет мир!"`

Функции кодирования



Asc(string) – возвращает числовой код первого ASCII-символа строки.

Пример: `nInd = Asc("A")` □ `nInd = 65`

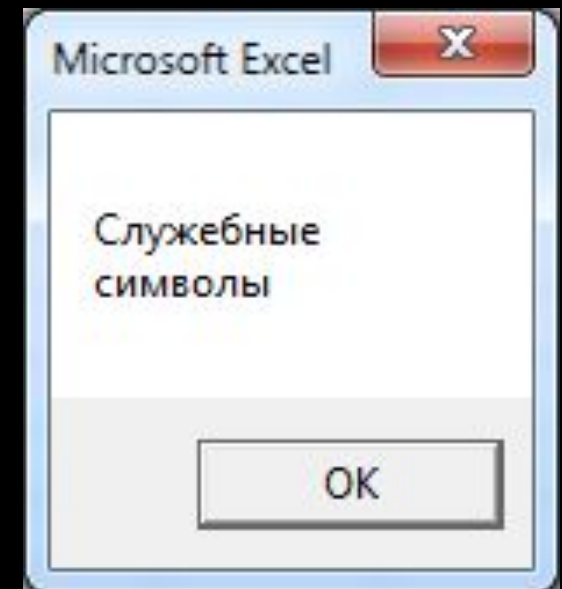
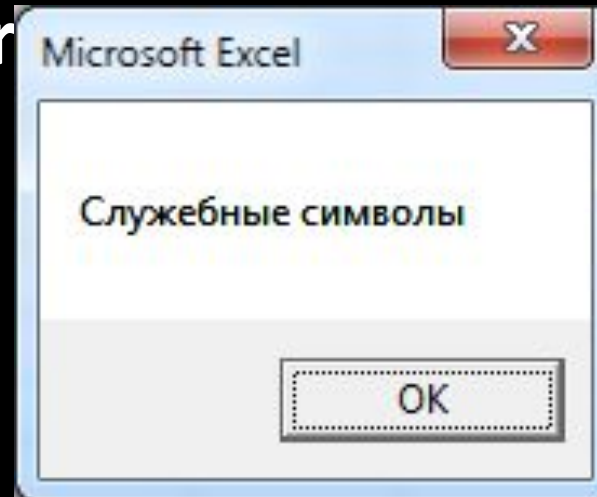
Chr(integer) – возвращает ASCII-символ, соответствующего числового кода в пределах 0-255.

Пример: `sCr = Chr(65)` □ `sCr`

Использование служебных символов

`MsgBox("Служебные символы")`

`MsgBox("Служебные"&Chr(13)
_&"СИМВОЛЫ")`



Строковые функции



Len(string) – возвращает длину строки.

nLength = **Len**("Привет") □ nLength = 6

Получение части строки



Left(string, N) – возвращает N количество символов из левой части строки string.

sText = **Left**("Привет Мир!", 6) □ sText = "Привет"

sText = **Left**("Привет Мир!", 0) □ sText = ""

sText = **Left**("Привет Мир!", 16) □ sText = "Привет Мир!"

Right(string, N) – возвращает N количество символов из правой части строки string.

sText = **Right**("Привет Мир!", 4) □ sText = "Мир!"

sText = **Right**("Привет Мир!", 0) □ sText = ""

sText = **Right**("Привет Мир!", 16) □ sText = "Привет Мир!"

Доступ к символам строки



Функция:

Mid(строка, начало [, длина]) - возвращает указанное количество символов из строки.

где

строка – строковая переменная или константа;

начало – позиция символа в строке;

длина - необязательный параметр, указывающий на количество возвращаемых символов.

sText = **Mid**("Привет Мир!", 8) □ sText = "Мир!"

sText = **Mid**("Привет Мир!", 8, 3) □ sText = "Мир"

sText = **Mid**("Привет Мир!", 12, 3) □ sText = ""

Доступ к символам строки



Оператор:

Mid(строка, начало [, длина]) - заменяет указанное количество символов из другой строки.

где

строка – строковая переменная;

начало – позиция символа в строке;

длина - необязательный параметр, указывающий на количество заменяемых символов.

sText = "Привет Мир!"

Mid(sText, 8) = "Юля" □ sText = "Привет Юля!"

Mid(sText, 8, 1) = "Пэр" □ sText = "Привет Пир!"

Замена символов в



Replace(**строке** Строка, Найти, Заменить) – возвращает измененную строку.

где

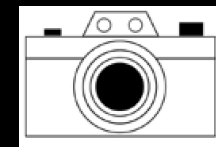
Строка – строковая переменная или константа;

Найти – строковая переменная или константа с фрагментом для поиска в строке;

Заменить – строковая переменная или константа для замены найденного фрагмента

sText=Replace(“Привет Мир!”, “Мир”, “Пир”) □ sText=“Привет Пир!”
sText = Replace(“Привет Мир!”, “и”, “а”) □ sText = “Правет Мар!”

Поиск фрагмента в строке



InStr([*Пуск*,] *Строка*, *Поиск*) – возвращает положение первого появления одной строки внутри другой.
где

Пуск – необязательный параметр, числовая переменная или константа, указывает на позицию, с которой осуществляется поиск в строке.

По умолчанию, поиск начинается с начала строки.

Строка – строковая переменная или

Поиск – строковая переменная или константа с фрагментом для поиска в Строке.

$n = \text{InStr}(\text{“Привет”, “и”}) \quad \square \quad n=3;$

$n = \text{InStr}(1, \text{“Привет”, “и”}) \quad \square \quad n=3.$

Поиск фрагмента в строке



InStrRev(*Строка*, *Поиск* [, *Начало*]) – возвращает позицию вхождения строки *Поиск* в *Строку*, начиная с конца.

где

Строка – строковая переменная или константа;

Поиск – строковая переменная или константа фрагмента поиска;

Начало – числовая переменная или константа, с которого начинается поиск, начиная с конца

$m = \text{InStrRev}(\text{"Привит"}, \text{"и"}) \square m = 5;$

$m = \text{InStrRev}(\text{"Привит"}, \text{"и"}, 4) \square m = 3.$

Сравнение



строк

Операторы условия: $<$, $>$, $=$, $>=$, $<=$, $<>$

nBool = "Вар" < "Дар" ☐ nBool = true

194 < 196 = true

(истина)
nBool = "вар" < "Дар" ☐ nBool = false (ложь)

226 < 196 = false

194 ☐ "В"

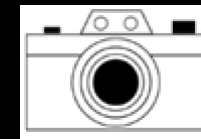
196 ☐ "Д"

226 ☐ "В"

nBool = "Вар" < "Вор" ☐ nBool = true

(истина)

Сравнение



StrComp(*Строка1*, *Строка2*) – возвращает числовой результат сравнения строк.

где

Строка1 – строковая переменная или константа;

Строка2 – строковая переменная или константа.

-1 ☐ Строка1 < n = **StrComp**("Вар", "Дар") ☐ n = -1

0 ☐ Строка1 = Строка2; n = **StrComp**("Вар", "Вар") ☐ n = 0

1 ☐ Строка1 > Строка2. n = **StrComp**("вар", "Дар") ☐ n = 1

Самостоятельно

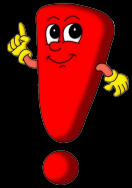
Оператор **Like** – сравнения строк по шаблону

Преобразование



LCase(строка) **строка** – возвращает строку, которую можно преобразовать в нижний регистр.

где строка – строковая переменная или константа.



Только буквы преобразуются в нижний регистр; все буквы нижнего регистра и небуквенные символы остаются неизменными.

sText = **LCase**("C123MM RUS199") □ sText="c123MM rus199"

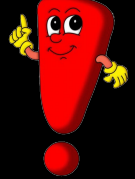
Преобразование



UCase(Строка) ~~Строк~~ возвращает строку, которую можно преобразовать в верхний

где строка – строковый-переменная или

константа



Только строчные буквы преобразуются в верхний регистр; все буквы в верхнем регистре и небуквенные символы остаются неизменными.

```
sText = UCase("Hello World 1234")
```

```
□ sText = "HELLO WORLD 1234"
```

Самостоятельно **StrConv, StrReverse**

Удаление пробелов



LTrim(строка) – возвращает копию строки без начальных пробелов

RTrim(строка) – возвращает копию строки без конечных пробелов

Trim(строка) – возвращает копию строки без начальных и конечных пробелов

где строка – строковая переменная или константа

sText = **LTrim**(" <Пример> ") □ sText = "<Пример> "

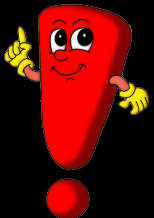
sText = **RTrim**(" <Пример> ") □ sText = " <Пример> "

sText = **Trim**(" <Пример> ") □ sText = "<Пример> "

Создание массива



Array(список аргументов) – возвращает массив из списка где список аргументов - это разделенный запятыми список

 Массив созданный функцией **Array**, имеет индексы, начинающиеся с установленного параметра **Option Base**. Для определения начального и конечного индекса используйте функции **LBound** и **UBound**

Dim M

M = **Array**("Понедельник", "Вторник",
"Среда")

For i=**LBound**(M) **To** **UBound**(M)

sText = M(i)

Next

sText = M(0) = "Понедельник"

sText = M(1) = "Вторник"

sText = M(2) = "Среда"

Создание массива



Самостоятельно изучить
Split и **Join**

Поиск символа в строке



Даны три слова. Определить количество символов
“а”



Sub ЛР4()

Dim M, N() As Integer

M = Array("Правда", «Амур", «Комсомолец")

ReDim N(UBound(M))

For i = LBound(M) To UBound(M)

M(i) = LCase(M(i))

nPos = 0

Do

nPos = InStr(nPos + 1, M(i), "a")

N(i) = If(nPos > 0, N(i) + 1, N(i))

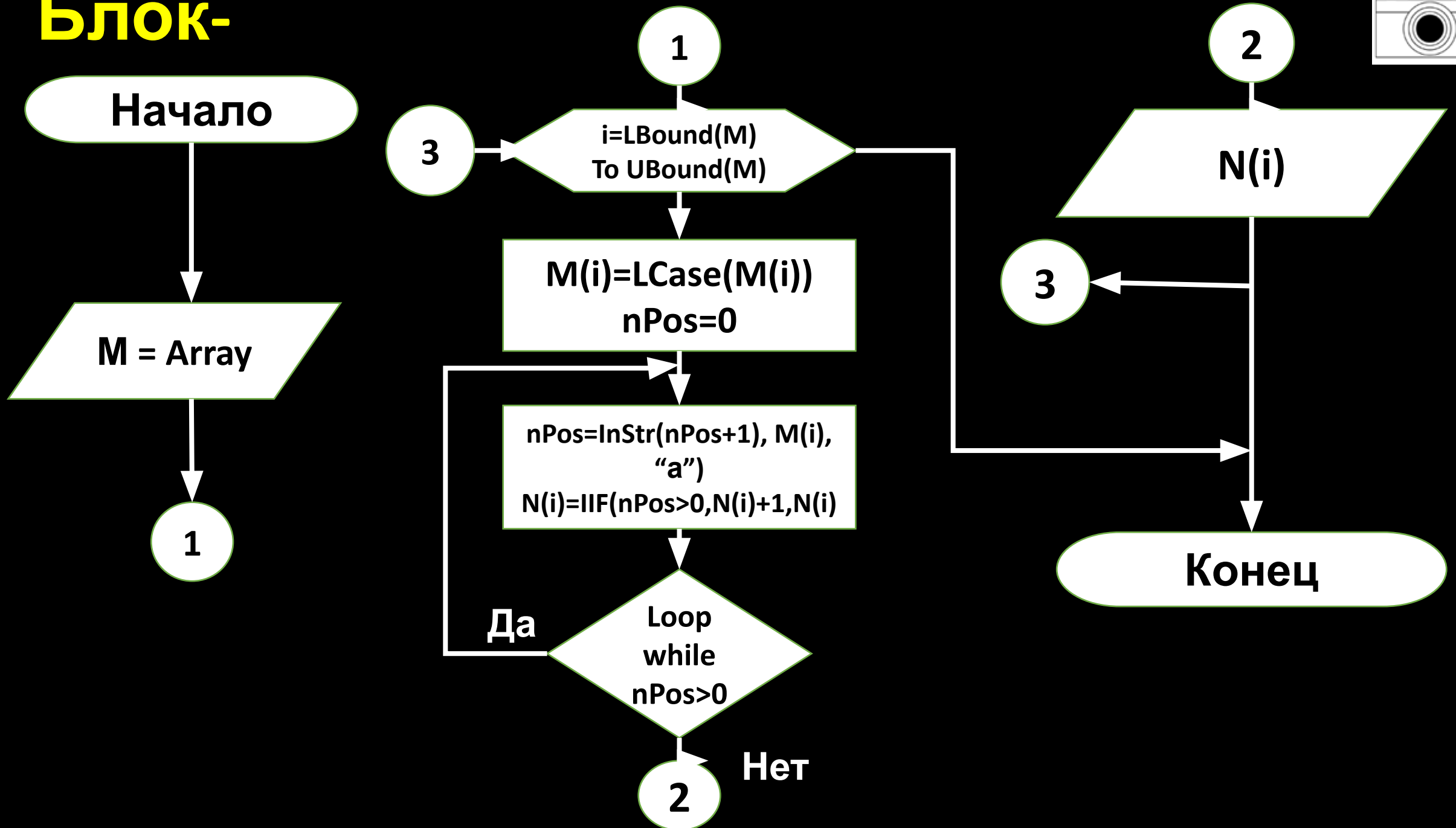
Loop While nPos > 0

MsgBox («В слове" + M(i) + " количество символов а=" &

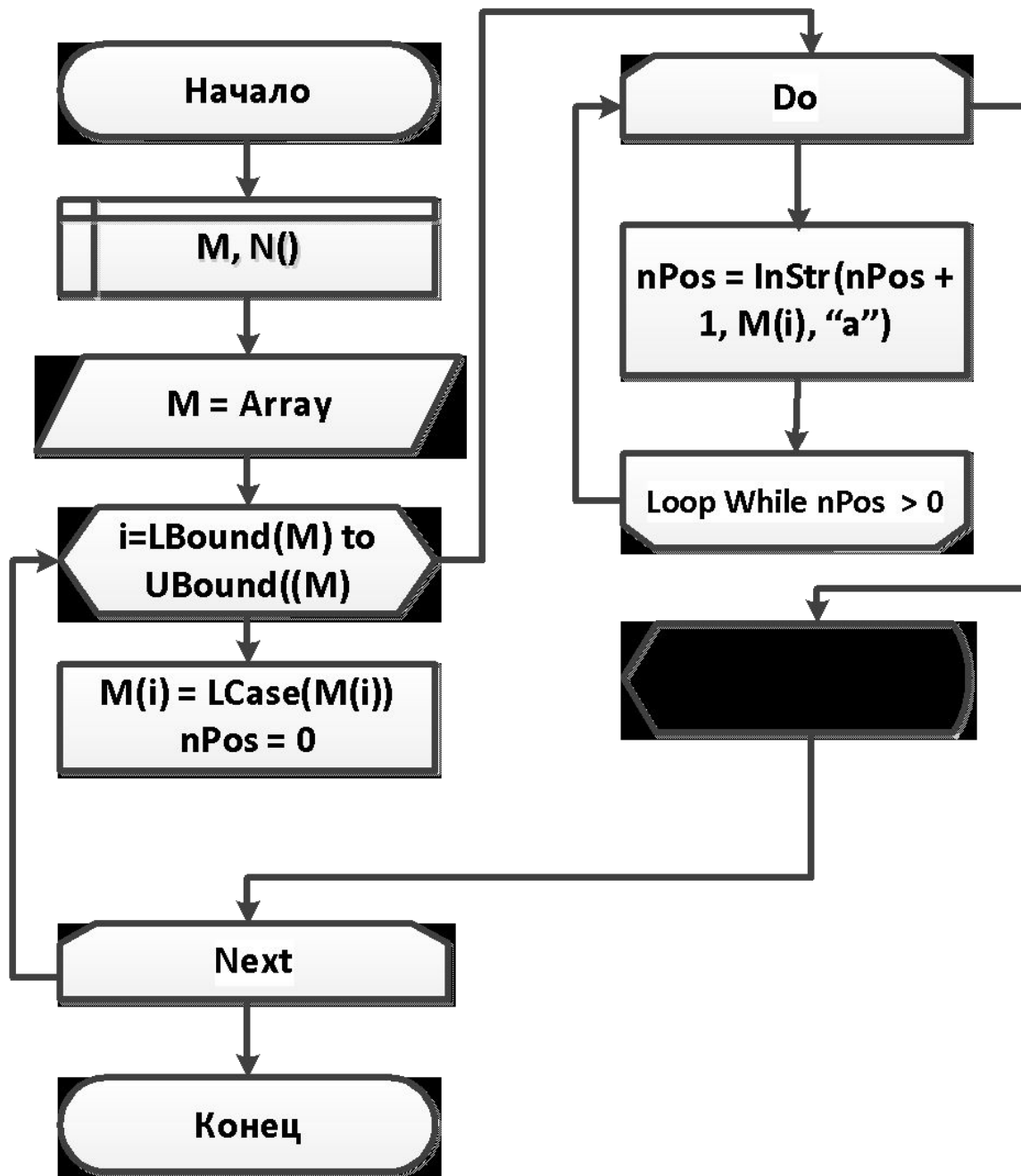
N(i))

End Sub

Блок-



Блок- схема



**Конец
лекции**