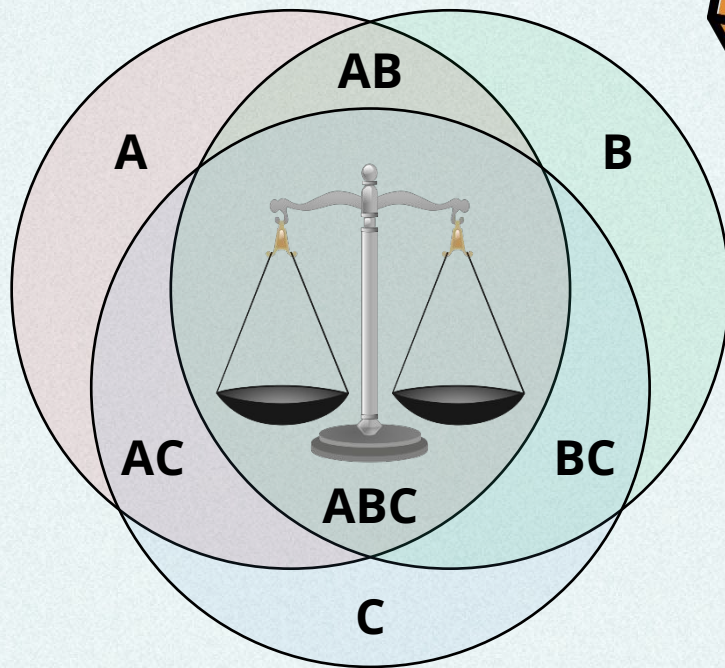
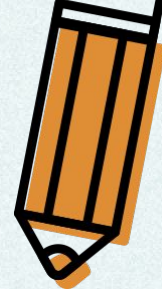
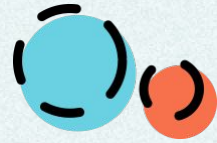


Символьный тип данных

Программирование обработки
информации







Символьный тип данных

1

Значения
символьного
типа.

2

Функции для
символьных
данных.

3

Принцип
последовательного
кодирования алфавитов.



Символьная величина

Величина символьного типа (char)

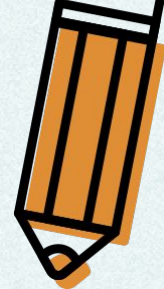
может хранить любой символ
компьютерного алфавита.

Кодовая таблица:

unicode-16.

Размерность переменной:

2 байта.



Порядковый тип

Величина символьного типа

может принимать конечное упорядоченное множество значений.

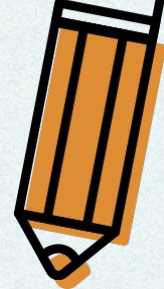
Коды символов:

0 .. 31 – управляющие символы;

32 – пробел;

33 .. 127 – буквы английского алфавита, цифры, знаки препинания и знаки арифметических операций;

128 .. 65 535 – символы региональных алфавитов.



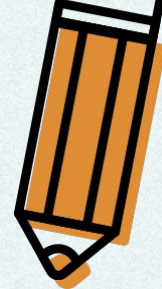
Задача

В файле **input.txt** в первых 2 строках записаны 2 слова одинаковой длины до 20 символов. Вывести на экран расстояние между ними.

$$s = 3$$

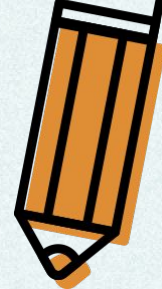
казак
| | | | |
x ✓ x ✓ x
| | | | |
радар

Написание программы



```
program rasstoyanie;  
var  
  f: text;  
  a, b: array [1..20] of char;  
  i, n, s: byte;  
begin  
  assign (f, 'input.txt');  
  reset (f);  
  n:=0;  
  while not (eoln (f)) do  
  begin  
    n:=n+1;  
    read (f, a[n]);  
  end;  
  readln (f);
```

```
for i:=1 to n do  
  read (f, b[i]);  
close (f);  
s:=0;  
for i:=1 to n do  
  if a[i]<>b[i]  
  then s:=s+1;  
write ('Расстояние между словами,  
содержащимися в файле: ', s);  
end.
```



Функции для
работы с
символьными
величинами

Преобразование
символа в его код

`ord (c)`

Преобразование
кода в символ

`chr (x)`

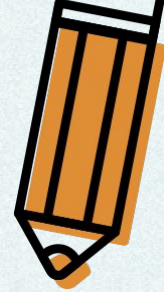


Задача

Написать программу, которая выводит на экран коды с 33 по 127 и соответствующие им символы.

81 → Q

Написание программы




```
program simvoly;  
var  
  i: byte;  
begin  
  for i:=33 to 127 do  
    writeln (i:3, ' - ', chr (i));  
end.
```

Принцип последовательного кодирования алфавитов

33 - !	47 - /	61 - =	75 - K	89 - Y	103 - g	117 - u
34 - "	48 - 0	62 - >	76 - L	90 - Z	104 - h	118 - v
35 - #	49 - 1	63 - ?	77 - M	91 - [105 - i	119 - w
36 - \$	50 - 2	64 - @	78 - N	92 - \	106 - j	120 - x
37 - %	51 - 3	65 - A	79 - O	93 -]	107 - k	121 - y
38 - &	52 - 4	66 - B	80 - P	94 - ^	108 - l	122 - z
39 - '	53 - 5	67 - C	81 - Q	95 - _	109 - m	123 - {
40 - (54 - 6	68 - D	82 - R	96 - `	110 - n	124 -
41 -)	55 - 7	69 - E	83 - S	97 - a	111 - o	125 - }
42 - *	56 - 8	70 - F	84 - T	98 - b	112 - p	126 - ~
43 - +	57 - 9	71 - G	85 - U	99 - c	113 - q	127 - □
44 - ,	58 - :	72 - H	86 - V	100 - d	114 - r	
45 - -	59 - ;	73 - I	87 - W	101 - e	115 - s	
46 - .	60 - <	74 - J	88 - X	102 - f	116 - t	

Принцип последовательного кодирования алфавитов



Примеры истинных высказываний:

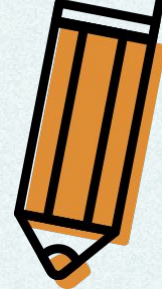
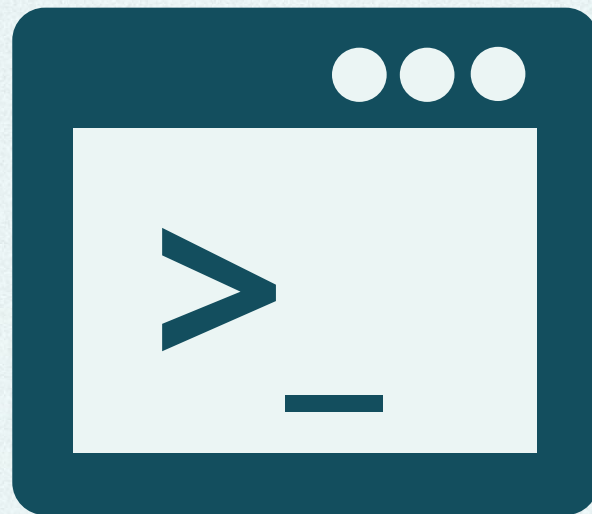
'0' > '5' > '9', 'b' > 't' > 'z', 'D' > 'Q' > 'Z'.

Символ является цифрой:

'0' ≤ c ≤ '9'.

Символ является буквой английского алфавита:

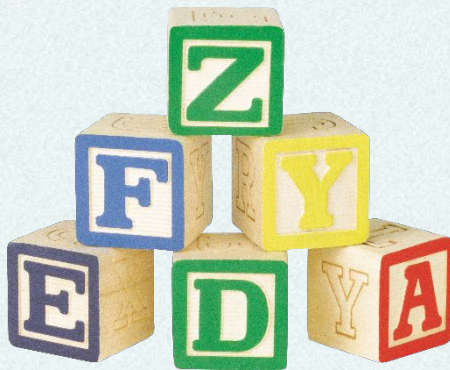
'a' ≤ c ≤ 'z' или 'A' ≤ c ≤ 'Z'.



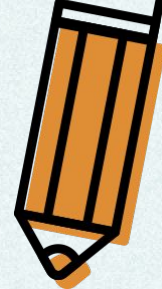
Задача

Написать программу, которая заполняет массив из **n** элементов случайными строчными буквами английского алфавита, после чего сортирует его элементы по невозрастанию. Программа должна вывести на экран как исходный, так и отсортированный массив.

$$1 \leq n \leq 30$$



Написание и тестирование программы

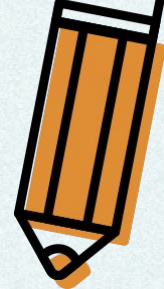


```
program sortorovka_simvolov;  
var  
  s: array [1..30] of char;  
  n, i: byte;
```

```
procedure sort ();  
var  
  p: boolean;  
  i: byte;  
  k: char;  
begin  
  repeat  
    p:=true;
```

```
    for i:=1 to n-1 do  
      if s[i]<s[i+1]  
        then begin  
          p:=false;  
          k:=s[i];  
          s[i]:=s[i+1];  
          s[i+1]:=k;  
        end;  
    until p;  
  end;
```


Написание программы



begin

writeln ('Программа, которая генерирует массив из n случайных английских букв и сортирует его по невозрастанию.');

write ('n=');

readln (n);

for i:=1 to n **do**

 s[i]:=chr (ord ('a')+random (26));

write ('Исходный массив:');

for i:=1 to n **do**

 write (s[i]);

writeln;

sort ();

write ('Отсортированный массив:');

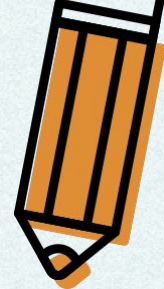
for i:=1 to n **do**

 write (s[i]);

writeln;

end.

Символьный тип данных



Величина символьного типа (char)

может хранить любой символ компьютерного алфавита.

Функции символьных величин:

ord (c) – преобразует символ **c** в его код в кодовой таблице;

chr (x) – преобразует код **x** в соответствующий символ кодовой таблицы.

Переменная символьного типа

занимает 2 байта оперативной памяти.

Принцип последовательного кодирования алфавитов

выполняется для букв английского алфавита и цифр во всех кодовых таблицах символов.

