

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное  
учреждение высшего образования  
«Алтайский государственный технический университет им. И. И. Ползунова»

Университетский технологический колледж

Отчет защищен с оценкой \_\_\_\_\_  
Преподаватель \_\_\_\_\_ *С. В. Умбетов*  
« \_\_\_\_\_ » \_\_\_\_\_ 2025 г.

Отчёт по лабораторной работе № 1  
«Реализация шифра ROT13 на JavaScript»  
ЛР 09.03.01.14.002

Студент группы 1ИСП-21  
группа

А.А. Кайль  
и.о., фамилия

Преподаватель ассистент, к. т. н.  
должность, ученая степень

С. В. Умбетов  
и.о., фамилия

БАРНАУЛ 2025

## Лабораторная работа №1

**Цели и задачи работы:** необходимо создать html страницу и js код, страница должна быть валидной стандарту HTML5. В коде реализовать шифрование ROT13 для четного варианта и расшифровку для нечетного.

Задание принял: \_\_\_\_\_ Кайль А.А.

Подпись

ФИО

## Ход работы

ROT13 («поворот на 13 позиций») — это простой и широко известный шифр замены букв, который используется на онлайн-форумах и в текстовых файлах для сокрытия текста, например предупреждений о спойлерах или оскорбительном контенте. Этот метод заменяет букву на 13-ю по счёту в алфавите. А становится N, В становится О, С становится Р и так далее.

ROT13 является обратным к самому себе, то есть для отмены ROT13 применяется тот же алгоритм, поэтому одно и то же действие можно использовать для кодирования и декодирования.

Алгоритм не обеспечивает криптографическую безопасность и часто приводится в качестве канонического примера слабого шифрования.

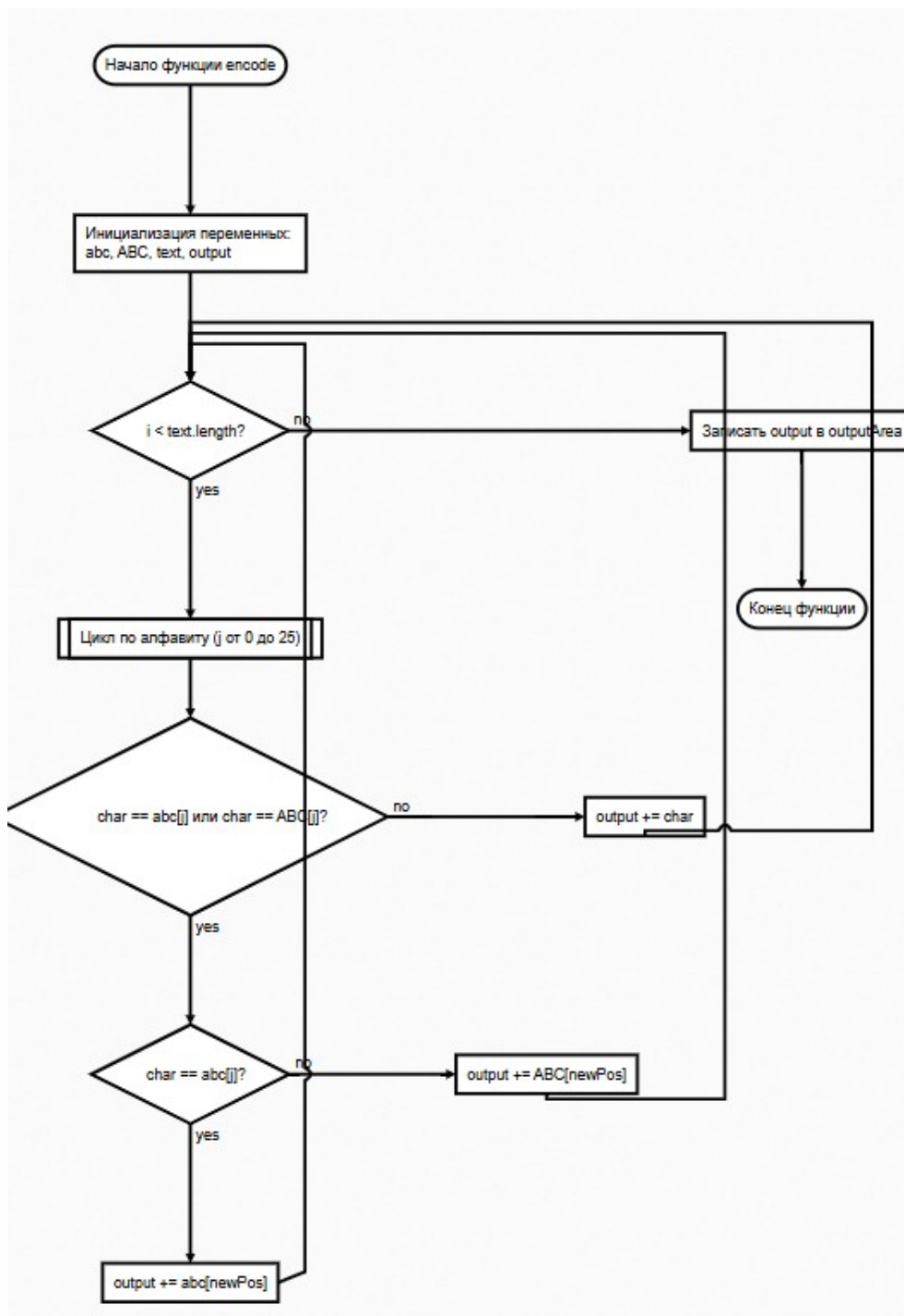


Рисунок 1 — Блок-схема

```
function encode() {
  let abc = 'abcdefghijklmnopqrstuvwxyz';
  let ABC = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ';
  let text = document.getElementById('inputArea').value;
  let output = '';
```

Сначала создадим основную функцию encode(), abc – строчные буквы, ABC – заглавные буквы, text — для шифрования из текстового поля с id="inputArea". и output чтобы записать сюда зашифрованный текст.

```
for (let i = 0; i < text.length; i++) {
  let char = text[i];
  let found = false;
```

Это цикл, который проходит по каждому символу введённого текста. `char` – текущий символ (например, `h` в слове `hello`) `found` – флаг, который показывает была ли буква найдена в алфавите.

```
for (let j = 0; j < abc.length; j++) {  
    if (char === abc[j] || char === ABC[j]) {
```

`for ((let j = 0; j < abc.length; j++))` вложенный цикл проверяет есть ли текущий символ (`char`) в алфавите (`abc` или `ABC`).

`if (char === abc[j] || char === ABC[j])` если символ совпадает с буквой в алфавите (строчной или заглавной) выполняется шифрование.

```
        let shift = 13;  
        let newPos = (j + shift) % abc.length;
```

`newPos = (j + shift) % abc.length` вычисляет позицию буквы после сдвига. Если `j + shift` больше 26 позиция следует в начало.

Буква `n` (индекс 13):  $(13 + 13) \% 26 = 0 \rightarrow a$

```
if (char === abc[j]) {  
    output += abc[newPos];  
} else {  
    output += ABC[newPos];  
}  
found = true;  
break;
```

Если строчный то берём новую букву из `abc[newPos]`, если заглавный `ABC[newPos]`

Для цифр знаков препинаний и тд.

```
if (!found) {  
    output += char;  
}
```

Если символ не был найден в алфавите то выводится без изменений

```
index.html M # style.css M JS script.js M X
practice > script > JS script.js > ...
1  function encode() {
2      let abc = 'abcdefghijklmnopqrstuvwxyz';
3      let ABC = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ';
4      let text = document.getElementById('inputArea').value;
5      let output = '';
6
7      for (let i = 0; i < text.length; i++) {
8          let char = text[i];
9          let found = false;
10
11         for (let j = 0; j < abc.length; j++) {
12             if (char === abc[j] || char === ABC[j]) {
13                 let newPos = (j + 13) % 26;
14
15                 if (char === abc[j]) {
16                     output += abc[newPos];
17                 } else {
18                     output += ABC[newPos];
19                 }
20                 found = true;
21                 break;
22             }
23         }
24
25         if (!found) {
26             output += char;
27         }
28     }
29
30     document.getElementById('outputArea').value = output;
31 }
32 document.addEventListener('DOMContentLoaded', function() {
33     document.getElementById('inputArea').addEventListener('input', encode);
34 });
```

Рисунок 2 — Код JS

# Тестирование

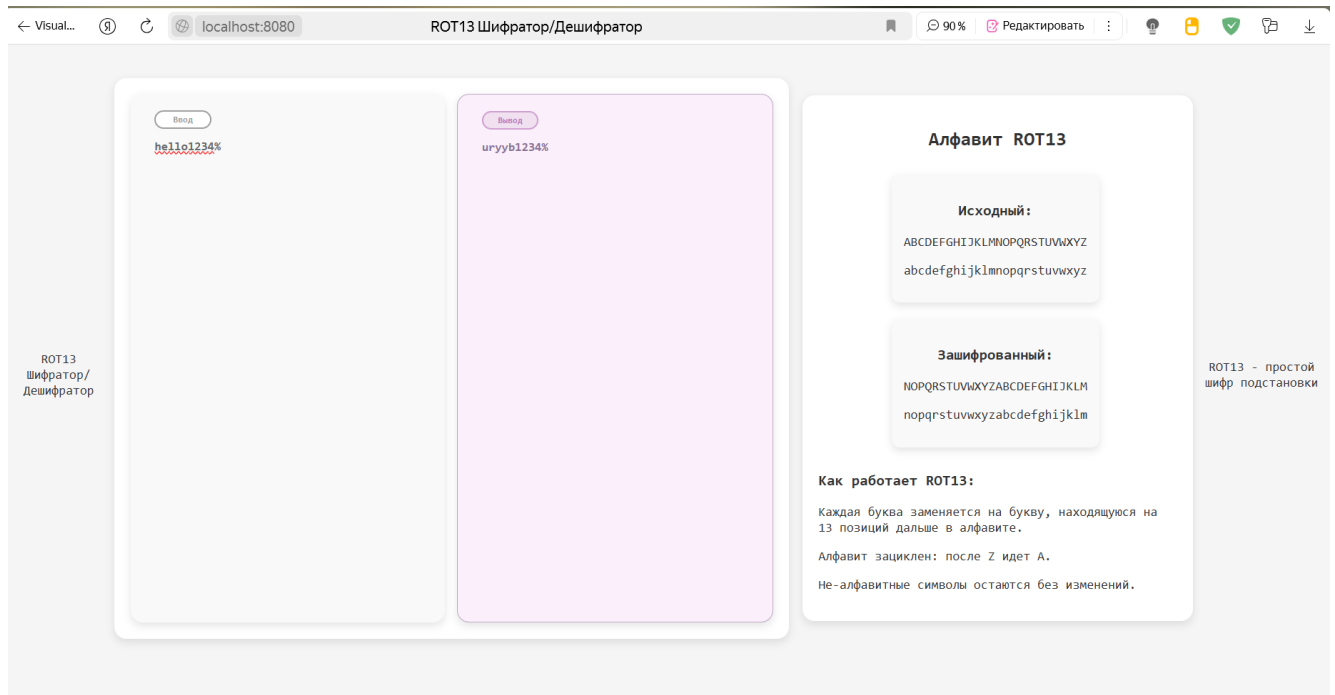


Рисунок 3 — Тестирование шифровка

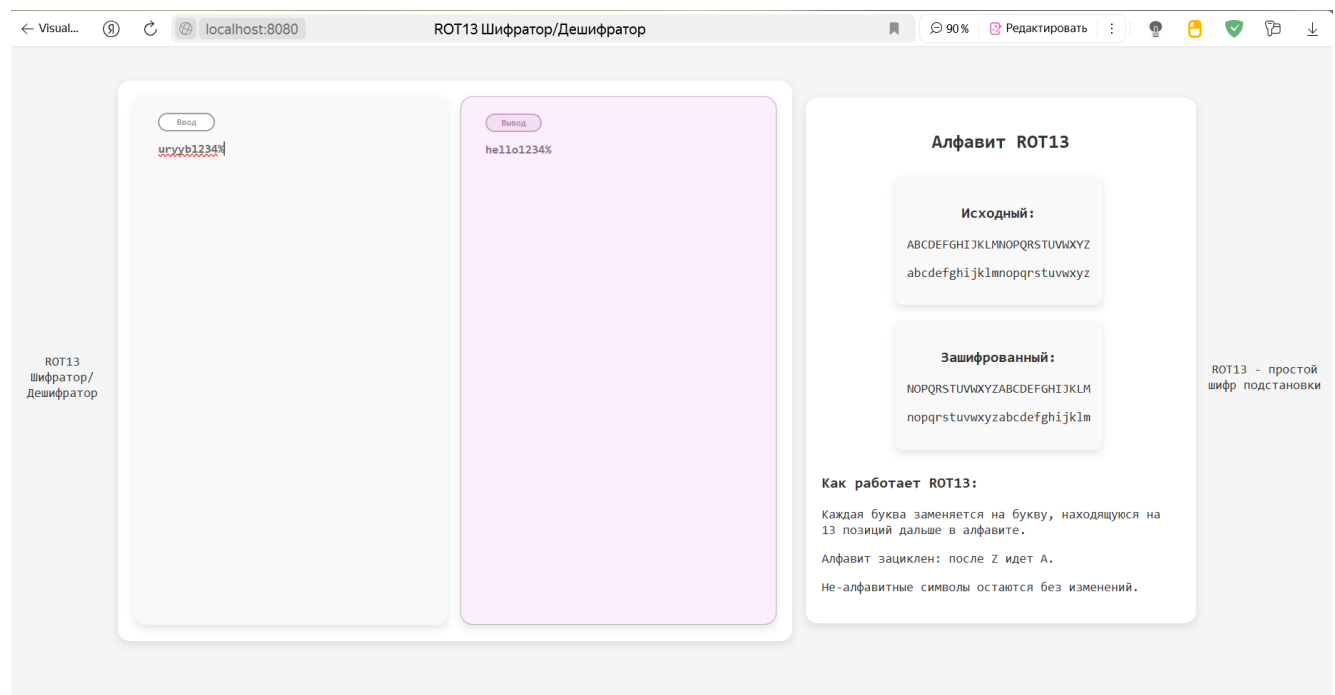


Рисунок 4 — Тестирование дешифровка

## Проверка

Таблица 1: Проверка шифровки

Символ	Тип	Позиция (j)	newPos = $(j+13)\%26$	Измененный символ	Результат
h	строчная	7	$(7+13)\%26 = 20$	u (abc[20])	u
e	строчная	4	$(4+13)\%26 = 17$	r (abc[17])	ur
l	строчная	11	$(11+13)\%26 = 24$	y (abc[24])	ury
l	строчная	11	$(11+13)\%26 = 24$	y (abc[24])	uryy
o	строчная	14	$(14+13)\%26 = 1$	b (abc[1])	uryyb
1	цифра	-	-	1 (без изменений)	uryyb1
%	символ	-	-	% (без изменений)	uryyb1234%

Таблица 2: Проверка дешифровки

Символ	Тип	Позиция (j)	newPos = $(j+13)\%26$	Изначальный символ	Результат
u	строчная	20	$(20+13)\%26 = 7$	h (abc[7])	h
r	строчная	17	$(17+13)\%26 = 4$	e (abc[4])	he
y	строчная	24	$(24+13)\%26 = 11$	l (abc[11])	hel
y	строчная	24	$(24+13)\%26 = 11$	l (abc[11])	hell
b	строчная	1	$(1+13)\%26 = 14$	o (abc[14])	hello
1	цифра	-	-	1 (без изменений)	hello1
%	символ	-	-	% (без изменений)	hello1234%



## Код

### HTML

```
index.html M X # style.css M
practice > index.html > html > body
1 <!DOCTYPE html>
2 <html lang="ru">
3 <head>
4   <meta charset="UTF-8">
5   <title>ROT13 Шифратор/Дешифратор</title>
6   <link rel="stylesheet" href="./style/style.css">
7   <script src="./script/script.js"></script>
8 </head>
9 <body>
10 <header>
11   <p>ROT13 Шифратор/Дешифратор</p>
12 </header>
13
14 <main>
15   <div class="centralBlock">
16     <div class="inputPanel">
17       <p>Ввод</p>
18       <textarea id="inputArea" placeholder="Введите текст для шифрования/дешифрования..."></textarea>
19     </div>
20
21     <div class="outputPanel">
22       <p>Вывод</p>
23       <textarea id="outputArea" readonly></textarea>
24     </div>
25   </div>
26 </main>
27 <div class="alphabet-info">
28   <h2>Алфавит ROT13</h2>
29   <div class="alphabet-pair">
30     <div class="alphabet-original">
31       <h3>Исходный:</h3>
32       <p>ABCDEFGHIJKLMNOPQRSTUVWXYZ</p>
33       <p>abcdefghijklmnopqrstuvwxyz</p>
34     </div>
35     <div class="alphabet-rot13">
36       <h3>Зашифрованный:</h3>
37       <p>NOPQRSTUVWXYZABCDEFGHIJKLM</p>
38       <p>nopqrstuvwxyzabcdefghijklm</p>
39     </div>
40   </div>
41   <div class="operation-info">
42     <h3>Как работает ROT13:</h3>
43     <p>Каждая буква заменяется на букву, находящуюся на 13 позиций дальше в алфавите.</p>
44     <p>Алфавит за циклен: после Z идет A.</p>
45     <p>Не-алфавитные символы остаются без изменений.</p>
46   </div>
47 </div>
48
49 <footer>
50   <p>ROT13 - простой шифр подстановки</p>
51 </footer>
52
53
54 </body>
55 </html>
```

Рисунок 5 — Код html

## Nu Html Checker

This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change

### Showing results for contents of text-input area

Checker Input

Show ☒ source ☐ outline ☐ image report [Options...](#)

Check by text input ☐ css

```
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>rot13</title>
  <link rel="stylesheet" href="/style/style.css">
</head>
<body>
  <header>
  </header>

  <main>
    <div class="centralBlock">
      <div class="inputPanel">
        <p>Ввод</p>
```

Check

Use the Message Filtering button below to hide/show particular messages, and to see total counts of errors and warnings.

[Message Filtering](#)

Document checking completed. No errors or warnings to show.

Рисунок 6 — Проверка html на валидность

## CSS

```
practice > style > # style.css > .alphabet-original
1  body {
2      font-family: 'Consolas', monospace;
3      background-color: #f5f5f5;
4      margin: 0;
5      padding: 0;
6      color: #333;
7      display: flex;
8      align-items: center;
9      justify-content: center;
10     min-height: 100vh;
11 }
12
13 header, footer {
14     text-align: center;
15     padding: 10px;
16     font-family: 'Consolas', monospace;
17 }
18
19 .centralBlock {
20     display: flex;
21     justify-content: center;
22     height: 80vh;
23     width: 60vw;
24     max-width: 800px;
25     min-width: 300px;
26     background-color: white;
27     padding: 20px;
28     border-radius: 15px;
29     box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
30     font-family: 'Consolas', monospace;
31     margin-bottom: 40px;
32 }
33
34 .inputPanel, .outputPanel {
35     flex: 1;
36     display: flex;
37     flex-direction: column;
38     border-radius: 15px;
39     padding: 15px;
40     position: relative;
41     min-width: 120px;
42     font-family: 'Consolas', monospace;
43 }
44
45 .inputPanel {
46     background-color: #f9f9f9;
47     box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
48     margin-right: 15px;
49 }
50
51 .outputPanel {
52     border: 1px solid #8c729589;
53     background-color: #fbefeb;
54     box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
55 }
56
57 .inputPanel p, .outputPanel p {
58     display: flex;
59     justify-content: center;
60     align-items: center;
61     width: 60px;
62     font-family: 'Consolas', monospace;
63     padding: 5px 10px;
64     border-radius: 15px;
65     font-weight: 750;
66     font-size: 11px;
67     margin-top: 5px;
```

Рисунок 7 — Стили css

```

index.html M  # style.css M X
practice > style > # style.css > .alphabet-original
85  textarea {
97      background-color: transparent;
98  }
99
100  .outputPanel textarea {
101      color: #8c7295;
102      caret-color: transparent;
103  }
104
105  .outputPanel textarea[readonly] {
106      cursor: default;
107  }
108  .alphabet-info {
109      background-color: white;
110      padding: 20px;
111      border-radius: 15px;
112      box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
113      margin: 0 auto 40px;
114      max-width: 500px;
115      width: 50%;
116      margin-left: 1%;
117  }
118
119  .alphabet-info h2 {
120      text-align: center;
121  }
122  .alphabet-pair {
123      display: flex;
124      justify-content: space-around;
125      margin: 20px 0;
126      flex-wrap: wrap;
127  }
128
129  .alphabet-original, .alphabet-rot13 {
130      text-align: center;
131      padding: 15px;
132      border-radius: 10px;
133      margin: 10px;
134      min-width: 200px;
135      background-color: #f9f9f9;
136      box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
137      margin-right: 15px;
138  }
139
140  @media (max-width: 480px) {
141      .alphabet-info {
142          width: 95%;
143          padding: 10px;
144      }
145
146      .alphabet-original, .alphabet-rot13 {
147          min-width: 150px;
148          padding: 10px;
149          font-size: 14px;
150      }
151
152      .centralBlock {
153          width: 95vw;
154          padding: 10px;
155      }
156
157      .inputPanel p, .outputPanel p {
158          width: 50px;
159          font-size: 10px;
160          padding: 4px 8px;
161          margin-left: 10px;
162      }

```

Рисунок 8 — Стили css @media



W3C результаты проверки CSS для TextArea (CSS3 + SVG)

**Поздравляем! Ошибок не обнаружено.**

Этот документ проходит проверку по стандарту [CSS3 + SVG](#) !

Рисунок 8 — Стили css проверка на валидность

## **Вывод**

В ходе данной лабораторной работы была создана функция, которая реализует алгоритм rot13 на JavaScript, который шифрует и дешифрует введенные данные. Проблемы: обработка регистра букв(раздельные алфавиты строчные и заглавные), использование % 26 для обратного сдвига.

<https://github.com/him1k0ta/practice.git>