

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

## 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.

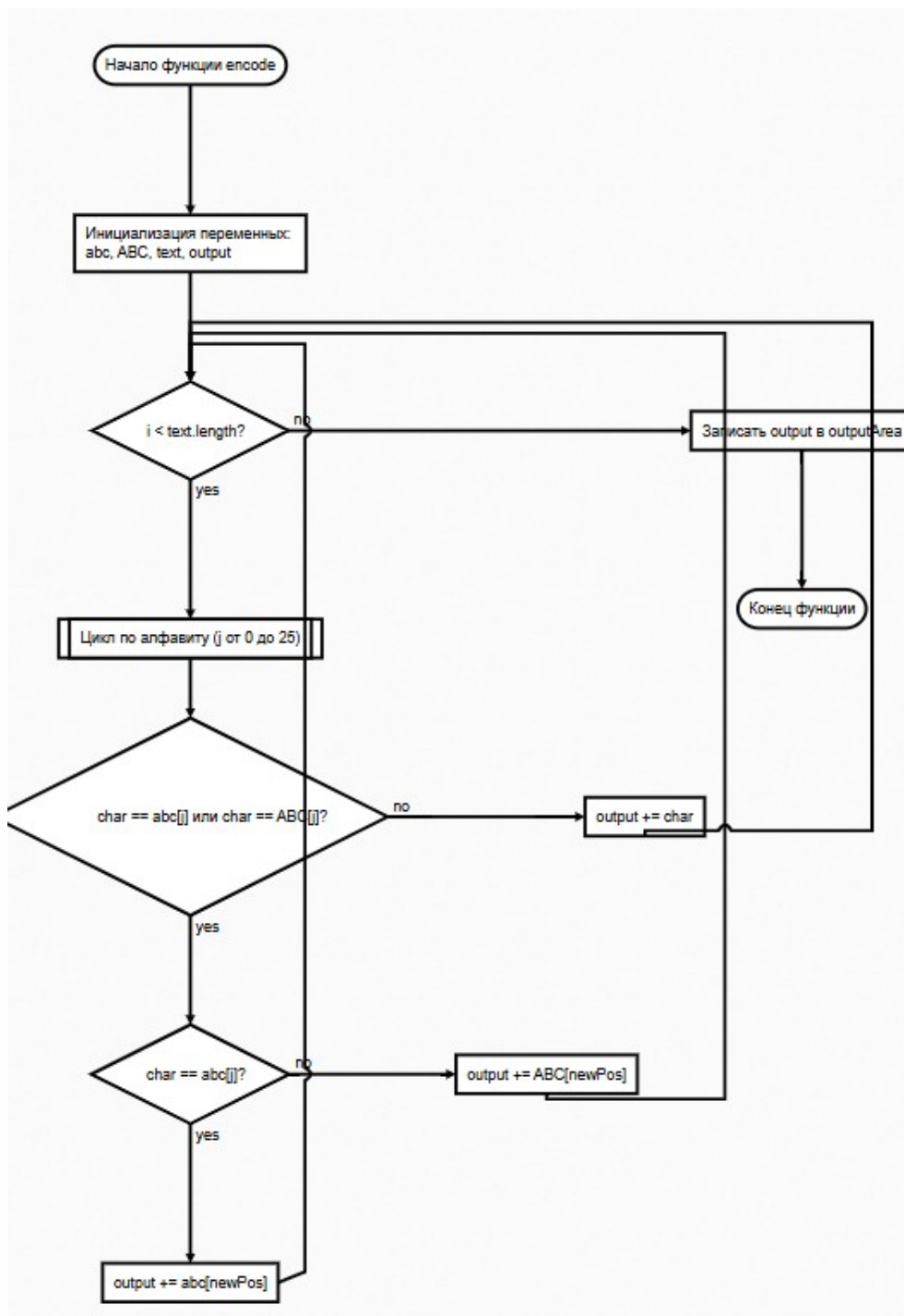


Рисунок 1 — Блок-схема

```
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

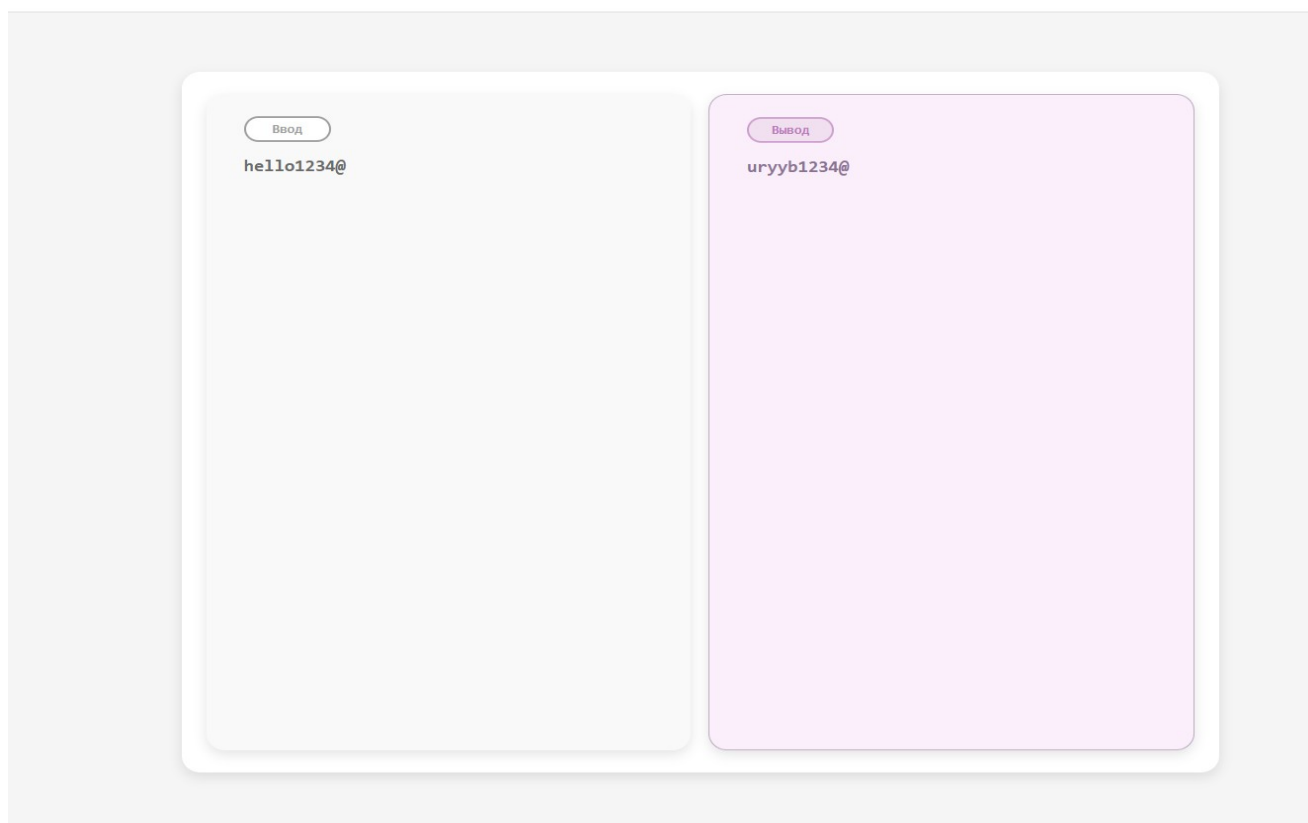


Рисунок 3 — Тестирование шифровка

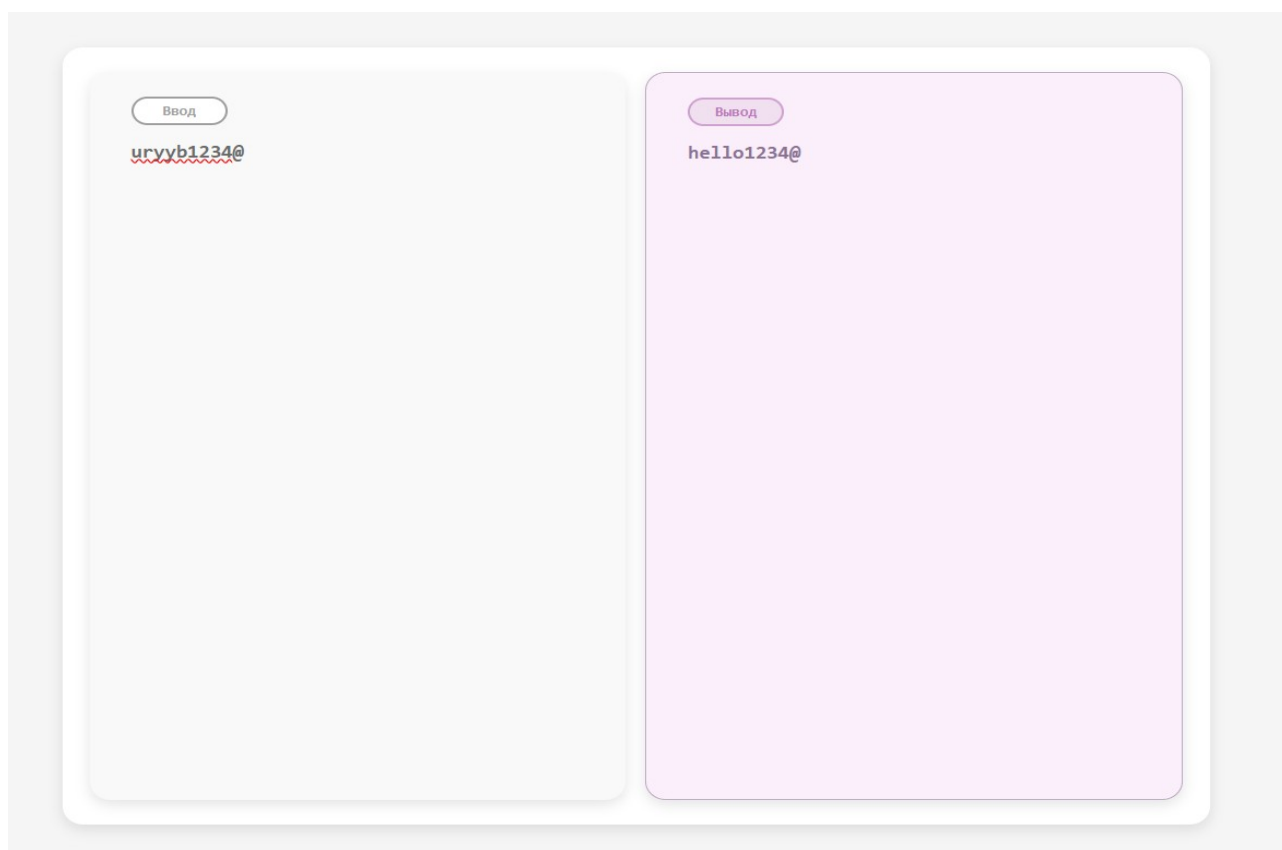


Рисунок 4 — Тестирование дешифровка

## Проверка

Таблица 1: Проверка шифровки

Символ	Тип	Позиция (j)	$\text{newPos} = (j+13)\%26$	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)	$\text{newPos} = (j+13)\%26$	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%



## HTML



```
index.html M X # style.css M JS script.js M
practice > index.html > html > body > main > div.centralBlock
1 <!DOCTYPE html>
2 <html lang="ru">
3 <head>
4   <meta charset="UTF-8">
5   <title>rot13</title>
6   <link rel="stylesheet" href="./style/style.css">
7 </head>
8 <body>
9   <header>
10  </header>
11
12  <main>
13    <div class="centralBlock">
14      <div class="inputPanel">
15        <p>Ввод</p>
16        <textarea id="inputArea"></textarea>
17      </div>
18
19      <div class="outputPanel">
20        <p>Вывод</p>
21        <textarea id="outputArea" readonly></textarea>
22      </div>
23    </div>
24  </main>
25
26  <footer>
27  </footer>
28
29  <script src="./script/script.js"></script>
30 </body>
31 </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

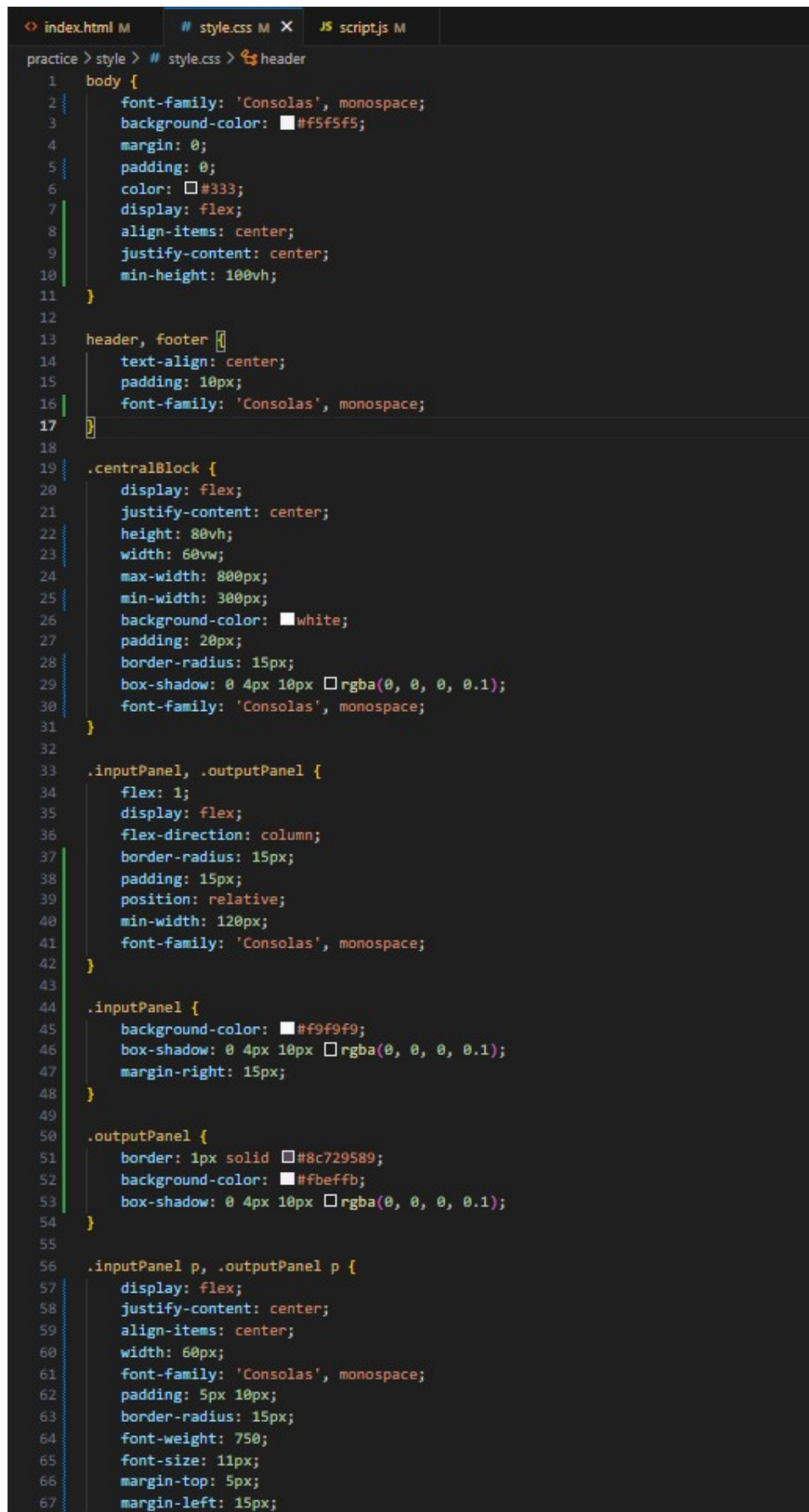
Use the Message Filtering button below to hide/show particular messages, and to see total counts of errors and warnings.

[Message Filtering](#)

Document checking completed. No errors or warnings to show.

Рисунок 6 — Проверка html на валидность

## CSS



The image shows a code editor with three tabs: 'index.html M', 'style.css M', and 'script.js M'. The 'style.css' tab is active, showing the following CSS code:

```
practice > style > # style.css > header
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 }
32
33 .inputPanel, .outputPanel {
34     flex: 1;
35     display: flex;
36     flex-direction: column;
37     border-radius: 15px;
38     padding: 15px;
39     position: relative;
40     min-width: 120px;
41     font-family: 'Consolas', monospace;
42 }
43
44 .inputPanel {
45     background-color: #f9f9f9;
46     box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
47     margin-right: 15px;
48 }
49
50 .outputPanel {
51     border: 1px solid #8c729589;
52     background-color: #fbeffb;
53     box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
54 }
55
56 .inputPanel p, .outputPanel p {
57     display: flex;
58     justify-content: center;
59     align-items: center;
60     width: 60px;
61     font-family: 'Consolas', monospace;
62     padding: 5px 10px;
63     border-radius: 15px;
64     font-weight: 750;
65     font-size: 11px;
66     margin-top: 5px;
67     margin-left: 15px;
```

Рисунок 7 — Стили css

```
index.html M style.css M X JS script.js M
practice > style > # style.css > header
70
71 .inputPanel p {
72     border: 2px solid #9e9e9e;
73     color: #9e9e9e;
74     padding: 1%;
75 }
76
77 .outputPanel p {
78     border: 2px solid #d8a8d0;
79     background-color: #f8e8f8;
80     color: #bd78bd;
81     padding: 1%;
82 }
83
84 textarea {
85     height: 90%;
86     padding: 0 15px;
87     border-radius: 10px;
88     resize: none;
89     font-family: 'Consolas', monospace;
90     font-size: 15px;
91     line-height: 1.5;
92     font-weight: 550;
93     border: none;
94     outline: none;
95     color: #6b6b6b;
96     background-color: transparent;
97 }
98
99 .outputPanel textarea {
100     color: #8c7295;
101     caret-color: transparent;
102 }
103
104 .outputPanel textarea[readonly] {
105     cursor: default;
106 }
107
108 /* Медиа-запросы для адаптивности */
109 @media (max-width: 768px) {
110     .central.block {
111         flex-direction: column;
112         height: auto;
113         width: 85vw;
114         padding: 15px;
115     }
116
117     .inputPanel, .outputPanel {
118         width: 100%;
119         margin-right: 0;
120         margin-bottom: 15px;
121     }
122
123     textarea {
124         height: 200px;
125     }
126 }
127
128 @media (max-width: 480px) {
129     .central.block {
130         width: 95vw;
131         padding: 10px;
132     }
133
134     .inputPanel p, .outputPanel p {
135         width: 50px;
136         font-size: 10px;
137         padding: 4px 8px;
138         margin-left: 10px;
139     }
140
141     textarea {
142         font-size: 14px;
143         padding: 0 10px;
144     }
145 }
```

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>