

```
1 ---
2 title: "Лабораторная работа №1"
3 format:
4   html:
5     toc: true
6     code-fold: true
7     code-line-numbers: true
8     code-tools: true
9
10 ---
11
12 ## Тема: Шифр простой замены
13
14 **Выполнила:** Исламова Сания Маратовна (студ. билет 1132249576)
15
16 ---
17
18 ``` {julia}
19 #| label: cipher-program
20 #| fig-cfp: "Реализация шифров Цезаря и Атбаш"
21 #| code-line-numbers: "true"
22 #| warning: false
23 #| eval: false
24
25 ## Шифр Цезаря
26
27
28
29 function main()
30
31   # Создаем алфавит для шифрования - все русские буквы в нижнем регистре
32
33   alphabet = collect("абвгдеёжзийклмнопрстуфхцчщъыьэюя")
34   # Сохраняем длину алфавита для использования в модульной арифметике
35
36   n = length(alphabet)
37
38   # Бесконечный цикл для работы программы до команды выхода
39   while true
40     # Выводим меню с доступными командами
41     println("Введите Ш чтобы зашифровать сообщение, Р чтобы расшифровать и В
чтобы выйти")
42     print(">>> ") # Приглашение для ввода команды
43
44     # Читаем ввод пользователя, приводим к нижнему регистру и удаляем пробелы
45     menu = lowercase(strip(readline()))
46
47     # Проверяем команду выхода из программы
48     if menu == "в"
49       println("Выход из программы...")
50       break # Прерываем цикл while
51     # Проверяем команду шифрования
52     elseif menu == "ш"
53       operation = "шифрование" # Сохраняем тип операции для вывода
54     # Проверяем команду расшифрования
55     elseif menu == "р"
56       operation = "расшифрование" # Сохраняем тип операции для вывода
57     # Если введена неизвестная команда
58     else
59       println("Неверная команда! Попробуйте снова.")
```

```

60         continue # Переходим к следующей итерации цикла
61     end
62
63     # Запрашиваем строку для обработки
64     print("Введите строку: ")
65     # Читаем строку, приводим к нижнему регистру и удаляем пробелы по краям
66     message = lowercase(strip(readline()))
67
68     # Запрашиваем ключ шифрования
69     print("Введите ключ: ")
70     # Обрабатываем ввод ключа с проверкой ошибок
71     try
72         # Парсим введенную строку в целое число
73         key = parse{Int, readline()}
74     catch e
75         # Если произошла ошибка (введено не число)
76         println("Ошибка: введите целое число для ключа!")
77         continue # Переходим к следующей итерации цикла
78     end
79
80     # Если выбрано расшифрование, инвертируем ключ
81     # (шифрование: сдвиг вперед, расшифрование: сдвиг назад)
82     if menu == "p"
83         key = -key # Умножаем ключ на -1
84     end
85
86     # Инициализируем пустую строку для результата
87     output = ""
88
89     # Проходим по каждому символу введенной строки
90     for letter in message
91         # Ищем позицию текущей буквы в алфавите
92         idx = findfirst(isequal(letter), alphabet)
93
94         # Если буква найдена в алфавите (не пробел, не знак препинания и т.д.)
95         if idx != nothing
96             # Вычисляем новую позицию с учетом ключа и длины алфавита
97             # mod обеспечивает циклический сдвиг (если вышли за границы -
начинаем сначала)
98             # -1 и +1 нужны из-за 1-индексации в Julia (индексы начинаются с 1,
а не с 0)
99             new_idx = mod(idx + key - 1, n) + 1
100             # Добавляем зашифрованную/расшифрованную букву к результату
101             output *= string(alphabet[new_idx])
102         else
103             # Если символ не из алфавита (пробел, запятая и т.д.), добавляем его
как есть
104             output *= string(letter)
105         end
106     end
107
108     # Выводим результат с указанием типа операции
109     println("Результат \$operation: \$output")
110     # Разделительная линия для визуального отделения результатов
111     println("-" ^ 50)
112 end
113 end
114
115 # Запуск программы - вызываем основную функцию
116 main()

```

```

117
118
119
120 ## Шифр Атбаш
121
122
123
124 function main()
125
126     # Создаем русский алфавит для шифрования Атбаш
127     # Атбаш - шифр подстановки, где первая буква заменяется на последнюю,
128     # вторая - на предпоследнюю, и т.д.
129
130     alphabet = collect("абвгдеёжзийклмнопрстуфхцчщъыьэюя")
131     # Сохраняем длину алфавита
132
133     n = length(alphabet)
134
135     # Бесконечный цикл для работы программы до команды выхода
136     while true
137         # Выводим меню с доступными командами
138         println("Введите Ш чтобы зашифровать сообщение, Р чтобы расшифровать и В
чтобы выйти")
139         print(">>> ") # Приглашение для ввода команды
140
141         # Читаем ввод пользователя, приводим к нижнему регистру и удаляем пробелы
142         menu = lowercase(strip(readline()))
143
144         # Проверяем команду выхода из программы
145         if menu == "в"
146             println("Выход из программы...")
147             break # Прерываем цикл while
148         # Проверяем команду шифрования
149         elseif menu == "ш"
150             operation = "шифрование" # Сохраняем тип операции для вывода
151         # Проверяем команду расшифрования
152         elseif menu == "р"
153             operation = "расшифрование" # Сохраняем тип операции для вывода
154         # Если введена неизвестная команда
155         else
156             println("Неверная команда! Попробуйте снова.")
157             continue # Переходим к следующей итерации цикла
158         end
159
160         # Запрашиваем строку для обработки
161         print("Введите строку: ")
162         # Читаем строку, приводим к нижнему регистру и удаляем пробелы по краям
163         message = lowercase(strip(readline()))
164
165         # Инициализируем пустую строку для результата
166         output = ""
167
168         # Проходим по каждому символу введенной строки
169         for letter in message
170             # Ищем позицию текущей буквы в алфавите
171             idx = findfirst(isequal(letter), alphabet)
172
173             # Если буква найдена в алфавите (не пробел, не знак препинания и т.д.)
174             if idx != nothing
175                 # В шифре Атбаш преобразование симметрично:

```

```

176         # Шифрование и расшифрование выполняются одинаково -
177         # буква с индексом i заменяется на букву с индексом (n - i + 1)
178         # Например:
179         # 'а' (1 позиция) -> 'я' (33 позиция при n=33)
180         # 'б' (2 позиция) -> 'ю' (32 позиция)
181         # и т.д.
182         new_idx = n - idx + 1
183         # Добавляем преобразованную букву к результату
184         output *= string(alphabet[new_idx])
185     else
186         # Если символ не из алфавита (пробел, запятая и т.д.), добавляем его
187         как есть output *= string(letter)
188     end
189 end
190
191 # Выводим результат с указанием типа операции
192 println("Результат $operation: $output")
193 # Разделительная линия для визуального отделения результатов
194 println("-" ^ 50)
195 end
196 end
197
198 # Запуск программы - вызываем основную функцию
199 main()

```