

Шифры престановки

Федюшина Ярослава Андреевна

Содержание

Цель работы	4
Задание	5
Выполнение лабораторной работы	5
Вывод	7

Список иллюстраций

1 Маршрут 5

Цель работы

Целью данной лабораторной работы является изучение алгоритмов шифрования перестановки, принцип работы и реализация на языке Julia.

Задание

Реализовать все шифры программно.

Выполнение лабораторной работы

#Маршрутное шифрование

```

Function routeEncryption(text:String, password:String, mimeType:
    ContentType, replace:(lowercase:String) => String) {
    n = ceil(text.length/text / n)
    text = text + "a"*(n - math.floor(text.length/text))
    table = reshape(collect(text), text, n)
    sorted_indices = sortperm(collect(password))
    cipher = Char[]
    for j in sorted_indices
        for i in 1:n
            push!(cipher, table[i, j])
        end
    end
    return join(cipher)
end

text = "нельзя недооценивать противника"
password = "пароль"
n = 6
println(routeEncryption(text, password, n))

```

Рис. 1: Маршрут

#Шифрование с помощью решеток

```

1  # Функция: найти минимальный из трех чисел, используя list()
2  def min_of_three(a, b, c):
3      # Преобразуем числа в список
4      numbers = [a, b, c]
5      # Возвращаем минимальное значение
6      return min(numbers)
7
8  # Пример использования
9  a = 10
10 a = 20
11 a = 30
12 # Вызов функции
13 min_value = min_of_three(a, b, c)
14 # Вывод результата
15 print("Минимальное значение:", min_value)
16
17 # Функция: проверить, является ли число четным
18 def is_even(n):
19     # Возвращаем True, если число четное, иначе False
20     return n % 2 == 0
21
22 # Пример использования
23 n = 10
24 n = 20
25 n = 30
26 # Вызов функции
27 is_even_result = is_even(n)
28 # Вывод результата
29 print("Число является четным:", is_even_result)
30
31 # Функция: подсчитать сумму чисел в списке
32 def sum_of_list(numbers):
33     # Инициализируем переменную для суммы
34     total = 0
35     # Проходим по каждому элементу списка и суммируем
36     for number in numbers:
37         total += number
38     # Возвращаем сумму
39     return total
40
41 # Пример использования
42 numbers = [1, 2, 3, 4, 5]
43 # Вызов функции
44 total_sum = sum_of_list(numbers)
45 # Вывод результата
46 print("Сумма чисел в списке:", total_sum)
47
48 # Функция: найти максимальное значение в списке
49 def max_of_list(numbers):
50     # Инициализируем переменную для максимума
51     max_value = None
52     # Проходим по каждому элементу списка и находим максимум
53     for number in numbers:
54         if max_value is None or number > max_value:
55             max_value = number
56     # Возвращаем максимальное значение
57     return max_value
58
59 # Пример использования
60 numbers = [1, 2, 3, 4, 5]
61 # Вызов функции
62 max_value = max_of_list(numbers)
63 # Вывод результата
64 print("Максимальное значение в списке:", max_value)
65
66 # Функция: проверить, является ли строка палиндромом
67 def is_palindrome(s):
68     # Приводим строку к нижнему регистру и убираем пробелы
69     s = s.lower().replace(" ", "")
70     # Проверяем, равна ли строка своей обратной
71     return s == s[::-1]
72
73 # Пример использования
74 s = "A man a plan a canal Panama"
75 # Вызов функции
76 is_palindrome_result = is_palindrome(s)
77 # Вывод результата
78 print("Строка является палиндромом:", is_palindrome_result)
79
80 # Функция: найти длину списка
81 def list_length(lst):
82     # Возвращаем длину списка
83     return len(lst)
84
85 # Пример использования
86 lst = [1, 2, 3, 4, 5]
87 # Вызов функции
88 length = list_length(lst)
89 # Вывод результата
90 print("Длина списка:", length)
91
92 # Функция: найти сумму квадратов чисел в списке
93 def sum_of_squares(numbers):
94     # Инициализируем переменную для суммы квадратов
95     total = 0
96     # Проходим по каждому элементу списка и суммируем квадраты
97     for number in numbers:
98         total += number ** 2
99     # Возвращаем сумму квадратов
100    return total
101
102 # Пример использования
103 numbers = [1, 2, 3, 4, 5]
104 # Вызов функции
105 sum_squares = sum_of_squares(numbers)
106 # Вывод результата
107 print("Сумма квадратов чисел в списке:", sum_squares)
108
109 # Функция: проверить, является ли строка строкой
110 def is_string(s):
111     # Возвращаем True, если строка строкой, иначе False
112     return isinstance(s, str)
113
114 # Пример использования
115 s = "Hello, World!"
116 s = 123
117 s = True
118 # Вызов функции
119 is_string_result = is_string(s)
120 # Вывод результата
121 print("Строка является строкой:", is_string_result)
122
123 # Функция: найти сумму чисел в диапазоне
124 def sum_range(start, end):
125     # Инициализируем переменную для суммы
126     total = 0
127     # Проходим по каждому числу в диапазоне и суммируем
128     for i in range(start, end + 1):
129         total += i
130     # Возвращаем сумму
131     return total
132
133 # Пример использования
134 start = 1
135 end = 10
136 # Вызов функции
137 sum_range_result = sum_range(start, end)
138 # Вывод результата
139 print("Сумма чисел в диапазоне:", sum_range_result)
140
141 # Функция: найти сумму чисел в диапазоне
142 def sum_range(start, end):
143     # Инициализируем переменную для суммы
144     total = 0
145     # Проходим по каждому числу в диапазоне и суммируем
146     for i in range(start, end + 1):
147         total += i
148     # Возвращаем сумму
149     return total
150
151 # Пример использования
152 start = 1
153 end = 10
154 # Вызов функции
155 sum_range_result = sum_range(start, end)
156 # Вывод результата
157 print("Сумма чисел в диапазоне:", sum_range_result)
158
159 # Функция: найти сумму чисел в диапазоне
160 def sum_range(start, end):
161     # Инициализируем переменную для суммы
162     total = 0
163     # Проходим по каждому числу в диапазоне и суммируем
164     for i in range(start, end + 1):
165         total += i
166     # Возвращаем сумму
167     return total
168
169 # Пример использования
170 start = 1
171 end = 10
172 # Вызов функции
173 sum_range_result = sum_range(start, end)
174 # Вывод результата
175 print("Сумма чисел в диапазоне:", sum_range_result)
176
177 # Функция: найти сумму чисел в диапазоне
178 def sum_range(start, end):
179     # Инициализируем переменную для суммы
180     total = 0
181     # Проходим по каждому числу в диапазоне и суммируем
182     for i in range(start, end + 1):
183         total += i
184     # Возвращаем сумму
185     return total
186
187 # Пример использования
188 start = 1
189 end = 10
190 # Вызов функции
191 sum_range_result = sum_range(start, end)
192 # Вывод результата
193 print("Сумма чисел в диапазоне:", sum_range_result)
194
195 # Функция: найти сумму чисел в диапазоне
196 def sum_range(start, end):
197     # Инициализируем переменную для суммы
198     total = 0
199     # Проходим по каждому числу в диапазоне и суммируем
200     for i in range(start, end + 1):
201         total += i
202     # Возвращаем сумму
203     return total
204
205 # Пример использования
206 start = 1
207 end = 10
208 # Вызов функции
209 sum_range_result = sum_range(start, end)
210 # Вывод результата
211 print("Сумма чисел в диапазоне:", sum_range_result)
212
213 # Функция: найти сумму чисел в диапазоне
214 def sum_range(start, end):
215     # Инициализируем переменную для суммы
216     total = 0
217     # Проходим по каждому числу в диапазоне и суммируем
218     for i in range(start, end + 1):
219         total += i
220     # Возвращаем сумму
221     return total
222
223 # Пример использования
224 start = 1
225 end = 10
226 # Вызов функции
227 sum_range_result = sum_range(start, end)
228 # Вывод результата
229 print("Сумма чисел в диапазоне:", sum_range_result)
230
231 # Функция: найти сумму чисел в диапазоне
232 def sum_range(start, end):
233     # Инициализируем переменную для суммы
234     total = 0
235     # Проходим по каждому числу в диапазоне и суммируем
236     for i in range(start, end + 1):
237         total += i
238     # Возвращаем сумму
239     return total
240
241 # Пример использования
242 start = 1
243 end = 10
244 # Вызов функции
245 sum_range_result = sum_range(start, end)
246 # Вывод результата
247 print("Сумма чисел в диапазоне:", sum_range_result)
248
249 # Функция: найти сумму чисел в диапазоне
250 def sum_range(start, end):
251     # Инициализируем переменную для суммы
252     total = 0
253     # Проходим по каждому числу в диапазоне и суммируем
254     for i in range(start, end + 1):
255         total += i
256     # Возвращаем сумму
257     return total
258
259 # Пример использования
260 start = 1
261 end = 10
262 # Вызов функции
263 sum_range_result = sum_range(start, end)
264 # Вывод результата
265 print("Сумма чисел в диапазоне:", sum_range_result)
266
267 # Функция: найти сумму чисел в диапазоне
268 def sum_range(start, end):
269     # Инициализируем переменную для суммы
270     total = 0
271     # Проходим по каждому числу в диапазоне и суммируем
272     for i in range(start, end + 1):
273         total += i
274     # Возвращаем сумму
275     return total
276
277 # Пример использования
278 start = 1
279 end = 10
280 # Вызов функции
281 sum_range_result = sum_range(start, end)
282 # Вывод результата
283 print("Сумма чисел в диапазоне:", sum_range_result)
284
285 # Функция: найти сумму чисел в диапазоне
286 def sum_range(start, end):
287     # Инициализируем переменную для суммы
288     total = 0
289     # Проходим по каждому числу в диапазоне и суммируем
290     for i in range(start, end + 1):
291         total += i
292     # Возвращаем сумму
293     return total
294
295 # Пример использования
296 start = 1
297 end = 10
298 # Вызов функции
299 sum_range_result = sum_range(start, end)
300 # Вывод результата
301 print("Сумма чисел в диапазоне:", sum_range_result)
302
303 # Функция: найти сумму чисел в диапазоне
304 def sum_range(start, end):
305     # Инициализируем переменную для суммы
306     total = 0
307     # Проходим по каждому числу в диапазоне и суммируем
308     for i in range(start, end + 1):
309         total += i
310     # Возвращаем сумму
311     return total
312
313 # Пример использования
314 start = 1
315 end = 10
316 # Вызов функции
317 sum_range_result = sum_range(start, end)
318 # Вывод результата
319 print("Сумма чисел в диапазоне:", sum_range_result)
320
321 # Функция: найти сумму чисел в диапазоне
322 def sum_range(start, end):
323     # Инициализируем переменную для суммы
324     total = 0
325     # Проходим по каждому числу в диапазоне и суммируем
326     for i in range(start, end + 1):
327         total += i
328     # Возвращаем сумму
329     return total
330
331 # Пример использования
332 start = 1
333 end = 10
334 # Вызов функции
335 sum_range_result = sum_range(start, end)
336 # Вывод результата
337 print("Сумма чисел в диапазоне:", sum_range_result)
338
339 # Функция: найти сумму чисел в диапазоне
340 def sum_range(start, end):
341     # Инициализируем переменную для суммы
342     total = 0
343     # Проходим по каждому числу в диапазоне и суммируем
344     for i in range(start, end + 1):
345         total += i
346     # Возвращаем сумму
347     return total
348
349 # Пример использования
350 start = 1
351 end = 10
352 # Вызов функции
353 sum_range_result = sum_range(start, end)
354 # Вывод результата
355 print("Сумма чисел в диапазоне:", sum_range_result)
356
357 # Функция: найти сумму чисел в диапазоне
358 def sum_range(start, end):
359     # Инициализируем переменную для суммы
360     total = 0
361     # Проходим по каждому числу в диапазоне и суммируем
362     for i in range(start, end + 1):
363         total += i
364     # Возвращаем сумму
365     return total
366
367 # Пример использования
368 start = 1
369 end = 10
370 # Вызов функции
371 sum_range_result = sum_range(start, end)
372 # Вывод результата
373 print("Сумма чисел в диапазоне:", sum_range_result)
374
375 # Функция: найти сумму чисел в диапазоне
376 def sum_range(start, end):
377     # Инициализируем переменную для суммы
378     total = 0
379     # Проходим по каждому числу в диапазоне и суммируем
380     for i in range(start, end + 1):
381         total += i
382     # Возвращаем сумму
383     return total
384
385 # Пример использования
386 start = 1
387 end = 10
388 # Вызов функции
389 sum_range_result = sum_range(start, end)
390 # Вывод результата

```

#Таблица Виженера

```

def cipher_generate_key(key: str) -> str:
    key = key.lower()
    key_length = len(key)
    plain_length = len(plaintext)
    key = key * (plain_length // key_length + 1)
    key = key[:plain_length]
    return key

def cipher_encrypt(plaintext: str, key: str) -> str:
    cipher = ""
    for i in range(len(plaintext)):
        p = plaintext[i]
        k = key[i]
        if p.isalpha():
            if p.isupper():
                cipher += chr((ord(p) - ord('A') + ord(k) - ord('A')) % 26 + ord('A'))
            else:
                cipher += chr((ord(p) - ord('a') + ord(k) - ord('a')) % 26 + ord('a'))
        else:
            cipher += p
    return cipher

def cipher_decrypt(ciphertext: str, key: str) -> str:
    cipher = ""
    for i in range(len(ciphertext)):
        c = ciphertext[i]
        k = key[i]
        if c.isalpha():
            if c.isupper():
                cipher += chr((ord(c) - ord('A') + 26 - ord(k) + ord('A')) % 26 + ord('A'))
            else:
                cipher += chr((ord(c) - ord('a') + 26 - ord(k) + ord('a')) % 26 + ord('a'))
        else:
            cipher += c
    return cipher

```

Вывод

В ходе выполнения данной лабораторной работы были изучены три шифра перестановки, а так же были реализованы на языке Julia.