

Отчёт по лабораторной работе 3

Супонина Анастасия Павловна

Содержание

Цель работы	1
Задание	1
Теоретическое введение	1
Используемые функции.....	2
Выполнение лабораторной работы	2
Выводы.....	4
Список литературы.....	4

Список иллюстраций

‘Введение функции и запись переменных’	2
‘Создание цикла for с побитовым сложением’	3
‘Функция шифровки и дешифровки’	3
‘Вывод переменных’	3
‘Итоговый код’	4
‘Результат выполнения программы’	4

Список таблиц

Элементы списка иллюстраций не найдены.

Цель работы

Изучить шифрование гаммированием и реализовать программу шифрования используя язык программирования Julia.

Задание

Программно реализовать на языке Julia шифрование гаммированием

Теоретическое введение

Шифрование гаммированием — это симметричный метод шифрования, при котором на открытый текст накладывается последовательность, сформированная из случайных чисел.

Важным в шифровании гаммированием является то, что сгенерированный случайно ключ должен быть той же длины, что и сообщение, которое необходимо зашифровать.

В данном виде шифрования используется *побитовое сложение*, а данная операция является обратной, поэтому для того, чтобы расшифровать результат нужно просто прогнать его через шифр с тем же самым ключом.

Пример: $a = 01000011$, $b = 01110010$ тогда $c = a \text{ xor } b$:

```
01000011
01110010
- - - - -
00110001
```

$c = 00110001$, а чтобы получить обратно a , меняем их местами в формуле

```
00110001
01110010
- - - - -
01000011
```

Используемые функции

`randstring` - создает случайную строку из заданных данных, заданной длины `xor` - производит побитовое сложение `codepoint` - возвращает кодовую точку Unicode соответствующую символу

Выполнение лабораторной работы

Создаю переменные k - для записи ключа к шифру и $text$ - для записи сообщения которое нужно зашифровать. Также для создания случайного ключа ввожу функцию **Random** и использую `randstring` ('ввожу переменные из которых делается выбор', 'задаю длину ключа').

```
using Random

k = ""
text = "Have a nice day"
k *= randstring(['A':'Z'; 'a':'z'; '0':'9'], length(text))
```

‘Введение функции и запись переменных’

Для шифрования нам нужно побитовое сложение произвожу его поэлементно при помощи цикла `for`. Само побитовое сложение выполняю при помощи функции `xor`. Также не забываем вернуть элементы в формат строки, используя для этого `Char`.

```

for i in 1:length(text)
    cipher_text_bit = xor(codepoint(text[i]), codepoint(k[i]))
    cipher_text *= Char(cipher_text_bit)
end

```

‘Создание цикла for с побитовым сложением’

Не смотря на то, что создавая ключ через randstring, я задавала длину такую же как у введенного сообщения, для страховки создаю if, который проверяет условие равенства длин сообщения и ключа. Далее вписываю подготовленный цикл for, и добавляю return, чтобы функция возвращала зашифрованный текст.

```

function cipher(text, k) #функция шифрования и дешифровки
    if length(text) == length(k)
        cipher_text = ""
        for i in 1:length(text)
            cipher_text_bit = xor(codepoint(text[i]), codepoint(k[i]))
            cipher_text *= Char(cipher_text_bit)
        end
    end
    return cipher_text
end

```

‘Функция шифровки и дешифровки’

Для вывода переменных я использовала несколько разных способов: printstyled - позволяет меня цвет выводимой записи println - отвечает за перенос следующей записи на новую строку

В выводе дешифрованного текста я отправляю полученный зашифрованный текст обратно в созданную функцию с тем же ключом шифрования.

```

cipher_text = cipher(text, k)

printstyled("Текст:", text; color = :green)
println(" Ключ шифра:", k)
printstyled("Зашифрованный текст:", cipher_text; color = :blue)
printstyled(" Дешифрованный текст:", cipher(cipher_text, k); color = :green)

```

‘Вывод переменных’

Здесь представлен итоговый вид кода.

```
ШифрованиеГаммированием.jl U X
MathSec > ШифрованиеГаммированием.jl > cipher
1 using Random
2
3 k = ""
4 text = "Have a nice day"
5 k *= randstring(['A':'Z'; 'a':'z'; '0':'9'], length(text)) #создание случайного ключа длины текста
6
7 function cipher(text, k) #функция шифрования и дешифровки
8     if length(text) == length(k)
9         cipher_text = ""
10        for i in 1:length(text)
11            cipher_text_bit = xor(codepoint(text[i]), codepoint(k[i]))
12            cipher_text *= Char(cipher_text_bit)
13        end
14    end
15    return cipher_text
16 end
17
18 cipher_text = cipher(text, k)
19
20 printstyled("Текст:", text; color = :green)
21 println(" Ключ шифра:", k)
22 printstyled("Зашифрованный текст:", cipher_text; color = :blue)
23 printstyled(" Дешифрованный текст:", cipher(cipher_text, k); color = :green)
```

PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

Зашифрованный текст:2,?N?y=«QDЛ,. Дешифрованный текст:Have a nice day
Текст:Have a nice day Ключ шифра:KpHkKEU0GY0jrGP
Зашифрованный текст:♥<>Дk\$u^.:UJ=&) Дешифрованный текст:Have a nice day
Текст:Have a nice day Ключ шифра:635BxBeSDA64zGf
Зашифрованный текст:~RC'X#E=-"S!Δ&▼ Дешифрованный текст:Have a nice day
Текст:Have a nice day Ключ шифра:EGP2f5rgIITuxC5
&&wFTRов *1Uu"L Дешифрованный текст:Have a nice day
Текст:Have a nice day Ключ шифра:mZ7b0OVj3vqVF57
Зашифрованный текст:%;A-.v♦Z\$9v"TN Дешифрованный текст:Have a nice day
Текст:Have a nice day Ключ шифра:mQLYHzQtvgKIzW8
Зашифрованный текст:%0:<h>▼\$.iΔ6A Дешифрованный текст:Have a nice day

‘Итоговый код’

После запуска наблюдаю, что программа работает верно, цвета вводного текста и дешифрованного одинаковые и текст тоже одинаковый. Также выполняю несколько раз, чтобы убедиться, что случайная генерация ключа работает и программа также работает с любым из сгенерированных ключей.

```
Текст:Have a nice day Ключ шифра:635BxBeSDA64zGf
Зашифрованный текст:~RC'X#E=-"S!Δ&▼ Дешифрованный текст:Have a nice day
```

‘Результат выполнения программы’

Выводы

В процессе выполнения работы, я разобралась с принципом работы шифрования гаммированием. Изучила новые функции Julia, такие как `randstring`, `xor` и `codepoint`. Реализовала шифрование гаммированием на языке программирования Julia.

Список литературы

... Пособие по лабораторной работе 3 {file:///C:/Users/bermu/Downloads/lab03.pdf}