

---

# Основы информационной безопасности. Обучение в записи

## *Урок 10. Семинар: Безопасная разработка приложений*


---

### Оглавление

Задание 1.....	2
Задание 2.....	4
Задание 3.....	6
Задание 4.....	8
Задание 5.....	9
Задание 6.....	10
Домашнее задание: .....	11


## Задание 1

Семинар 5. Безопасная разработка приложений



### Задание №1 «Input validation»

💡 Вам нужно сделать валидацию входных данных (Input validation) для умножения данных в заявке-заказе



5 минут

```
#Input validation
x = int(input('1st factor:'))
y = int(input('2nd factor:'))
print('Total =', x*y)

1st factor:5
2nd factor:4
Total = 20
```

---

```
def error_checking():
    while type:
        x = input('1st factor: ')
        y = input('2nd factor: ')

    try:
        x = int(x)
        y = int(y)
        while (x <= 0 or y <= 0 or x > 1000 or y > 1000):
            print("Заказ не может быть меньше/равен 0 или больше 1000. Повторите ввод:")
            x = int(input('1st factor: '))
            y = int(input('2nd factor: '))

    except ValueError:
        print('Вы ввели некорректное значение '
              '(данные должны включать только натуральные числа без пробелов).\nПовторите ввод:')
    else:
        break

    return print('Total = ', x * y)

print(error_checking())
```

```
task_01.py x task_02.py x task_03.py x task_04.py x task_05.py x task_06.py x homework.py x
1 def error_checking():
2     while type:
3         x = input('1st factor: ')
4         y = input('2st factor: ')
5
6         try:
7             x = int(x)
8             y = int(y)
9             while (x <= 0 or y <= 0 or x > 1000 or y > 1000):
10                 print('Заказ не может быть меньше/равен 0 или больше 1000. Повторите ввод:')
11                 x = int(input('1st factor: '))
12                 y = int(input('2st factor: '))
13
14         except ValueError:
15             print('Вы ввели некорректное значение '
16                   '(данные должны включать только натуральные числа без пробелов).\nПовторите ввод:')
17         else:
18             break
19
20     return print('Total = ', x * y)
21
22 print(error_checking())
23
error_checking() > while type > try > while (x <= 0 or y <= 0 or x > ...
Run: task_01 x
C:\Users\user\Desktop\GeekBrains-specialization-ELECTIVES\13. Fundamentals of Information Security\Seminar-05\Python\task_01.py
1st factor: 0
2st factor: 2
Заказ не может быть меньше/равен 0 или больше 1000. Повторите ввод:
1st factor: 20000
2st factor: 3
Заказ не может быть меньше/равен 0 или больше 1000. Повторите ввод:
1st factor: *
Вы ввели некорректное значение (данные должны включать только натуральные числа без пробелов).
Повторите ввод:
1st factor: 2
2st factor: 1000
Total = 2000
```

## Задание 2

Семинар 5. Безопасная разработка приложений


Задание №2 «Input validation»

Вам нужно сделать валидацию входных данных (Input validation) и санитизацию выходных данных (Sanitize) в команды к операционной системе при создании каталогов для хранения материалов заявок-заказов

```
#Input validation + Output encoding (sanitization)
import os
input_path = input('Catalogue path:')
command = f'mkdir {input_path}'
os.popen(command)

Catalogue path:C:\\_WORK\\TEST
```

C:\\_WORK	
Имя	Тип
TEST	Папка с файлами



10 минут

Семинар 5. Безопасная разработка приложений

Задание №2 (проблематика)

```
#Input validation + Output encoding (sanitization)
import os
input_path = input('Catalogue path:')
command = f'mkdir {input_path}'
os.popen(command)

Catalogue path:C:\\_WORK\\test1 & %windir%\\system32\\notepad.exe
```

We found 1 issues in your code

0 high severity 1 medium severity 0 low severity

**M** Command Injection

VULNERABILITY | CVE-22

```
#Input validation + Output encoding (sanitization)
import os
input_path = input('Catalogue path:')
command = f'mkdir {input_path}'
os.popen(command)
```

Unsanitized input from user input flows into os.popen, where it is used as a shell command. This may result in a Command Injection vulnerability.

Безымянный — Блокнот

Файл Правка Формат Вид Справка

\* <https://snvk.io/code-checker/python/>

*import os*

*input\_path = input('Catalog path: ')*

*spec\_symbols = ['\*', '?', '<', '>', '|', '&']*

*check = [characters in input\_path for characters in spec\_symbols]*

*while True in check:*

*for i in range(len(check)):*

*check[i] = False*

*print('Incorect catalogue path')*

*input\_path = input('Catalog path: ')*

*check = [characters in input\_path for characters in spec\_symbols]*

*command = f'mkdir {input\_path}'*

*os.popen(command)*

*print('Catalogue was successfully created')*

```
Python - task_02.py
task_01.py x task_02.py x task_03.py x task_04.py x task_05.py x task_06.py x homework.py x
1 import os
2
3 input_path = input('Catalog path: ')
4 spec_symbols = ['*', '?', '<', '>', '|', '&']
5 check = [characters in input_path for characters in spec_symbols]
6 while True in check:
7     for i in range(len(check)):
8         check[i] = False
9     print('Incorrect catalogue path')
10    input_path = input('Catalog path: ')
11    check = [characters in input_path for characters in spec_symbols]
12 command = f'mkdir {input_path}'
13 os.popen(command)
14 print('Catalogue was successfully created')
```

Run: task\_02

```
"C:\Users\user\Desktop\GeekBrains-specialization-ELECTIVES\13. Fundamentals of Information Security\Seminar-05\Python\Scripts\python.exe"
"C:\Users\user\Desktop\GeekBrains-specialization-ELECTIVES\13. Fundamentals of Information Security\Seminar-05\Python\task_02.py"
Catalog path: C:\Users\user\Desktop\test2 & C:\Users\user\Desktop\test3
Incorrect catalogue path
Catalog path: C:\Users\user\Desktop\test2
Catalogue was successfully created

Process finished with exit code 0
```

## Задание 3

Семинар 5. Безопасная разработка приложений

### Задание №3 «Input validation»

Вам нужно сделать валидацию входных данных (Input validation) для блока с выполнением произвольного кода\*

```
#Input validation (Arbitrary Code Execution)
compute_user_input = input('\nFactors and operator for computing: ')
if not compute_user_input:
    print("No input")
else:
    print("Result: ", eval(compute_user_input))
```

Factors and operator for computingte: 5\*4  
Result: 20

5 минут

\* Необходимость динамического выполнения кода (3 встроенные функции: `eval()`, `exec()` и `compile()`)

Семинар 5. Безопасная разработка приложений

### Задание №3 (проблематика)

```
#Input validation (Arbitrary Code Execution)
compute_user_input = input('\nFactors and operator for computing: ')
if not compute_user_input:
    print("No input")
else:
    print("Result: ", eval(compute_user_input))
```

We found 1 issues in your code

**Code Injection**

Unsantitized input from user input flows into eval, where it is executed as Python code. This may result in a Code Injection vulnerability.

Type something here to compute: `__import__('os').popen('dir').read()`

Result: `<#rfa[E~*f I IëE C:\Users\...\Python_labs`

\* <https://snyk.io/code-checker/python/>

```
compute_user_input = input('\nFactors and operator for computing: ')
if not compute_user_input:
    print('No input')
else:
    print('Result: ', eval(compute_user_input, {'_builtins_':{}}))
```

'''

### Предупреждение:

Использование `eval()` может быть небезопасно, особенно при обработке пользовательских вводов, так как пользователь может ввести строку кода, которая может выполнить произвольный код на вашей машине. Поэтому рекомендуется быть осторожным и избегать использования `eval()` при обработке пользовательских вводов, если вы не уверены в их безопасности.

### Альтернативы `eval()`:


*Если вам нужно выполнить код, который не является выражением, то лучше использовать `exec()`, который позволяет выполнять произвольный код, но не возвращает результат. Если вам нужно скомпилировать строку кода и выполнить ее позже, то можно использовать функцию `compile()`, которая компилирует строку кода в объект кода, который можно затем выполнить с помощью `exec()` или `eval()`.*


'''

## Задание 4

Семинар 5. Безопасная разработка приложений

Задание №4 «Output encoding»

 Вам нужно сделать унифицированное преобразование выходных данных (Output encoding)

  
5 минут

```
#Output encoding
Name_input = input('\nFirst and last name: ')
print(f'Your data: {Name_input}')
```

First and last name: jon jones  
Your data: jon jones

```
#Output encoding
Name_input = input('\nFirst and last name: ')
print(f'Your data: {Name_input}')
```

First and last name: JOn JOnes  
Your data: JOn JOnes

```
first_name = input('Please enter your first name: ')
last_name = input('Please enter your last name: ')
print('Your data:', first_name.capitalize(), last_name.capitalize())
```

```
01.py x task_02.py x task_03.py x task_04.py x task_05.py x task_06.py x homework.py x
Задание №4 «Output encoding»

Вам нужно сделать унифицированное преобразование выходных данных
(Output encoding)

'''

first_name = input('Please enter your first name: ')
last_name = input('Please enter your last name: ')
print('Your data:', first_name.capitalize(), last_name.capitalize())

task_04 x
"C:\Users\user\Desktop\GeekBrains-specialization-ELECTIVES\13. Fundamentals of Information
Security\Seminar-05\Python\Scripts\python.exe"
"C:\Users\user\Desktop\GeekBrains-specialization-ELECTIVES\13. Fundamentals of Information
Security\Seminar-05\Python\task_04.py"
Please enter your first name: IVAN
Please enter your last name: IVANov
Your data: Ivan Ivanov
```



## Задание 5

Семинар 5. Безопасная разработка приложений

### Задание №5 «Error handling and logging»

Вам нужно расследовать попытки несанкционированного доступа при создании каталогов для хранения материалов заявок-заказов (Error handling and logging) – создать Log-файл

test\_log.log — Блокнот

Файл Правка Формат Вид Справка

INFO:root:202:16.324175 Suspicious value: C:\\\_WORK\\test?  
INFO:root:202:34.835209 Suspicious value: C:\\\_WORK\\test?  
INFO:root:202:59.388801 Suspicious value: C:\\\_WORK\\test & python --version  
INFO:root:202:10.671440 Suspicious value: C:\\\_WORK\\test & %windir%\\system32\\notepad.exe

```
import os
import datetime
import logging

logging.basicConfig(level=logging.DEBUG, filename='test_log.log', filemode='w')

input_path = input('Catalogue path: ')
spec_symbols = ['*', '?', '<', '>', '|', '&']
check = [characters in input_path for characters in spec_symbols]
while True in check:
    for i in range(len(check)):
        check[i] = False
    logging.info(f'{datetime.datetime.now()} Suspicious value: {input_path}')
    print('Incorect catalogue path')
    input_path = input('Catalog path: ')
    check = [characters in input_path for characters in spec_symbols]
command = f'mkdir {input_path}'
os.popen(command)
print('Catalogue was successfully created')
```

```
import os
import datetime
import logging

logging.basicConfig(level=logging.DEBUG, filename='test_log.log', filemode='w')

input_path = input('Catalogue path: ')
spec_symbols = ['*', '?', '<', '>', '|', '&']
check = [characters in input_path for characters in spec_symbols]
while True in check:
    for i in range(len(check)):
        check[i] = False
    logging.info(f'{datetime.datetime.now()} Suspicious value: {input_path}')
    print('Incorect catalogue path')
    input_path = input('Catalog path: ')
    check = [characters in input_path for characters in spec_symbols]
command = f'mkdir {input_path}'
os.popen(command)
print('Catalogue was successfully created')
```

task\_05

"C:\Users\User\Desktop\GeekBrains-specialization-ELECTIVES\13. Fundamentals of Information Security\Seminar-05\Python\Scripts\python.exe"  
"C:\Users\User\Desktop\GeekBrains-specialization-ELECTIVES\13. Fundamentals of Information Security\Seminar-05\Python\task\_05.py"  
Catalogue path: C:\Users\User\Desktop\test  
Catalogue was successfully created

## Задание 6

Семинар 5. Безопасная разработка приложений



### Задание №6 «Authentication and password management»



Вам нужно безопасно сохранить пароли пользователей (Authentication and password management)

```
#Authentication and password management
dict_users = {'user1': 'password1', 'user2': 'password2'}
print(dict_users)

{'user1': 'password1', 'user2': 'password2'}
```

*import hashlib*

*dict\_users = {'user\_1': 'password\_1', 'user\_2': 'password\_2'}*

*for i in dict\_users:*

*dict\_users[i] = hashlib.sha256(dict\_users[i].encode()).hexdigest()*

*print(dict\_users)*

```
'''
Задание №6 «Authentication and password management»

Вам нужно безопасно сохранить пароли пользователей (Authentication and password management)
'''
import hashlib

dict_users = {'user_1': 'password_1', 'user_2': 'password_2'}
for i in dict_users:
    dict_users[i] = hashlib.sha256(dict_users[i].encode()).hexdigest()
print(dict_users)
```


dict\_users

```
task_06
"C:\Users\user\Desktop\GeekBrains-specialization-ELECTIVES\13. Fundamentals of Information Security\Seminar-05\Python\Scripts\python.exe"
"C:\Users\user\Desktop\GeekBrains-specialization-ELECTIVES\13. Fundamentals of Information Security\Seminar-05\Python\task_06.py"
{'user_1': '38c5ae2bcd1f12aa269e45ae8c8762f030630a5137dd6d1c799c019626f33096', 'user_2': 'ea76b1e251a0c876b3d96d2c81f12736df3bed36ecb293f79c493c089924cbdc'}

Process finished with exit code 0
```

## Домашнее задание:

Семинар 5. Безопасная разработка приложений




### Домашнее задание

💡 Напишите программу на Python, которая будет проверять вводимый пользователем пароль на сложность (Authentication and password management):

- не менее 8 символов
- наличие прописных и строчных букв
- наличие цифр
- и переводит его в хэш-значение

💡 Результат: файл формата .py или .ipynb



```
import hashlib
import re
```

```
def password_check(password):
    count = 0
    for char in password:
        if 'a' <= char <= 'я' or 'A' <= char <= 'Я':
            print('Пароль должен содержать только латинские буквы!')
            count += 1
            break
    if not re.search(r'\d', password):
        print('Пароль должен содержать не менее одного числа!')
        count += 1
    if len(password) < 8:
        print('Длина пароля должна быть не меньше 8 символов!')
        count += 1
    if not re.search(r'[A-Z]', password):
        print('Пароль должен содержать не менее одной прописной буквы!')
        count += 1
    if not re.search(r'[a-z]', password):
        print('Пароль должен содержать не менее одной строчной буквы!')
        count += 1
    if count > 0:
        return '\n-----\n' \
            'Пароль НЕ соответствует требованиям надежности!\n' \
            '-----'
    else:
        print('Сложность пароля удовлетворяет требованиям сервиса!\nХэш-значение пароля: ')
        return hashlib.sha256(password.encode()).hexdigest()
```

```

passw = input('\nТребования к паролю: '
              'не менее 8 символов, наличие прописных и строчных латинских букв, '
              'наличие цифр'
              '\nВведите пароль: ')

print(password_check(passw))

```

The screenshot shows a Python IDE with a file named `homework.py` open. The code defines a `password_check` function that validates a password based on several criteria: it must contain only Latin letters, at least one digit, be at least 8 characters long, and contain at least one uppercase and one lowercase letter. If the password fails any of these checks, the function returns a message indicating it does not meet the requirements. If it passes all checks, it prints a success message and returns the SHA-256 hash of the password.

```

14 import hashlib
15 import re
16
17
18 def password_check(password):
19     count = 0
20     for char in password:
21         if 'a' <= char <= 'я' or 'A' <= char <= 'Я':
22             print('Пароль должен содержать только латинские буквы!')
23             count += 1
24             break
25     if not re.search(r'\d', password):
26         print('Пароль должен содержать не менее одного числа!')
27         count += 1
28     if len(password) < 8:
29         print('Длина пароля должна быть не меньше 8 символов!')
30         count += 1
31     if not re.search(r'[A-Z]', password):
32         print('Пароль должен содержать не менее одной прописной буквы!')
33         count += 1
34     if not re.search(r'[a-z]', password):
35         print('Пароль должен содержать не менее одной строчной буквы!')
36         count += 1
37     if count > 0:
38         return '\n-----\n' \
39             'Пароль НЕ соответствует требованиям надежности!\n' \
40             '-----'
41     else:
42         print('Сложность пароля удовлетворяет требованиям сервиса!\nХэш-значение пароля: ')
43         return hashlib.sha256(password.encode()).hexdigest()
44
45 password_check()

```

The Run console shows the following output:

```

Требования к паролю: не менее 8 символов, наличие прописных и строчных латинских букв, наличие цифр
Введите пароль: nju$444www
Сложность пароля удовлетворяет требованиям сервиса!
Хэш-значение пароля:
24f770b174bc720e3a91f21a224b07e22f07bfcf9405f93bf6b8a1a6cdf85060

```

---

Ссылка на репозиторий:  
<https://github.com/olgashenkel/GeekBrains-specialization-ELECTIVES/tree/main/13.%20Fundamentals%20of%20Information%20Security>