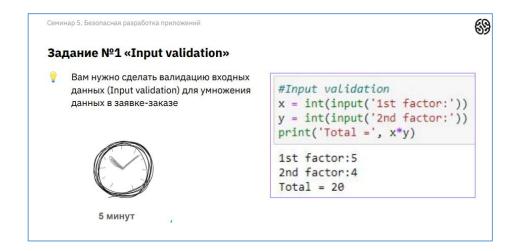
# Основы информационной безопасности. Обучение в записи

Урок 10. Семинар: Безопасная разработка приложений

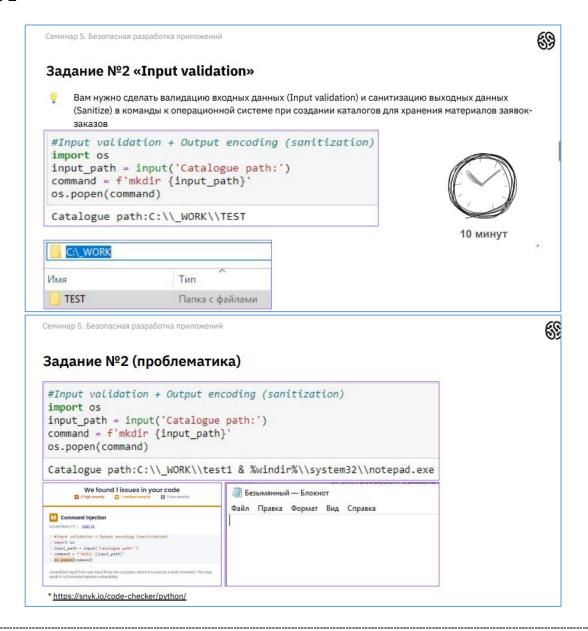
# Оглавление

Задание 1	2
Задание 2	
Задание 3	6
Задание 4	8
Задание 5	9
Задание 6	10
Помашиее запание:	11



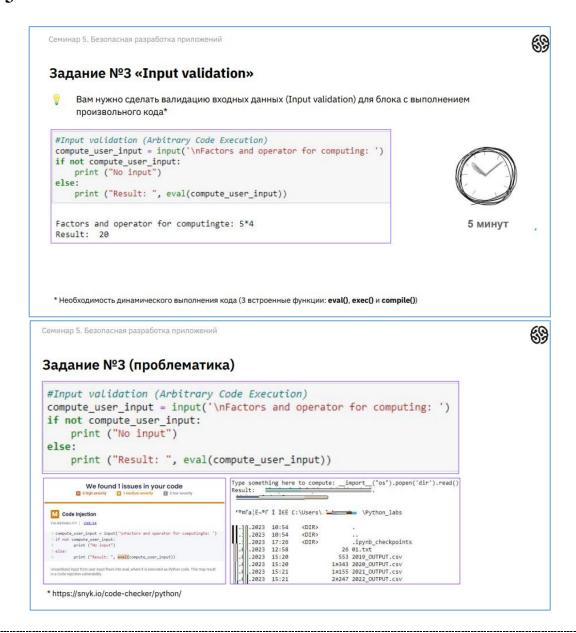
```
def error_checking():
  while type:
     x = input('1st factor:')
     y = input('2st factor: ')
     try:
       x = int(x)
       y = int(y)
       while (x \le 0 \text{ or } y \le 0 \text{ or } x > 1000 \text{ or } y > 1000):
          print('Заказ не может быть меньше/равен 0 или больше 1000. Повторите ввод:')
          x = int(input('1st factor: '))
          y = int(input('2st factor: '))
     except ValueError:
       print('Вы ввели некорректное значение '
           '(данные должны включать только натуральные числа без пробелов). \пПовторите ввод:')
     else:
       break
  return print('Total = ', x * y)
print(error_checking())
```

```
| Code | error_checking(): ReaderMode | while type: | x = input('1st factor: ') | y = input('2st factor: ') | y = input('2st factor: ') | y = inft(x) | y = int(x) | y = int(x) | y = int(x) | y = int((input('1st factor: ')) | x = int(input('1st factor: ')) | x = int(input('1st factor: ')) | y = int(input('2st factor: ') | y = int(input('2st factor: ')) | y = in
```



#### 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')
```



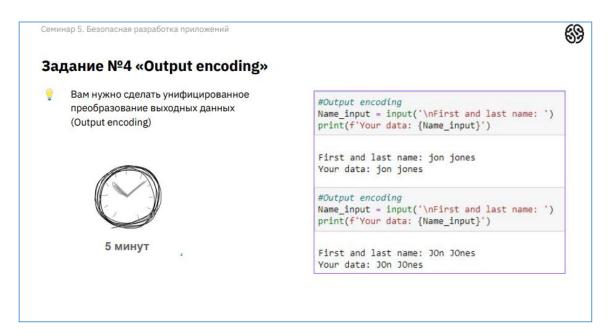
```
compute_user_input = input('\nFactors and operator for computing: ')
if not compute_user_input:
    print('No input')
else:
    print('Result: ', eval(compute_user_input, {'_buitins_':{}}))
```

#### 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.

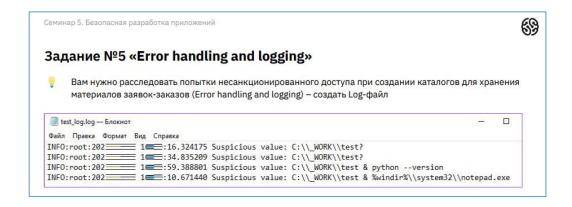
Альтернативы eval():

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().



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())

```
| Compact | Com
```



```
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
eimport 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\Scripts\python.exe"
        "C:\Users\user\Desktop\GeekBrains-specialization-ELECTIVES\13. Fundamentals of Information
        Security\Seminar-05\Python\task.o5.py"
        Catalogue was successfully created
        Catalogue was successfully created
```

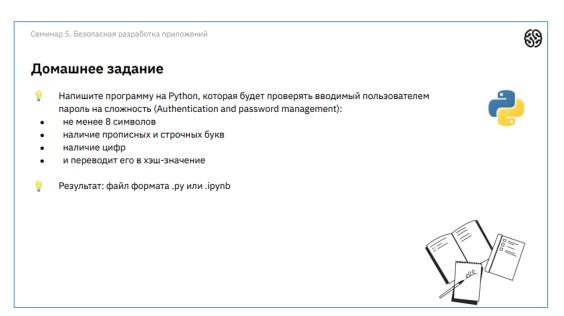


import hashlib

```
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)
```

```
COLPY X to task O2.py X to task O3.py X to task O4.py X to task O5.py X to task O5.py X to task O5.py X to task O6.py X to ta
```

# Домашнее задание:



import hashlib import re

```
def password_check(password):
  count = 0
  for char in password:
    if'a' \le char \le 'g' \ or' A' \le char \le 'H':
      print('Пароль должен содержать только латинские буквы!')
       count += 1
       break
  if not re.search(r'\backslash 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 \(\lambda_{----\}\)\(n'\)
         'Пароль HE coomветствует требованиям надежности!\n'\
  else:
    print('Сложность пароля удовлетворяет требованиям сервиса!\nXэш-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))

```
def password_check(password):
           for char in password:
password_check() > if not re.search(r'\d', passwor..
       Требования к паролю: не менее 8 символов, наличие прописных и строчных латинских букв, наличие цифр
       Хэш-значение пароля:
```