Дополнительный материал к работе №3

Задание 1. Подлинность пароля

Вам поручено разработать подсистему регистрации и авторизации для ресторана «Быстро и запятая».

Разобьем задачу на подзадачи.

1) Определим необходимые структуры и импортируемые библиотеки. Будем хранить пару логин-пароль в словаре, а список допустимых специальных символов – в множестве. Для работы с файлом нам потребуется библиотека pickle.

```
users = {}
symb = {'_', '!', '#', '$', '%', '&', '@', '-'}
import pickle
```

2) Создадим функцию для определения корректности имени пользователя, задаваемого при регистрации. Необходимые действия поясняет схема алгоритма на рисунке 1.

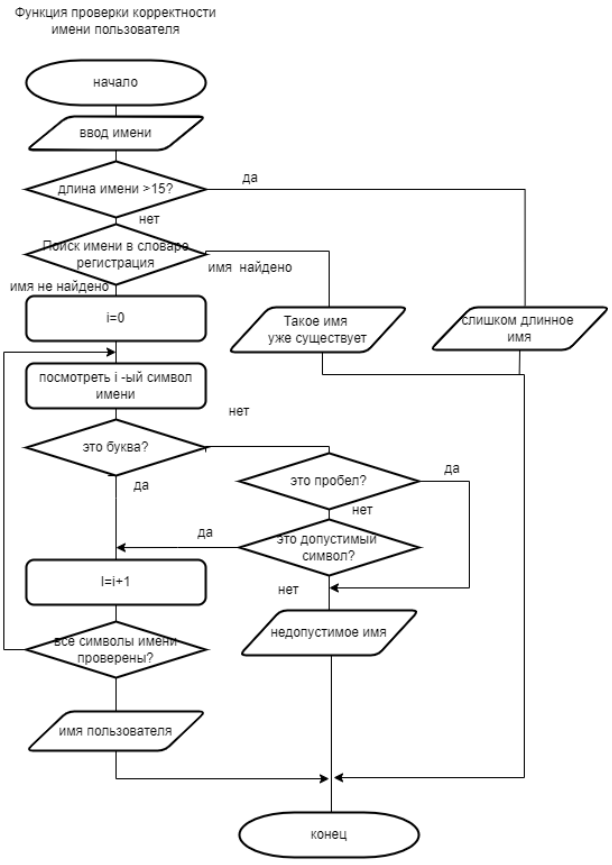

Рисунок 1 – Схема алгоритма

```
def correct_name ():
        '''функция проверки правильности имени пользователя'''
```

```
    login = input('User ')
    if len(login) > 15:
        print('Слишком длинное имя пользователя. Имя не должно превышать 15 символов')
        return 0
    elif login in users:
        print('Пользователь с таким именем уже зарегистрирован')
        return 0
    else:
        for i in login:
            if not(i.isalpha() or (i in symb)) or i.isspace() :
                print('Недопустимое имя пользователя. Имя пользователя может состоять из букв латинского алфавита и специальных символов. Попробуйте еще раз.')
                return 0
    return login
```

3) Создадим функцию определения корректности создаваемого при регистрации пароля.

```
def correct_password():
    '''функция проверки правильности пароля'''
    alflower, alfupper, symbol, digit, nosumb = False, False, False, False, True
    '''переменные для проверки состава пароля: буквы заглавные и строчные, допустимые символы, цифры'''
    password = input('Введите пароль ')
    if len(password) < 10:
        print('Слишком короткий пароль. Попробуйте еще раз. ')
        return 0
    else:
        for i in password:
            if i.isalpha() and i.islower():
                alflower = True
            elif i.isalpha() and i.isupper():
                alfupper = True
            elif i in symb:
                symbol = True
            elif i.isdigit():
                digit = True
            else:
                nosumb = False
                break
        if alflower and alfupper and symbol and digit and nosumb :
            return password
        else:
            print('Неверный формат пароля. Пароль должен состоять из прописных и строчных букв, цифр и спецсимволов. Попробуйте еще раз.')
            return 0
```

4) Определим действия, происходящие при регистрации и авторизации с учетом того, что пользователю предоставляется не более трех попыток для ввода имени и пароля.

```
def registr():
    '''функция регистрации пользователя'''
    print('Заполните форму регистрации')
    print('Введите имя пользователя')
    for count in range(3):
        login = correct_name()
        if login == 0:
            continue
        else:
            password = correct_password()
            if password != 0:
                users[login] = password
                print('Регистрация успешно завершена.')
                with open('pas.txt', 'wb') as file:
                    pickle.dump(users, file)
                print(users)
                break
            else:
                continue

def author():
    '''функция авторизации зарегистрированного пользователя'''
    for i in range(3):
        login = input('Введите имя пользователя')
        password = input('Введите пароль')
        if (login in users):
            if users[login] == password:
                print('Добро пожаловать!')
                break
            elif users[login].lower() == password.lower():
                print('Неверное имя пользователя или пароль')
                print('Проверьте, не нажат ли CapsLock.')
            else:
                print('Неверное имя пользователя или пароль')
        else:
            print('Неверное имя пользователя или пароль')
```

5) Основная программа будет вызывать функции авторизации или регистрации в зависимости от первоначального ответа пользователя.

```
'''основная программа'''
ans = input('Добрый день! Вы зарегистрированы в системе? (да/нет)')
try:
    with open('pas.txt', 'rb') as file:
        users = pickle.load(file)
        print(users)
except IOError:
    print('Файл не найден')
if ans == 'да':
    author()
else:
    registr()
```

Задание 2. «Допилим» расписание

На базе программ «Мое расписание» из предыдущих лабораторных работ, создадим итоговый вариант.

```
import pickle

def add_sbj():
	day_for_add = input('Введите день ')
	subj = input('Введите занятие ')
	if day_for_add in shedule:
		print('Добавлено занятие в существующий день')
		shedule[day_for_add].append(subj)
	else:
		print('Добавлен новый день в расписание')
		shedule[day_for_add] = [subj]
	print (shedule)


def show_day():
	day_for_find = input('Выберите день ')
	if day_for_find in shedule:
		for i in shedule[day_for_find]:
			print(i)
	else:
		print('На этот день нет планов')


def find_subj():
	flag = 0
	subj = input('Введите занятие ')
	for i in shedule:
		for j in shedule[i]:
			if j == subj:
				print('День проведения занятия – ', i)
				flag = 1
				break
		if flag == 1:
			break
	else:
		print('Такая задача не запланирована')

def info():
	print('Список допустимых команд',  'help - вызов справки по командам,', 'add - добавить занятие на заданный день,', 'find - определить в какой день поставлено в расписание заданное занятие,', 'show - посмотреть расписание на заданный день,', 'save - сохранить расписание в файл,', 'end - завершение работы программ', sep = '\n')


shedule = {}
info()
```

```
try:
        '''Контролируемый блок. Определяет возможность возникновения ошибок при выполнении команд'''
        with open('out.txt', 'rb') as file:
                shedule = pickle.load(file)
except IOError:
        '''Выполняется только в случае, если в блоке try произошла ошибка'''
        print('Файл не найден')

while True:
        command = input('Введите команду ')
        if command == 'add':
                add_sbj()
        elif command == 'help':
                info()
        elif command == 'show':
                show_day()
        elif command == 'find':
                find_subj()
        elif command == 'save':
                with open("out.txt", "wb") as file:
                        pickle.dump(shedule,file)
        elif command == 'end':
                break
        else:
                print('Такой команды нет')
```