

Министерство образования и науки Российской Федерации
Федерально государственное автономное образовательное
учреждение высшего образования
«Санкт-Петербургский национальный исследовательский
университет информационных технологий, механики и оптики»




Мегафакультет: Компьютерных технологий и управления
Факультет: Безопасности информационных технологий
Кафедра: Проектирования и безопасности компьютерных систем
Направление (специальность): 10.03.01 «Информационная безопасность»

Лабораторная работа №3
на тему
«Формирование счета на оплату услуг»
ВАРИАНТ № 3

Выполнил:

Студент гр.N3352

/Распутина А.А.

Проверил:

Федоров Иван Романович

Санкт-Петербург

2020 г.

Цель работы:

Программно реализовать автоматическое формирование Счета на оплату в формате .pdf на основании данных из л/р 1 и 2.

IP-адрес 192.168.250.27

Номер телефона 915783624

Задание:

Вариант № 3

По полученным результатам тарификации услуг «Телефония» и «Интернет» в лабораторных работах 1, 2 сформировать счет на оплату в формате .pdf.

Все поля печатной формы должны заполняться разработанным программным модулем. Название банка, имена покупателей и прочие формальные поля можно заполнить любыми значениями. Стоимость услуг берется из предыдущих двух работ.

В качестве результата работы необходимо представить программный модуль для генерации печатной формы счета на оплату и полученный файл .pdf. Средства реализации выбираются студентом самостоятельно.

Практическая часть

Для корректной работы программы необходимо загрузить определенные программные модули и пакеты для Python.

Pip - это система управления пакетами, которая используется для установки и управления программными пакетами, написанными на Python. Прежде чем с помощью pip устанавливать python-пакеты, нужно сначала установить сам pip (при его отсутствии. В поздних версиях Python pip включен в поставку)

```
> sudo apt-get install python3-pip
```

С помощью модуля python-docx можно создавать и изменять документы MS Word с расширением .docx. Чтобы установить этот модуль, необходимо выполнить команду

```
> pip3 install python-docx  
(pip install docx)
```

Также необходимо установить модуль python docxtpl для работы с шаблонами файлов:

```
> pip3 install docxtpl  
(pip install docxtpl)
```

Альтернативный вариант установки всех пакетов – импорт через ide – для этого нужно нажать на импортируемую библиотеку и выбрать “install package \$name”:

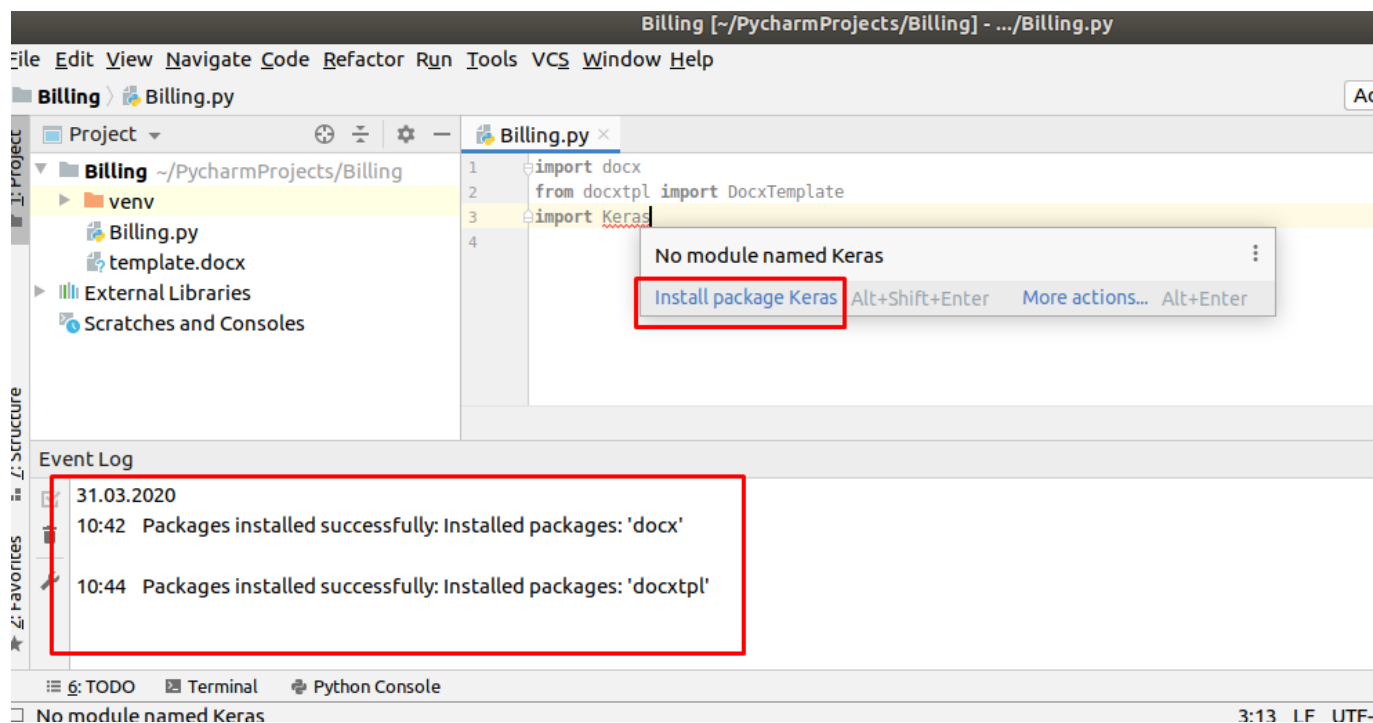


Рис. 1 – Импорт программных пакетов в PyCharm

После загрузки пакетов необходимо обновить пакеты в системе:

```
>sudo apt-get update
```

Apt-get update загружает списки пакетов из репозитория и "обновляет" их, чтобы получить информацию о новейших версиях пакетов и их зависимостях.

После запуска программы необходимо ввести номер телефона и IP-адрес для расчета суммы счета на оплату. Для корректной работы программы необходимо наличие файлов с данными из предыдущих лабораторных работ 1 и 2 в директории исполнения. После выполнения программы в директории сформируется счет на оплату в 2х вариантах – docx и pdf:

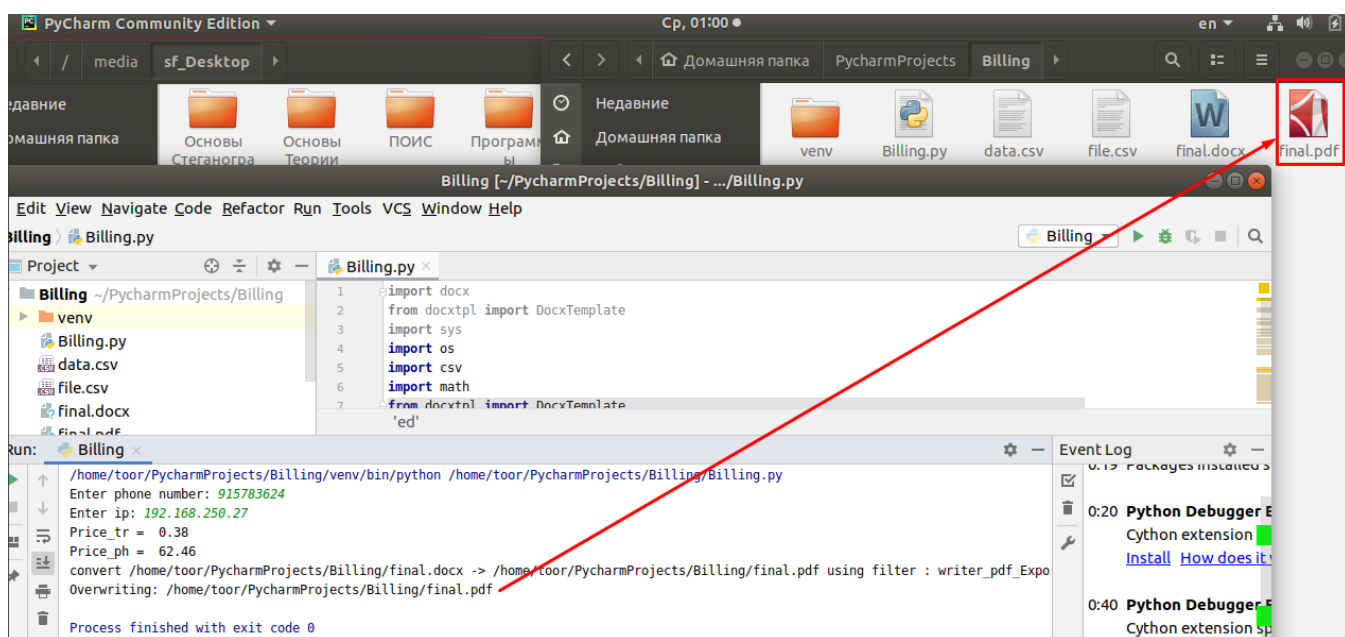


Рис.2 – Результат работы программы

Содержимое счета на оплату - FINAL.PDF:

		БИК	12345
		Сч. №	12345432123563511
Банк получателя ПАО Сбербанк (ИНН 7707083893, ОГРН 1027700132195)			
ИНН 1234567890	КПП 0987654321	Сч. №	02345432123563511
Получатель ООО Моя Фирма - Supplier			

Счет № 5 от 31 марта 2020 г.

Поставщик: ООО Моя Фирма - Supplier
(Исполнитель):
Покупатель: ООО Фирма-Buyer
(Заказчик):
Основание: Коммерческое предложение № 5

№	Наименование работ, услуг	Кол-во	Ед	Цена	Сумма
1	Услуги Сотовой связи	1	шт	62.46	62.46
2	Оплата Интернет	1	шт	0.38	0.38

Итого: 62.84
В том числе НДС: 10.47
Всего к оплате: 62.84

Всего наименований 2
Сумма прописью шестьдесят два руб. восемьдесят четыре коп.

Внимание!
Оплата данного счета означает согласие с условиями поставки товара.
Уведомление об оплате обязательно, в противном случае не гарантируется наличие товара на складе.
Товар отпускается по факту прихода денег на р/с Поставщика, самовывозом, при наличии доверенности и паспорта.

Руководитель Анастасия Алексеевна Бухгалтер Главный бухгалтер Анастасия Алексеевна

Рис.3 – Заполненные поля FINAL.PDF

Запуск программы на исполнение производится при помощи команды из директории Billing с файлом Billing.py (файлы с данными data.csv и file.csv необходимы для корректной работы программы, файл template.docx необходим для формирования счета на оплату):

python3 Billing.py

(либо python Billing.py)

После запуска программы необходимо ввести номер телефона и ip-адрес.

IP-адрес 192.168.250.27

Номер телефона 915783624

В результате работы программы будет сформирован и заполнен данными файл pdf и docx.

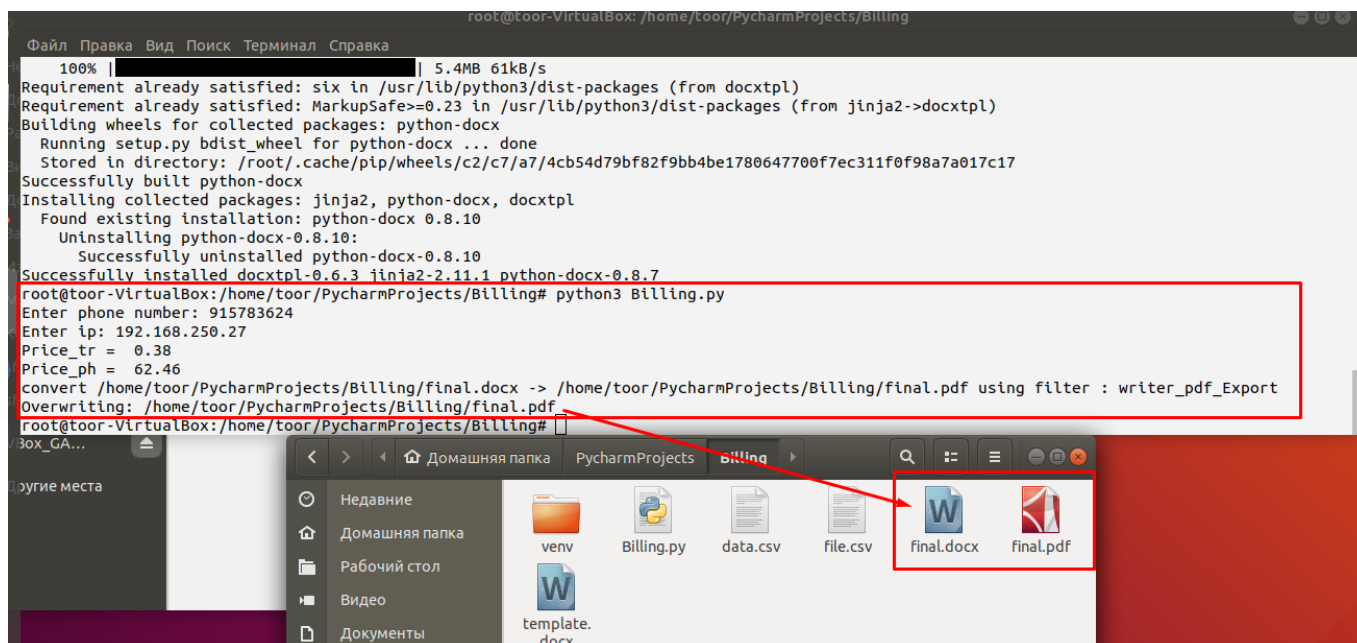


Рис. 4 – Запуск программы на исполнение

Выводы

В результате проделанной работы были консолидированы реализованные ранее программы для тарификации услуг типа «Телефония» и «Интернет» в общий модуль для формирования счета на оплату в формате pdf.

Приложение Billing.py:

```
import docx
from docxtpl import DocxTemplate
import sys
import os
import csv
import math
from docxtpl import DocxTemplate

data_traff = {'out': [], 'in': []} # Array for storing incoming, outgoing traffic

data = {'outCalls': [], 'inCalls': [], 'sms': []} # Array for storing incoming,
outgoing calls and the number of SMS

number = ["ноль", "один", "два", "три", "четыре", "пять", "шесть", "семь", "восемь", "девять"]
teen =
["десять", "одиннадцать", "двенадцать", "тринадцать", "четырнадцать", "пятнадцать", "шестна
дцать", "семнадцать", "восемнадцать", "девятнадцать"]
decades
=["двадцать", "тридцать", "сорок", "пятьдесят", "шестьдесят", "семьдесят", "восемьдесят", "д
евяносто"]

def csv_dict_reader(file_obj, ph_number): #Function for parsing a csv - file and
filling the array with data
    reader = csv.DictReader(file_obj, delimiter=',')

    for line in reader:
        if line['msisdn_origin'] == ph_number:
            data['outCalls'].append(line['call_duration'])
```

```

        data['sms'].append(line['sms_number'])
    if line['msisdn_dest'] == ph_number:
        data['inCalls'].append(line['call_duration'])

def billing(data): # Payment calculation
    price = 0
    call_duration_out_num = 0
    for call_duration_out in data['outCalls']:
        call_duration_out_num += float(call_duration_out)
    if call_duration_out_num > 20:
        call_duration_out_num -= 20
        price += round(math.ceil(call_duration_out_num*2*100)/100, 2)

    for call_duration_in in data['inCalls']:
        price += (float(call_duration_in))*0

    for sms_number in data['sms']:
        price += round(float(sms_number)*2, 2)

    return price

def csv_dict_reader_traff(file_obj, ip): #Function for parsing a csv - file and
filling the array with data
    reader = csv.DictReader(file_obj, delimiter=',')

    for line in reader:
        if line['da'] == ip:
            data_traff['in'].append(line['ibyt'])
        if line['sa'] == ip:
            data_traff['out'].append(line['obyte'])

def traffic(data): # Payment calculation
    price = 0
    traf_Mb = 0
    sum_traf = 0

    for traf_out in data['out']:
        sum_traf += float(traf_out)
    for traf_in in data['in']:
        sum_traf += float(traf_in)
    traf_Mb = sum_traf / (2**20) # From bytes to Mb
    price += round(math.ceil(traf_Mb*100)/100, 2)*1 # - 1 rub / Mb
    return price

def num_to_str(Sum):
    if Sum <= 9:
        return number[Sum]
    elif Sum >= 10 and Sum <= 19:
        tens = Sum % 10
        print(teen[tens])
    elif Sum > 19 and Sum <= 99:
        ones = math.floor(Sum / 10)
        twos = ones - 2
        tens = Sum % 10
        if tens == 0:
            return decades[twos]
        elif tens != 0:
            return decades[twos] + " " + number[tens]

with open("file.csv") as f_obj:
    with open("data.csv") as f_obj_phone:
        ph_number = input("Enter phone number: ")
        ip = input("Enter ip: ")
        csv_dict_reader(f_obj_phone, ph_number)
        csv_dict_reader_traff(f_obj, ip)

```

```

Price_tr = traffic(data_traff)
Price_ph = billing(data)
print('Price_tr = ', Price_tr )
print('Price_ph = ', Price_ph)
Sum = Price_tr+Price_ph;

arr = math.modf(Sum)
f_part = num_to_str(int(arr[1])) + " руб. "
sec_part = num_to_str(math.ceil(arr[0]*100)) + " коп."

doc = DocxTemplate("template.docx") #Get the template

context = {
'product' : 'Услуги Сотовой связи',
'qty' : '1',
'price' : Price_ph,
'sum' : Price_ph,
'product1' : 'Оплата Интернет',
'qty1' : '1',
'price1' : Price_tr,
'sum1' : Price_tr,
'fin_sum' : Sum,
'fin_nds' : round(Sum*20/120,2),
'fin_sum_n' : Sum,
'rows' : '2',
'ed' : 'шт',
'string_sum' : f_part + sec_part,
'bank' : 'ПАО Сбербанк (ИНН 7707083893, ОГРН 1027700132195)',
'inn' : '1234567890',
'kpp' : '0987654321',
'supp' : 'ООО Моя Фирма - Supplier',
'buyer' : 'ООО Фирма-Buyer',
'director' : 'Анастасия Алексеевна',
'bik' : '12345',
'account' : '12345432123563511',
'account2' : '02345432123563511',
'doc_num' : '5',
'data' : '31 марта 2020',
'base' : 'Коммерческое предложение № 5',
'accountant' : 'Главный бухгалтер Анастасия Алексеевна'
}
doc.render(context) # Rendering doc - file
doc.save("final.docx") # Save doc - file

myCmd = 'libreoffice --convert-to pdf final.docx'
os.system (myCmd)

```