Министерство образования и науки РФ

Федеральное государственное автономное

образовательное учреждение высшего образования

Университет ИТМО

**факультет безопасности**

**Информационных технологий**

Лабораторная работа №1

по теме «Обработка и тарификация CDR»  
по предмету «Управление мобильными устройствами»

Работу выполнил

Студент группы №3347

очного отделения:

Якимов Ярослав

Проверил

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Федоров И.Р.



Санкт-Петербург, 2020

**Цель работы:** познакомиться с форматом представления данных – CDR и как он применяется на практике.

**Задачи:**

* Парсинг файла csv, содержащий Call Detail Records
* Тарификация выбранных записей по своему варианту

**Средства реализации:** Для реализации был выбран язык программирования Python, как отвечающий всем современным требованиям в разработке. Кроме того, он предоставляет возможность написать быстрый код, как по времени разработки, так и по времени исполнения. Тем более он имеет встроенные модули для работы с файлами csv.

Исходный код (так же по [ссылке](https://github.com/coltmister/MobileDevYakimov)):

import csv

import time

from math import ceil

balance = 0 # Итоговый счет для номера

input\_file = "data.csv"

msisdn\_origin = '933156729'

first\_sms = 10 # Количество "первых" смс, как только доходит до 0 - считается всё по основному тарифу

first\_minutes\_out = 0 # Количество "первых" исходящих минут

first\_minutes\_in = 0 # Количество "первых" входящих минут

in\_cost = 0 # Cтоимость входящих

out\_cost = 2 # Стоимость исходящих

sms\_cost = 2 # Стоимость смс

def check(row, msisdn\_origin, in\_cost, out\_cost, sms\_cost, first\_sms\_cost=0, first\_minutes\_out\_cost=0,

first\_minutes\_in\_cost=0):

global balance

global first\_sms

global first\_minutes\_out

if msisdn\_origin == row['msisdn\_origin']: # Если это исходящие звонки, то тарификация по исходящим

balance += (ceil(

float(row['call\_duration'])) - first\_minutes\_out) \* out\_cost + first\_minutes\_out \* first\_minutes\_out\_cost

# Считаем сколько стоят исходящие минуты, и т.к. "первые" минуты считаются по звонку, то мы их не сбрасываем

if first\_sms != 0: # смс сбрасываются, так что

if first\_sms < int(row['sms\_number']):

balance += (int(row['sms\_number']) - first\_sms) \* sms\_cost + first\_sms \* first\_sms\_cost

first\_sms = 0

else:

first\_sms -= int(row['sms\_number'])

else:

balance += (int(row['sms\_number'])) \* sms\_cost

if msisdn\_origin == row['msisdn\_dest']: # Входящие звонки

balance += (ceil(

float(row['call\_duration'])) - first\_minutes\_in) \* in\_cost + first\_minutes\_in \* first\_minutes\_in\_cost

return balance

start\_time = time.time()

data = csv.DictReader(open(input\_file))

result = [check(row, msisdn\_origin, in\_cost, out\_cost, sms\_cost) for row in data]

with open('result.txt', 'w') as f:

f.write(f'Total billing: {balance}\n')

f.write(f"Took {time.time() - start\_time} seconds")

**Выводы:** Я познакомился с форматом CDR и научился тарифицировать абонентов, на основании файлов с записями о звонках и смс