**Информационные системы управления корпоративным контентом**

**Практическая работа №9**

**ФИО:** Гусев Тимофей Андреевич

**Группа:** ИКБО-08-21

**Дата предоставления в СДО:**

**Задание 1. Дополнить скрипт для генерации документа с чтением произвольного количества интервалов отпуска из текстового документа. Для каждого интервала в таблицу необходимо вставлять новую строку.**

На рисунке 1 представлено содержимое исходного файла с данными data.txt.

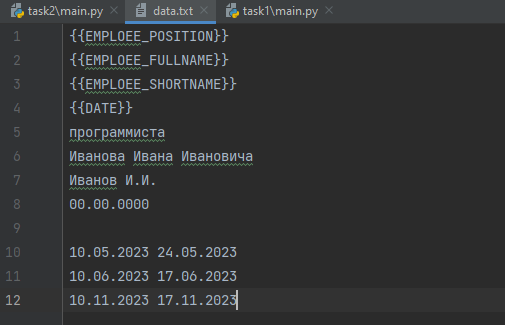


Рисунок 1 – Содержимое файла data.txt

На листинге 1 представлен код функции по перезаписи данных в документ.

Листинг 1 – Код функции replace\_text

if key in paragraph.text:  
 paragraph.text = paragraph.text.replace(key, value)

На листинге 2 представлен код функции по чтению файла с data.txt в словарь.

Листинг 2 – Код функции read\_data

def read\_data():  
 data = {}  
 pattern = r"{{\w{1,}}}"  
 keys = []  
 values = []  
 durations = []  
 with open("data.txt", "r", encoding="UTF-8") as f:  
 lines = f.readlines()  
 split\_index = lines.index("\n")  
 items = lines[:split\_index]  
 dates = lines[split\_index + 1:]  
 for line in items:  
 if re.fullmatch(pattern, line.strip()):  
 keys.append(line.strip())  
 else:  
 values.append(line.strip())  
 for date in dates:  
 date1, date2 = date.split(" ")  
 date1 = date1.strip()  
 date2 = date2.strip()  
 dif = str((datetime.strptime(date2, "%d.%m.%Y") - datetime.strptime(date1, "%d.%m.%Y")).days)  
 durations.append([date1, date2, dif])  
 for i in range(len(keys)):  
 if keys[i] == "{{DATE}}":  
 data[keys[i]] = datetime.today().date().strftime("%d.%m.%y")  
 else:  
 data[keys[i]] = values[i]  
 data["DURATIONS"] = durations  
 return data

На листинге 3 представлен код основной функции, выполняющей подстановку данных в шаблон.

Листинг 3 – Код функции main

def main():  
 data = read\_data()  
 template = 'template.docx'  
 result = 'result.docx'  
 template\_doc = Document(template)  
 for key, value in data.items():  
 for paragraph in template\_doc.paragraphs:  
 replace\_text(paragraph, key, value)  
 my\_table = template\_doc.tables[0]  
 n = 1  
 for line in data["DURATIONS"]:  
 row = my\_table.add\_row()  
 i = -1  
 for cell in row.cells:  
 if i == -1:  
 cell.text = str(n)  
 else:  
 cell.text = line[i]  
 i += 1  
 n += 1  
 template\_doc.save(result)

На рисунке 2 представлен файл result.docx, сгенерировавшийся в результате выполнения программы.

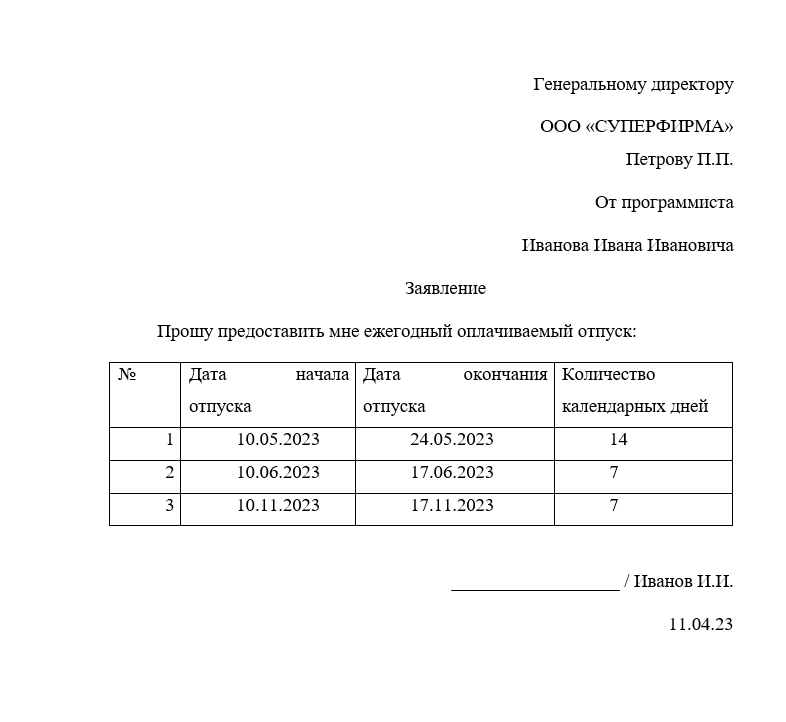


Рисунок 2 – Содержимое файла result.docx

**Задание 2. Реализовать скрипт для генерации накладных из таблицы с данными:**

**➔ в таблице на каждую компанию содержится несколько строк с товарами (фильтровать или вносить ручные изменения в xslx-файл не разрешено);**

**➔ номера накладных ежедневно начинается с 1 и увеличиваются на 1 с каждой последующей в этот день;**

**➔ для каждой компании создается один docx-файл (одна накладная) с названием накладная\_01\_НазваниеКомпании.docx, в которую включаются все товарные позиции;**

**➔ в каждой накладной должны подсчитываться итоговое количество товарных позиций и итоговая стоимость.**

На рисунке 3 представлено содержимое исходного файла с даннымиdata.xlsx.

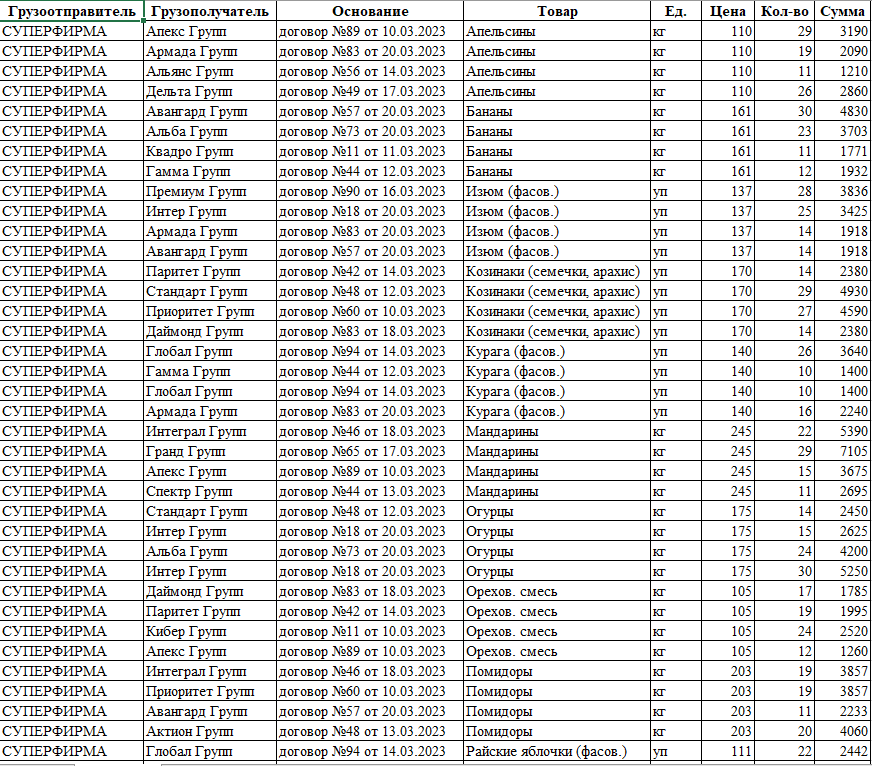


Рисунок 3 – Содержимое файла data.xlsx

На листинге 4 представлен код функции по чтению данных из файла data.xlsx в словарь.

Листинг 4 – Код функции read\_data

def read\_data():  
 "Читает данные из файла excel в dict"  
 data = {}  
 df = pd.read\_excel("data.xlsx")  
 df = df.sort\_values("Грузополучатель")  
 for i, row in df.iterrows():  
 data[row["Грузополучатель"]] = {}  
 data[row["Грузополучатель"]]["Позиции"] = []  
 data[row["Грузополучатель"]]["{{NUMBER}}"] = len(list(data.keys()))  
 for i, row in df.iterrows():  
 data[row["Грузополучатель"]]["{{TO}}"] = row["Грузополучатель"]  
 data[row["Грузополучатель"]]["{{FROM}}"] = row["Грузоотправитель"]  
 data[row["Грузополучатель"]]["{{DOCUMENT}}"] = row["Основание"]  
 data[row["Грузополучатель"]]["{{DATE}}"] = datetime.today().date().strftime("%d.%m.%y")  
 data[row["Грузополучатель"]]["Позиции"].append([len(data[row["Грузополучатель"]]["Позиции"]) + 1, row["Товар"], row["Ед."], row["Цена"], row["Кол-во"], row["Сумма"]])  
 data[row["Грузополучатель"]]["{{GLOBAL\_PRICE}}"] = sum(el[5] for el in data[row["Грузополучатель"]]["Позиции"])  
 data[row["Грузополучатель"]]["{{GLOBAL\_COUNT}}"] = len(data[row["Грузополучатель"]]["Позиции"])  
 data[row["Грузополучатель"]]["{{DOCUMENT}}"] = row["Основание"]  
 return data

На листинге 5 представлен код функции по перезаписи данных в документ.

Листинг 5 – Код функции replace\_text

if key in paragraph.text:  
 paragraph.text = paragraph.text.replace(key, value)

На листинге 6 представлен код основной функции, выполняющей подстановку данных в шаблон.

Листинг 6 – Код функции main

def main():  
 "Основная функция"  
 data = read\_data()  
 pattern = r"{{\w{1,}}}"  
 template = 'template.docx'  
 n = 1  
 for key, value in data.items():  
 result = f'накладная\_{n:02}\_{key}.docx'  
 template\_doc = Document(template)  
 for k, v in value.items():  
 if re.fullmatch(pattern, k.strip()):  
 for paragraph in template\_doc.paragraphs:  
 replace\_text(paragraph, k, str(v))  
 my\_table = template\_doc.tables[0]  
 for line in value["Позиции"]:  
 row = my\_table.add\_row()  
 i = 0  
 for cell in row.cells:  
 cell.text = str(line[i])  
 i += 1  
 template\_doc.save(f"result/{result}")  
 n += 1

На рисунке 4 представлен список сгенерированных в ходе выполнения программы файлов.

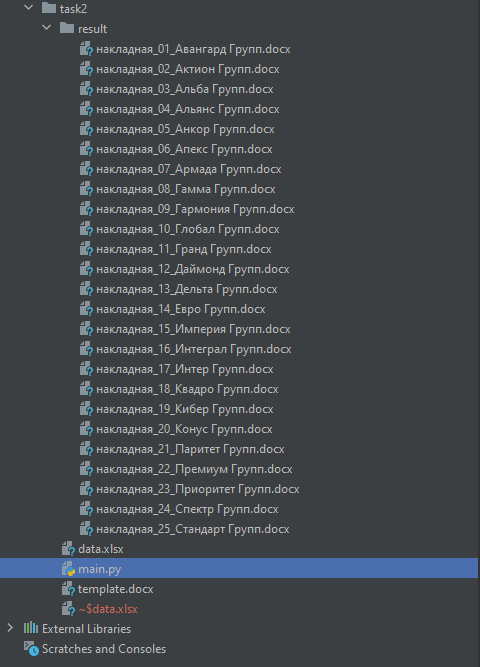


Рисунок 4 – Сгенерированные файлы

На рисунке 5 представлено содержимое однного из сгенерированных файлов.

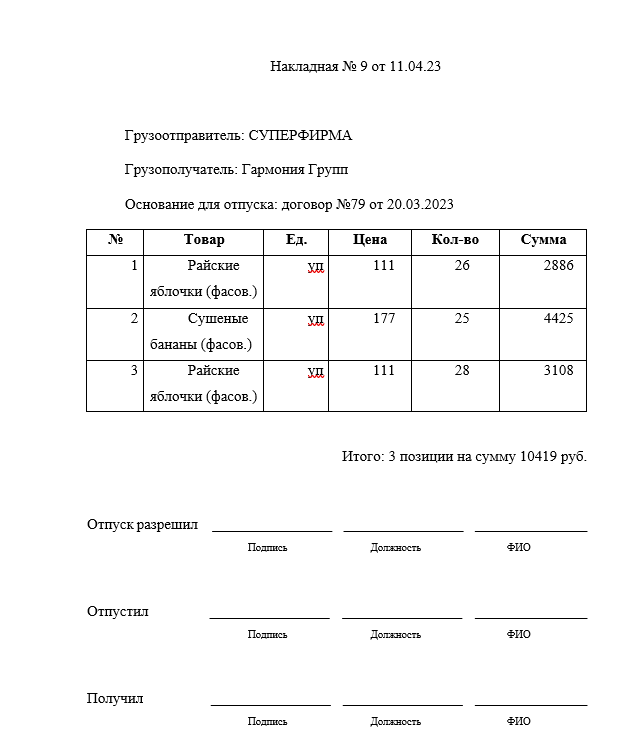


Рисунок 5 – Содержимое файла накладная\_09\_Гармония Групп.docx