Решение задач

Фомина К.С.

***Задача 1:***

*Дано: 10 мужчин, которые вместе в плане финансов организовывали 8 марта.*

*Виталий оплатил праздничный стол, потратив 30000р.*

*Василий оплатил скрипача, потратив 3000р*

*Больше затрат не было.*

*Виталий написал всем - я потратил 30000, скидывайте мне по 3000р.*

*Все кроме Василия скинулись. Василий написал - я не буду, ведь как раз 3000р потратил.*

*Виталий же высказался резко против этого.*

*Вопрос, почему Виталий был против?*

***Ответ:***

Общие затраты на организацию праздника составили 33000р. Если считать, что каждый вкладывает равную сумму, то эта сумма будет равна 3300р.

В описанной же ситуации Виталий потратил 30000р., из которых ему вернули 24000р. Получается 9 человек, включая Василия, потратили 3000р., а Виталий потратил 6000р. Естественно, его этот вариант не устраивает.

# Задача 2:

Выгрузку из файла "Задание 2\_1" дополнить полями "Цех" и "ID параметра".

Цех можно достать из поля "Параметр" как первые *n* символов до точки.

ID параметра можно достать из поля "Единица продукции" как первые *n* цифр до нижнего подчеркивания.

Для выгрузки сформировать дашборд Power BI со следующими данными (можно на разных вкладках):

Графики:

* Количество уникальных параметров ежедневно по дате производства с разбивкой по цехам
* Количество уникальных параметров ежедневно по дате инцидента с разбивкой по цехам
* Количество единиц продукции ежедневно по дате производства с разбивкой по цехам
* Количество единиц продукции ежедневно по дате инцидента с разбивкой по цехам
* Количество единиц продукции ежедневно по дате производства с разбивкой по параметрам
* Количество единиц продукции ежедневно по дате инцидента с разбивкой по параметрам

Прочие визуализации:

* День инцидента с максимальным количеством уникальных параметров: дата и число
* День инцидента с минимальным количеством уникальных параметров: дата и число
* День инцидента с максимальным количеством единиц продукции: дата и число
* День инцидента с минимальным количеством единиц продукции: дата и число
* Среднесуточное количество уникальных параметров за период: только число
* Среднесуточное количество единиц продукции за период: только число

Требования: все графики и показатели должны динамически обновляться при изменении выгрузки (при добавлении или убавлении строк). Должны быть фильтры по дате производства и по дате инцидента

# Задача 3:

Файл "Задание 3" содержит данные о продуктах (период производства и показатель ВГ).

Необходимо любыми средствами добавить автоматически рассчитываемое поле "Критичная позиция" (1 – крит., 0 – не крит).

Условия критичной позиции:

* Если продукт производился в обоих периодах (период 1 – "2023.01.01 - 2023.03.01", период 2 – "2023.04.01 - 2023.06.01"), разница показателя ВГ периода 1 – период 2 строго больше 5 и значение показателя ВГ во 2 периоде строго меньше 90
* Если продукт производился только в одном из двух периодов (любом), и значение показателя ВГ в этом периоде строго меньше 90

Для решения задачи использовался внутренний язык программирования Excel, написан макрос с названием CalculateCriticalPosition.

Решение задание находится в файле Задание 3.xlsm

Код скрипта находится в Приложении 1

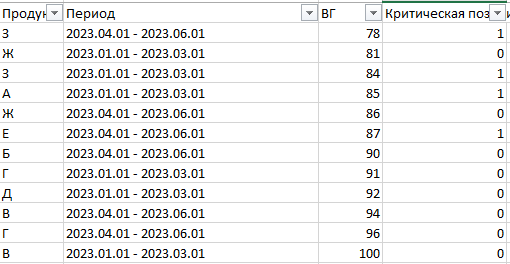
******

Рис.1. Поле «критическая позиция», заполненное с помощью скрипта

# Задача 4:

*Пользователь должен иметь возможность работать в написанном Вами интерфейсе.*

*Исходными данными является массив в файле Задание 4.*

*Действия:*

1. *Пользователь выбирает любую партию из предложенных, данная партия является образцом для последующих действий.*
2. *Ваша задача на выходе передать пользователю список или списки возможных комбинаций данного образца с другими партиями (в каждом списке должна быть партия-образец из пункта 1).*
3. *В каждом списке все партии должны соответствовать условиям:*
   1. *Значение параметра 1 должно совпадать*
   2. *Значение параметра 2 должно отличаться по абсолютной величине не более 10.*
   3. *Значение параметра 3 должно отличаться по абсолютной величине не более 0,8.*
   4. *Значение параметра 4 должно отличаться по абсолютной величине не более 1,5.*
   5. *Значение параметра 5 должно отличаться по абсолютной величине не более 1.*
   6. *Значение параметра 6 должно отличаться по абсолютной величине не более 9.*
   7. *Значение параметра 7 должно отличаться по абсолютной величине не более 1.*
4. *Обратите внимание:*
   1. *Если в списке к образцу подобрано 3 партии, то они все между собой должны соответствовать условиям из пункта 3.*
   2. *Повторяющиеся списки необходимо удалять (например, есть 2 списка 56 02+56 03+56 03 и 56 02+56 03; 56 02+56 03- лишний, его удаляем).*

Для решения данной задачи были использованы язык программирования C# и инструменты Windows Forms. Исходные файлы приложения находятся в папке «Ex4», для работы с приложением нужно запустить файл «Ex4.exe»

Используя эту программу, пользователь может найти все комбинации партий, удовлетворяющих условию, при чем выводятся только комбинаций, без подкомбинаций. Т.к. если нужны будут какие-то подкомбинации, их можно выделить из найденных комбинаций.

Код находится в Приложении 2

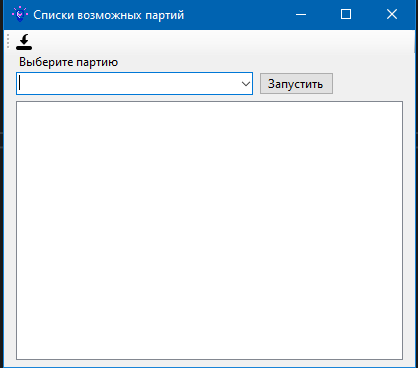


Рис.2. Интерфейс пользователя

Перед началом работы обязательно нужно загрузить файл с помощью кнопки . Затем нужно выбрать партию-образец, используя выпадающий список.

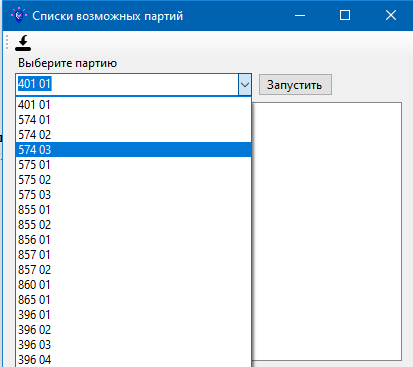


Рис.3. Список партий

После выбора партии останется только нажать на кнопку «запустить»

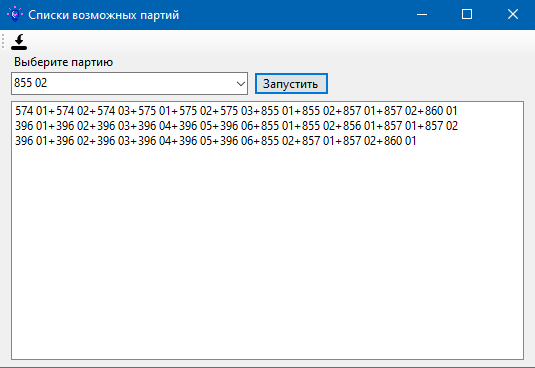


Рис.4. Список возможных комбинаций с партией 855 02

Приложение 1

Код для задания 3:

Sub CalculateCriticalPosition()

Dim ws As Worksheet

Set ws = ThisWorkbook.Sheets("Лист1")

Dim products As Object

Set products = CreateObject("Scripting.Dictionary")

Dim lastRow As Long

lastRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row ' Поиск последней строки с данными

Dim i As Long

Dim product As String

Dim period As String

Dim vg As Double

Dim key As String

' Проходим по всем товарам и записываем их ВГ в словарь

For i = 2 To lastRow

product = ws.Cells(i, "A").Value

period = ws.Cells(i, "B").Value

vg = ws.Cells(i, "C").Value

' Формируем ключ из названия продукта и периода

key = product & "|" & period

' Добавляем информацию о ВГ в словарь

If Not products.Exists(key) Then

products.Add key, vg

End If

Next i

' Обнуляем столбец "Критическая позиция"

ws.Range("D2:D" & lastRow).ClearContents

' Проверяем условия для каждого продукта

For i = 2 To lastRow

product = ws.Cells(i, "A").Value

period = ws.Cells(i, "B").Value

' Ключи для первого и второго периода

Dim keyFirstPeriod As String

Dim keySecondPeriod As String

keyFirstPeriod = product & "|2023.01.01 - 2023.03.01"

keySecondPeriod = product & "|2023.04.01 - 2023.06.01"

' Проверяем, существуют ли оба ключа

If products.Exists(keyFirstPeriod) And products.Exists(keySecondPeriod) Then

' Проверяем разницу ВГ и значение ВГ во втором периоде

If products(keyFirstPeriod) - products(keySecondPeriod) > 5 And products(keySecondPeriod) < 90 Then

ws.Cells(i, "D").Value = 1

Else

ws.Cells(i, "D").Value = 0

End If

ElseIf products.Exists(keyFirstPeriod) Or products.Exists(keySecondPeriod) Then

' Проверяем значение ВГ в периоде, если продукт производился только в одном

If (products.Exists(keyFirstPeriod) And products(keyFirstPeriod) < 90) Or \_

(products.Exists(keySecondPeriod) And products(keySecondPeriod) < 90) Then

ws.Cells(i, "D").Value = 1

Else

ws.Cells(i, "D").Value = 0

End If

End If

Next i

' Очищаем память

Set products = Nothing

End Sub

Приложение 2

Код для задания 4

**Файл Party.cs**

namespace Ex4

{

internal class Party

{

public string Number { get; set; }

public double Parameter1 { get; set; }

public double Parameter2 { get; set; }

public double Parameter3 { get; set; }

public double Parameter4 { get; set; }

public double Parameter5 { get; set; }

public double Parameter6 { get; set; }

public double Parameter7 { get; set; }

public bool MatchesCriteria(Party other)

{

return this.Parameter1 == other.Parameter1 &&

Math.Abs(this.Parameter2 - other.Parameter2) <= 10 &&

Math.Abs(this.Parameter3 - other.Parameter3) <= 0.8 &&

Math.Abs(this.Parameter4 - other.Parameter4) <= 1.5 &&

Math.Abs(this.Parameter5 - other.Parameter5) <= 1 &&

Math.Abs(this.Parameter6 - other.Parameter6) <= 9 &&

Math.Abs(this.Parameter7 - other.Parameter7) <= 1;

}

}

}

**Файл Combinations.cs:**

using OfficeOpenXml;

using System.IO;

using System.Collections.Generic;

namespace Ex4

{

public partial class CombinationsParty : Form

{

List<Party> parties = new List<Party>();

List<List<Party>> combinations = new List<List<Party>>();

//загрузка данных из файла в parties

public void UploatFile(string filePath)

{

parties.Clear();

ExcelPackage.LicenseContext = LicenseContext.NonCommercial;

FileInfo fileInfo = new FileInfo(filePath);

using (ExcelPackage package = new ExcelPackage(fileInfo))

{

ExcelWorksheet worksheet = package.Workbook.Worksheets[0];

int totalRows = worksheet.Dimension.End.Row;

int totalCols = worksheet.Dimension.Columns;

for (int i = 2; i <= totalRows; i++)

{

if (worksheet.Cells[i, 1].Value != null)

{

Party record = new Party

{

Number = worksheet.Cells[i, 1].Value.ToString(),

Parameter1 = Convert.ToDouble(worksheet.Cells[i, 2].Value),

Parameter2 = Convert.ToDouble(worksheet.Cells[i, 3].Value),

Parameter3 = Convert.ToDouble(worksheet.Cells[i, 4].Value),

Parameter4 = Convert.ToDouble(worksheet.Cells[i, 5].Value),

Parameter5 = Convert.ToDouble(worksheet.Cells[i, 6].Value),

Parameter6 = Convert.ToDouble(worksheet.Cells[i, 7].Value),

Parameter7 = Convert.ToDouble(worksheet.Cells[i, 8].Value),

};

parties.Add(record);

}

}

}

}

public CombinationsParty()

{

InitializeComponent();

}

//обработка нажатия на кнопку загрузки

private void UploadButton\_Click(object sender, EventArgs e)

{

using (OpenFileDialog openFileDialog = new OpenFileDialog())

{

openFileDialog.InitialDirectory = "D:\\"; // Вы можете указать другую начальную директорию

openFileDialog.Filter = "Excel Files|\*.xls;\*.xlsx;\*.xlsm"; // Фильтр для файлов Excel

openFileDialog.FilterIndex = 2;

openFileDialog.RestoreDirectory = true;

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

string filePath = openFileDialog.FileName;

UploatFile(filePath);

}

}

comboBoxParties.DataSource = parties;

comboBoxParties.DisplayMember = "Number"; // Текст, который будет отображаться в ComboBox

comboBoxParties.ValueMember = "Number"; // Значение, которое будет использоваться в ComboBox

}

private void FindCombinations(Party sampleParty, List<Party> AllParty)

{

// Проходим по всем партиям и ищем комбинации

foreach (var party in AllParty)

{

// Пропускаем партию-образец

if (party.Number == sampleParty.Number)

continue;

// Проверяем, соответствует ли текущая партия условиям

if (sampleParty.MatchesCriteria(party))

{

// Добавляем в комбинации, где образец уже есть

foreach (var combo in combinations)

{

// Проверяем, что все партии в комбинации соответствуют друг другу

if (combo.All(item => item.MatchesCriteria(party)) && !combo.Contains(party))

{

combo.Add(party);

}

}

// Создаем новую комбинацию с текущей партией и образцом

combinations.Add(new List<Party> { sampleParty, party });

}

}

// Удаляем повторяющиеся и подкомбинации

combinations = RemoveDuplicateСombinations(combinations);

}

private List<List<Party>> RemoveDuplicateСombinations(List<List<Party>> combinations)

{

// Удаляем дубликаты внутри комбинаций и сортируем их

for (int i = 0; i < combinations.Count; i++)

{

combinations[i] = combinations[i].Distinct().ToList();

combinations[i].Sort((a, b) => a.Number.CompareTo(b.Number));

}

// Список для хранения индексов подкомбинаций, которые нужно удалить

var subcombinationIndexes = new HashSet<int>();

// Сравниваем каждую комбинацию с остальными

for (int i = 0; i < combinations.Count; i++)

{

for (int j = 0; j < combinations.Count; j++)

{

// Не сравниваем комбинацию с самой собой

if (i == j) continue;

// Если комбинация i является подмножеством комбинации j, добавляем её индекс для удаления

if (!subcombinationIndexes.Contains(i) &&

new HashSet<Party>(combinations[j]).IsSupersetOf(combinations[i]))

{

subcombinationIndexes.Add(i);

}

}

}

// удаляем подкомбинации с конца

var combinationsWithoutSubcombinations = combinations

.Where((\_, index) => !subcombinationIndexes.Contains(index))

.ToList();

return combinationsWithoutSubcombinations;

}

private void StartButton\_Click(object sender, EventArgs e)

{

string numberParty=comboBoxParties.Text;

Party foundParty = parties.FirstOrDefault(p => p.Number == numberParty);

if (foundParty != null)

{

// Вызовите функцию с найденным объектом партии

FindCombinations(foundParty, parties);

}

else

{

// Обработка случая, когда партия с таким номером не найдена

Console.WriteLine("Партия с номером " + numberParty + " не найдена.");

}

listBoxParties.Items.Clear();

// Добавьте элементы в ListBox

foreach (var partyGroup in combinations)

{

// Создайте строку из номеров партий в группе, разделенных знаком "+"

string displayText = string.Join("+", partyGroup.Select(p => p.Number));

// Добавьте строку в ListBox

listBoxParties.Items.Add(displayText);

}

}

}

}