



Программирование в среде R

Шевцов Василий Викторович,
директор ДИТ РУДН, shevtsov_vv@rudn.university

Язык М

Редактор кода

Настраиваемый столбец

Добавьте столбец, который вычисляется на основании других столбцов.

Имя нового столбца

Пользовательский

Настраиваемая формула столбца ⓘ

=

Доступные столбцы

Страна
Округ
Область
Год
Численность

<< Вставить

Сведения о формулах Power BI Desktop

✓ Синтаксические ошибки не обнаружены.

OK

Отмена

Без имени — Редактор Power Query

Файл

Главная страница

Преобразование

Добавление столбца

Просмотр

Инструменты

Справка

Параметры запроса

Структура

☐ Строка формул

☐ Моноширинный ☐ Распределение столбцов

☒ Показать пробелы ☐ Профиль столбца

☐ Качество столбца

Предварительный просмотр данных

Перейти к столбцу.

Столбцы

☐ Всегда разрешать


Параметры

Расширенный редактор


Подробнее

Зависимости запроса

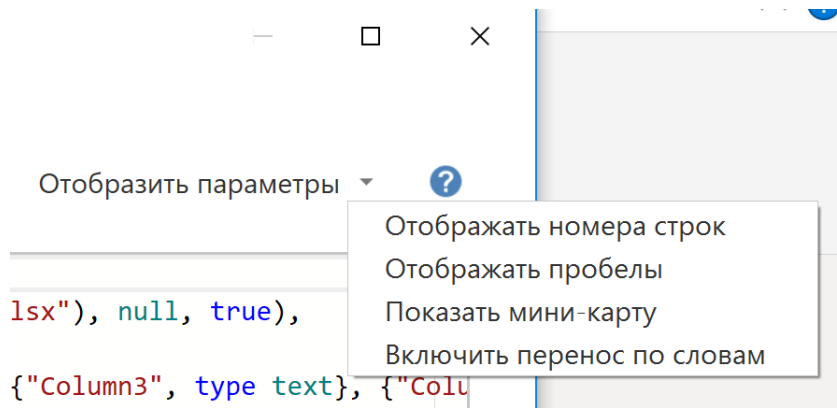
Зависимости

 RUDN university

3

 5100

Параметры



1

```
1 let  
2     Источник = Excel.W  
3     Лист1_Sheet = Исто  
4     #"Измененный тип"  
5     #"Повышенные загол  
6     #"Измененный тип1"  
7     #"Несвернутые стол  
8     #"Переименованные  
9     #"Измененный тип2"  
10 in  
11     #"Измененный тип2"
```

2

```
1 let  
2     ... Источник = Excel.W  
3     ... Лист1_Sheet = Исто  
4     ... #"Измененный тип"  
5     ... #"Повышенные загол  
6     ... #"Измененный тип1"  
7     ... #"Несвернутые стол  
8     ... #"Переименованные  
9     ... #"Измененный тип2"  
10 in  
11     ... #"Измененный тип2"
```

3

Отобразить параметры ?


```
18.xlsx"), n  
t}, {"Column  
, type text  
Атрибут", "З  
, "Численно
```

4

```
), null, true),  
lumn3", type text}  
mn8", Int64.Type},  
lumn13",  
  
ext}, {"Область",  
Int64.Type},  
}),  
, "Значение"),  
нность"})),
```

Комментарии

- однострочные комментарии (//)
- многострочные комментарии, начинающиеся с /* и заканчивающиеся на */.

 Расширенный редактор

Численность населения

Отобразить параметры ▾ ?

```
// однострочный комментарий
/*
    многострочный
    комментарий
*/
let
    Источник = Excel.Workbook(File.Contents("D:\ФА\2019-2020 уч.г\PowerBI\Семинар 4\Численность населения 2018.xlsx"), null),
    Лист1_Sheet = Источник{[Item="Лист1",Kind="Sheet"]}[Data],
    #"Измененный тип" = Table.TransformColumnTypes(Лист1_Sheet,{{"Column1", type text}, {"Column2", type text}, {"Column3",
    #"Повышенные заголовки" = Table.PromoteHeaders(#"Измененный тип", [PromoteAllScalars=true]),
    #"Измененный тип1" = Table.TransformColumnTypes(#"Повышенные заголовки",{{"Страна", type text}, {"Округ", type text}, {"
    #"Несвернутые столбцы" = Table.UnpivotOtherColumns(#"Измененный тип1", {"Страна", "Округ", "Область"}, "Атрибут", "Знач
    #"Переименованные столбцы" = Table.RenameColumns(#"Несвернутые столбцы",{{"Атрибут", "Год"}, {"Значение", "Численность"
    #"Измененный тип2" = Table.TransformColumnTypes(#"Переименованные столбцы",{{"Год", Int64.Type}})
in
    #"Измененный тип2"
```

✓ Синтаксические ошибки не обнаружены.

Готово

Отмена

Создание запросов

- М — это язык с учетом регистра.
- Комбинированный запрос состоит из переменных, выражений и значений, инкапсулированных в выражении let.
- Переменная может содержать пробелы. Для этого нужно указать идентификатор # и имя в кавычках, например #"Имя переменной".
 - `Variable name = expression`
 - `#"Variable name" = expression`
- Шаг формулы может быть пользовательской формулой.
- Каждый шаг формулы запроса основан на предыдущем, на который он ссылается по имени переменной.
- Вывод шага формулы запроса осуществляется с помощью инструкции in. Как правило, в качестве результирующего набора данных используется последний шаг запроса.

Создание источника из запроса

Главная страница Преобразование

Создать источник Последние источники Введите данные

Самые распространенные

- Excel
- SQL Server
- службы Analysis Services
- Текстовый или CSV-файл
- Интернет
- Канал OData
- Пустой запрос
- Дополнительно...

Запросы [3]

- Численность населения
- Запрос1
- А^Bс Запрос2

Параметры запроса

СВОЙСТВА

Имя

Запрос2

Все свойства

ПРИМЕНЕННЫЕ ШАГИ

Источник

Расширенный редактор

Запрос2

Отобразить параметры ?

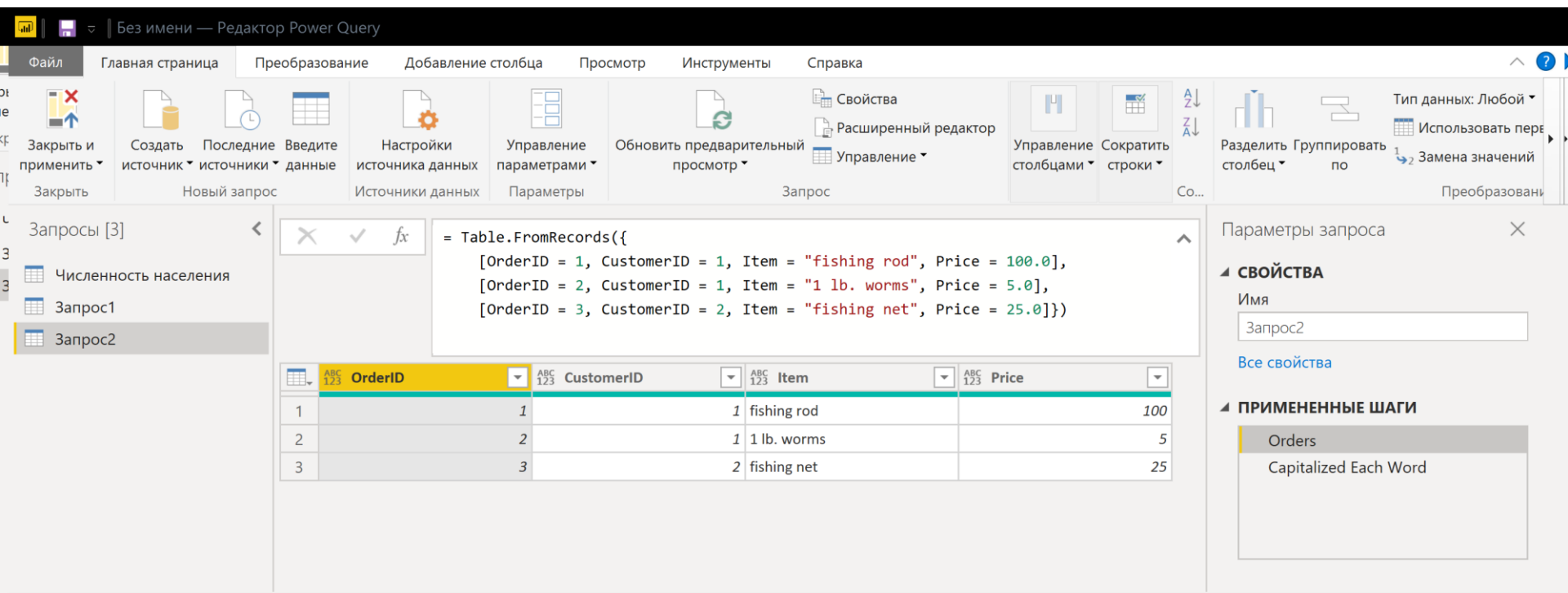
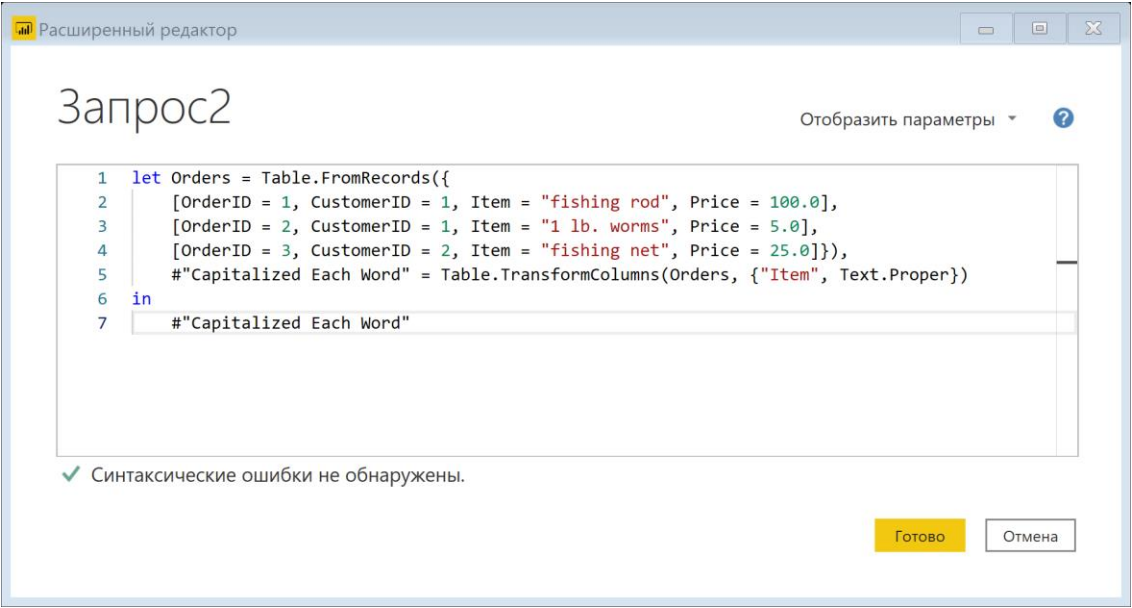
```
1 let
2   Источник = ""
3 in
4   Источник
```

✓ Синтаксические ошибки не обнаружены.

Готово Отмена

Запрос

Команды, записанные в редакторе, отражаются в примененных шагах отдельными записями



Код для вставки

```
let Orders = Table.FromRecords({
  [OrderID = 1, CustomerID = 1, Item = "fishing rod", Price = 100.0],
  [OrderID = 2, CustomerID = 1, Item = "1 lb. worms", Price = 5.0],
  [OrderID = 3, CustomerID = 2, Item = "fishing net", Price = 25.0]}),
#"Capitalized Each Word" = Table.TransformColumns(Orders, {"Item", Text.Proper})
in
#"Capitalized Each Word"
```

Шаги формулы

- Orders — создает таблицу [Table](#_Table_value) с данными из таблицы Orders.
- # "Capitalized Each Word" — для изменения первой буквы в каждом слове на прописную используется Table.TransformColumns().
- in #"Capitalized Each Word" — выводит таблицу, в которой каждое слово начинается с прописной буквы.

Базовые понятия

Значения и выражения

Значения — это значения, такие как число 1 или текстовая строка, или более сложные объекты, такие как таблицы. Также значения могут быть записаны как выражения, но при этом выражениями они не являются, например, $1+1$ возвращает значение 2.

Выражение — это формула, используемая для построения значений. Выражение может быть сформировано с использованием синтаксических конструкций.

Пример:

if $2 > 1$ then 2 else 1

Типы значений

1. Простые значения (primitive value) – числовые, логические, текст или null. Например:

123 — число,

true — логическое значение,

null – отсутствие данных, null

2. Список (list) – упорядоченная последовательность значений.

Фигурные скобки { и }

обозначают начало и конец списка. Пример:

{1, 2, 3} – список из чисел.

3. Запись (record) – это упорядоченная последовательность полей, где каждое поле

имеет имя и одно значение (любого типа). Пример:

[A = 1, B = 2, C = 3]

Типы значений

4. Таблица (table) – набор значений в виде строк и именованных столбцов. Нет определенного синтаксиса для создания таблицы, ее можно создать или обратиться к ней с помощью нескольких стандартных функций.

Например:

```
#table( {"A", "B"}, { {1, 2}, {3, 4} } )
```

5. Функция (function) преобразует набор входящих значений в одно результирующее.

Функция записывается путем перечисления параметров функции в круглых скобках (), за которыми следует знак перехода => далее указывают выражение, определяющее функцию.

```
(x, y) => (x + y) / 2
```

Операторы

Список операторов M

Язык формул Power Query M включает набор операторов, которые можно использовать в выражении. Операторы применяются к операндам для формирования символьных выражений. Например, в выражении $1 + 2$ числа 1 и 2 являются операндами, а оператор представлен оператором сложения (+).

Значение оператора может меняться в зависимости от типа значений операндов.

Оператор сложения (+)

Выражение	Равно
$1 + 2$	Суммирование числа: 3
<code>#time (12, 23, 0) + #duration (0, 0, 2, 0)</code>	Арифметические операции со временем: <code>#time(12,25,0)</code>

Оператор комбинирования (&)

Выражение	Равно
<code>"A" & "BC"</code>	Объединение текста: "ABC"
<code>{1} & {2, 3}</code>	Объединение списков: {1, 2, 3}
<code>[a = 1] & [b = 2]</code>	Слияние записей: [a = 1, b = 2]

Список операторов М

Общие операторы, которые применяются к значению NULL, логическому значению, числу, времени, дате, значению datetime, значению datetimezone, длительности, тексту, двоичному значению)

Оператор	Описание
>	Больше чем
>=	Больше или равно
<	Меньше чем
<=	Меньше или равно
=	Равно
<>	Не равно

Список операторов М

Логические операторы (в дополнение к общим операторам)

Оператор	Описание
or	Условное логическое ИЛИ
and	Условное логическое И
not	Логическое НЕ

Числовые операторы (в дополнение к общим операторам)

Оператор	Описание
+	Сумма
-	Разность
*	Продукт
/	Частное
+x	Унарный плюс
-x	Отрицание

Список операторов М

Текстовые операторы (в дополнение к общим операторам)

Оператор	Описание
&	Объединение

Операторы для списков, записей, таблиц

Оператор	Описание
=	Равно
<>	Не равно
&	Объединение

Список операторов M

Оператор поиска записей

Оператор	Описание
[]	Обращение к полям записи по имени.

Оператор индексатора списка

Оператор	Описание
{}	Обращение к элементу списка по его числовому индексу, отсчитываемому от нуля.

Операторы утверждения и совместимости типов

Оператор	Описание
is	Выражение "x is y" возвращает true, если тип значения x совместим с y, и false, если они несовместимы.
as	Выражение "x as y" утверждает, что значение x совместимо с y в соответствии с применением оператора is.

Список операторов M

Операторы даты

Оператор	x	y	Значение
$x + y$	время	длительность	Смещение даты по длительности
$x + y$	длительность	время	Смещение даты по длительности
$x - y$	время	длительность	Смещение даты по длительности, к которой применено отрицание
$x - y$	время	время	Длительность между датами
$x \& y$	дата	время	Объединенное значение datetime

Список операторов М

Операторы для значений datetime

Оператор	x	y	Значение
$x + y$	значение datetime	длительность	Смещение значения datetime по длительности
$x + y$	длительность	значение datetime	Смещение значения datetime по длительности
$x - y$	значение datetime	длительность	Смещение значения datetime по длительности, к которой применено отрицание
$x - y$	значение datetime	значение datetime	Длительность между значениями datetime

Список операторов М

Операторы для значений datetimezone

Оператор	x	y	Значение
$x + y$	значение datetimezone	длительность	Смещение значения datetimezone по длительности
$x + y$	длительность	значение datetimezone	Смещение значения datetimezone по длительности
$x - y$	значение datetimezone	длительность	Смещение значения datetimezone по длительности, к которой применено отрицание
$x - y$	значение datetimezone	значение datetimezone	Длительность между значениями datetimezone

Список операторов M

Операторы длительности

Оператор	x	y	Значение
$x + y$	значение datetime	длительность	Смещение значения datetime по длительности
$x + y$	длительность	значение datetime	Смещение значения datetime по длительности
$x + y$	длительность	длительность	Сумма длительностей
$x - y$	значение datetime	длительность	Смещение значения datetime по длительности, к которой применено отрицание
$x - y$	значение datetime	значение datetime	Длительность между значениями datetime
$x - y$	длительность	длительность	Разница длительностей
$x * y$	длительность	число	Длительность, увеличенная в N раз
$x * y$	число	длительность	Длительность, увеличенная в N раз
x / y	длительность	число	Частное длительности

Список операторов M

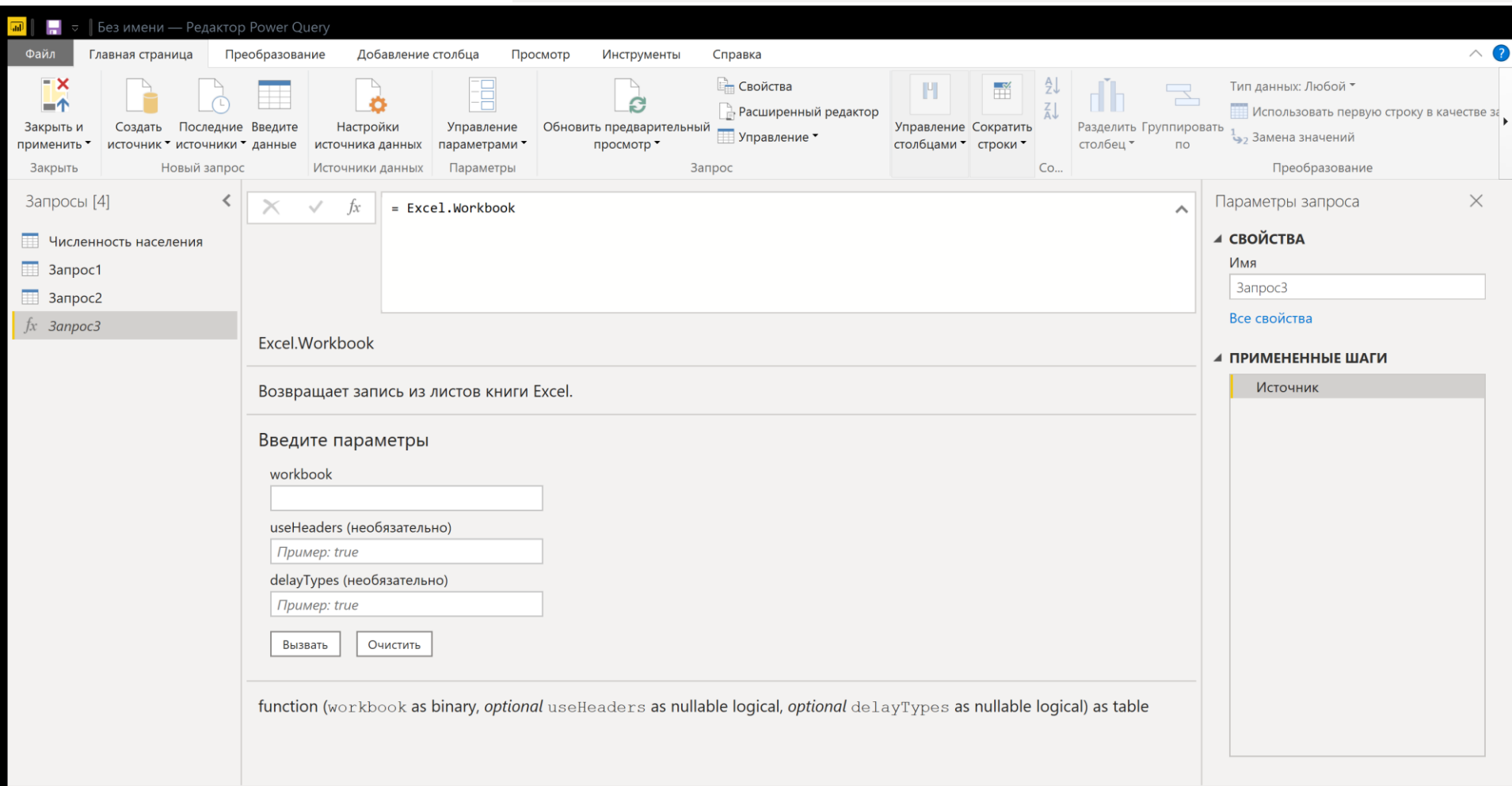
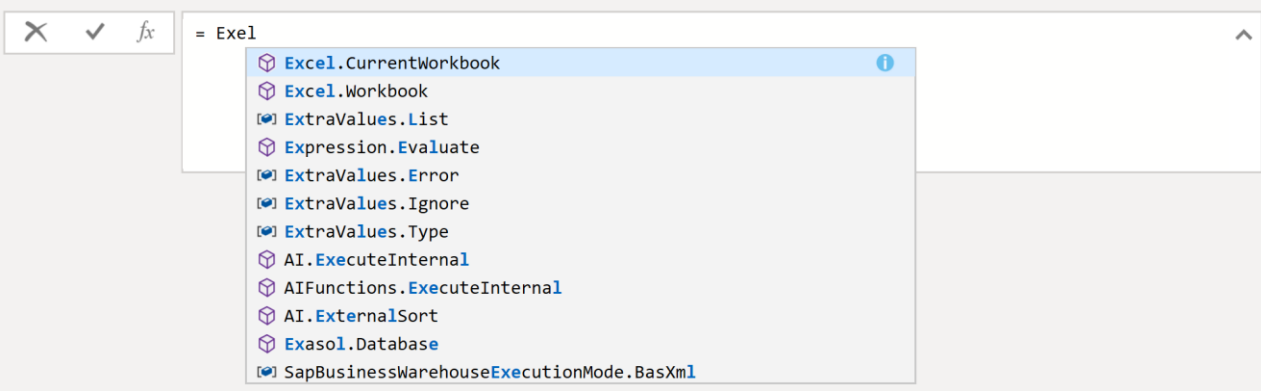
В языке M есть зарезервированные последовательности символов, которые нельзя использовать как операторы, это:

and as each else error false if in is let meta not otherwise or
section shared then true try type #binary #date #datetime #datetimezone
#duration #infinity #nan #sections #shared #table #time

Также есть стандартные операторы и знаки пунктуации:

, ; = < <= > >= <> + - * / & () [] { } @ ! ? =>

Редактор формул



= #shared

Весь перечень формул можно вывести, написав в строке формул = #shared
Появится список формул. Если щелкнуть по формуле, в нижней части экрана появится справочная информация. Из списка можно выбрать нужную формулу и тут же добавить в запрос

The screenshot shows the Microsoft Power Query Editor interface. At the top, the title bar reads "Без имени — Редактор Power Query". The ribbon includes tabs for "Средства для записей" (Records Tools) and "Преобразовать" (Transform). The main area is divided into three panes:

- Left Pane (Queries):** Labeled "Запросы [4]", it lists "Численность населения", "Запрос1", "Запрос2", and "Запрос3".
- Center Pane (Formula Bar and List):** The formula bar at the top contains "= #shared". Below it is a scrollable list of functions and tables, including "Численность населения" (Table), "Запрос1" (Table), "Запрос2" (Table), "Запрос3" (Record), and various Microsoft Azure and Power BI functions.
- Right Pane (Query Parameters):** Labeled "Параметры запроса", it contains sections for "СВОЙСТВА" (Properties) and "ПРИМЕНЕННЫЕ ШАГИ" (Applied Steps). The "СВОЙСТВА" section shows the query name "Запрос3".

= #shared

Весь перечень формул можно вывести, написав в строке формул = #shared
Появится список формул. Если щелкнуть по формуле, в нижней части экрана появится справочная информация. Из списка можно выбрать нужную формулу и тут же добавить в запрос

TEXTAL	Function
Text.From	Function
Text.Length	Function
Text.Range	Function

function (text as nullable text) as nullable number
Возвращает число символов в тексте `text`.
Пример: Определить, сколько символов содержится в тексте "Hello World".
Использование:
`Text.Length ("Hello World")`
Выходные данные:
11

Логические выражения if

Логические выражения записываются как if...then...else

if [Sales] > 1000 then "более 1000" else if [Sales] > 500 then "более 500" else "500 или меньше"

Оператор let

Выражение let позволяет вычислять набор значений, назначить имена, а затем используется в следующем выражении, которое записывается в in.

Например:

```
let  
Sales2017 =[ Year = 2017, FirstHalf = 1000, SecondHalf = 1100, Total = FirstHalf + SecondHalf ], // 2100  
Sales2018 =[ Year = 2018, FirstHalf = 1200, SecondHalf = 1300, Total = FirstHalf + SecondHalf ] // 2500  
in  
Sales2017[Total] + Sales2018[Total] // 4600
```

Оператор let

Для сложных запросов могут использоваться **вложения let**, что позволяет сделать запрос более читаемым.

Например:

```
let
    AreaCalculation = (x, y) =>
        let
            Area = x * y,
            DoubleArea = Area * 2
        in
            DoubleArea
in
    AreaCalculation
```

Списки

Список (list) – упорядоченная последовательность значений.

Фигурные скобки { и } обозначают начало и конец списка.

Например:

{1, 2, 3} – список из чисел,

{ 1 .. 365 } – список из чисел от 1 до 365,

{ 1, 5 .. 8, 11 } – числа 1, 5, 6, 7, 8, 11,

{ } – пустой список,

{ {1, 2, 3}, { 4 .. 6 } } – список из нескольких списков.

Списки можно преобразовать в таблицу с помощью контекстного меню Средства для списков -> Преобразование -> В таблицу.

Функции для создания списков

1. List.Numbers

Возвращает список чисел для заданного исходного значения, количества значений и значения приращений.

List.Numbers(start, count, optional increment)

start – исходное значение в списке (тип number),

count – количество значений, которое требуется создать в списке (number),

optional increment (необязательно) – значение, на которое выполняется

увеличение, т.е. шаг увеличения (number).

Пример: создание списка из 10 чисел, начинающийся с 1, с шагом увеличения +2

= List.Numbers(1, 10, 2)

Получится список: { 1, 3, 5, 7, 9, 11, 13, 15, 17, 19 }

Функции для создания списков

2. List.Dates

Возвращает список дат с заданным числом значений, начиная с даты начала, с указанным шагом приращения #duration.

List.Dates(start, count, step)

start – дата начала (тип date),

count – количество дат в списке (number),

step – шаг приращения (duration).

Пример: создание списка из пяти дат, начиная с 31.12.2011, с шагом приращения в один день.

= List.Dates(#date(2011, 12, 31), 5, #duration(1, 0, 0, 0))

Получится список:

31.12.2011

01.01.2012

02.01.2012

03.01.2012

04.01.2012

Функции для создания списков

3. Table.ToList

Преобразует таблицу в список с помощью функции объединения для каждой строки таблицы.

`Table.ToList(table, optional combiner)`

`table` – таблица для преобразования (тип `table`),

`optional combiner` – функция, которая применяется к строке таблицы для создания

единичного значения (nullable function).

Пример:

```
Table.ToList(  
    Table.FromRows( {  
        { Number.ToText(1), "Иванов", "123-4567" },  
        { Number.ToText(2), "Петров", "987-6543" },  
        { Number.ToText(3), "Сидоров", "543-7890" } } ),  
    Combiner.CombineTextByDelimiter( " ", " " )
```

)

Получится список:

	Список
1	1, Иванов, 123-4567
2	2, Петров, 987-6543
3	3, Сидоров, 543-7890

Функции для создания списков

4. Table.Column

Возвращает в виде списка столбец данных из указанного столбца таблицы.

Table.Column(table, column)

table – таблица (тип table),

column – столбец (text).

Пример:

Table.Column(

Table.FromRecords(

{ [CustomerID = 1, Name = "Иванов", Phone = "123-4567"],

[CustomerID = 2, Name = "Петров", Phone = "987-6543"],

[CustomerID = 3, Name = "Сидоров", Phone = "543-7890"] }

),

"Name")

Получится список:

	Список
1	Иванов
2	Петров
3	Сидоров

Функции обработки списков

Функция	Описание
List.Average	Возвращает среднее значение для элементов в списке. Результат вычисления будет с тем же типом данных, что и значения в списке.
List.Count	Определяет число элементов в списке.
List.Max	Возвращает максимальное значение в списке (или необязательное значение, если список пуст).
List.Min	Возвращает минимальное значение в списке (или необязательное значение, если список пуст).
List.Mode	Определяет элемент, который чаще всего появляется в списке. Если несколько элементов появляются одинаково часто, выбирается последний из них.
List.Product	Рассчитывает произведение чисел в списке, отличных от null.
List.Select	Возвращает список значений из заданного списка, которые удовлетворяют условию отбора.
List.Sort	Сортирует список по необязательным указанным критериям.
List.StandardDeviation	Возвращает оценку стандартного отклонения значений в списке.
List.Sum	Возвращает сумму всех значений в списке, отличных от null.

Записи

Запись (record) – это упорядоченная последовательность полей, где каждое поле имеет имя и одно значение (любого типа).

Примеры:

[x = 1, y = 2]

[firstname = "Иван", lastname = "Иванов"]

[Sales2016 = 1000, Sales2017 = 1100, TotalSales = Sales2016 + Sales2017]

[] – пустая запись

Запись можно преобразовать в таблицу с помощью контекстного меню

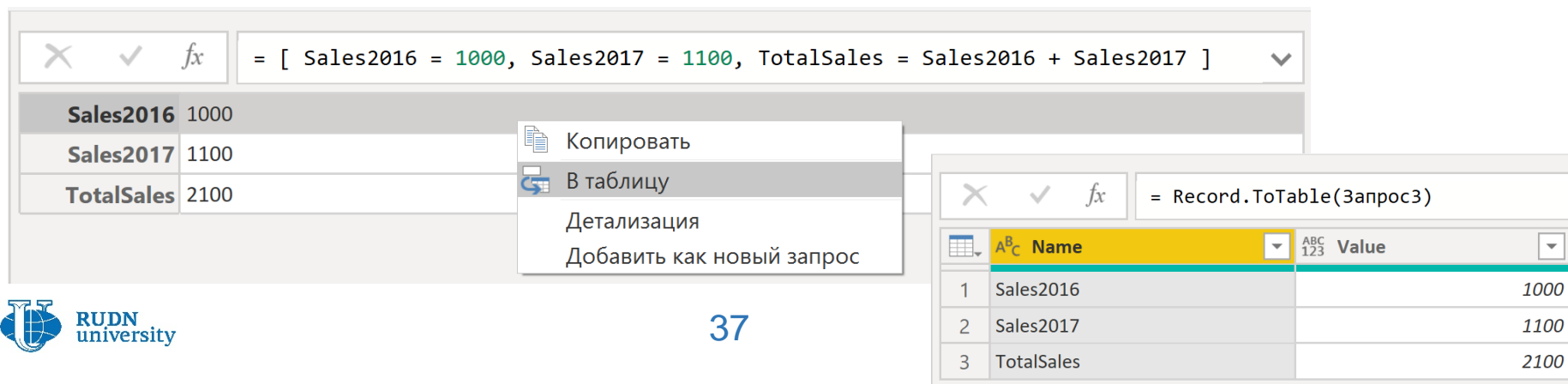
В таблицу.

Для создания записи кроме квадратных скобок можно использовать формулы

Record.FromList, Record.FromTable.

Например, запись

Record.FromList({1, 2}, {"a", "b"}) – то же самое, что и [a = 1, b = 2]



Formula bar: `= [Sales2016 = 1000, Sales2017 = 1100, TotalSales = Sales2016 + Sales2017]`

Sales2016	1000
Sales2017	1100
TotalSales	2100

Context menu options:

- Копировать
- В таблицу**
- Детализация
- Добавить как новый запрос

Resulting table structure:

ABC	Name	ABC	Value
1	Sales2016	1000	
2	Sales2017	1100	
3	TotalSales	2100	

Formula bar for result table: `= Record.ToTable(Запрос3)`

Функции обработки записей

Функция	Описание
Record.FieldCount	Определяет число полей в записи
Record.FieldNameNames	Возвращает названия полей в записи в виде текста
Record.ReorderFields	Изменяет порядок полей в записи
Record.ToTable	Преобразует запись в таблицу

Обращение к данным в списках и записях

Записи могут входить в списки. Можно обратиться к элементу в списке с помощью числового индекса, указанного в фигурных скобках { }.

В отличие от обычного Excel, в Power Query порядковые номера записей начинаются с нулевого индекса, а не с первого. Например, индексы 0 и 1 используются для ссылки на 1-ое и 2-ое значение в списке:

```
[  
Продажи =  
{  
[ Год = 2016, ПерваяПоловина = 1000, ВтораяПоловина = 1100,  
Всего = ПерваяПоловина + ВтораяПоловина ],  
[ Год = 2017, ПерваяПоловина = 1200, ВтораяПоловина = 1300,  
Всего = ПерваяПоловина + ВтораяПоловина ]  
},  
ВсегоПродажи = Продажи{0}[Всего] + Продажи{1}[Всего]  
]
```

Продажи	List
ВсегоПродажи	4600

Индексы

Если вы используете индекс, которого нет в списке, например Продажи{3}[Всего], по умолчанию будет возвращена ошибка. Но если добавить оператор ? в конец выражения — Продажи{3}[Всего]?, вы получите вместо ошибки пустое значение null.

Примеры обращения к элементам списков:

```
{1, [A=2], 3} {1}? // [A=2]
```

```
{true, false} {2}? // null
```

Примеры обращения к записям:

```
[A=1, B=2] [B] // 2
```

```
[A=1, B=2] [C]? // null
```

```
[A=1, B=2] [[B]] // [B=2]
```

```
[A=1, B=2] [[B], [C]]? // [B=2,C=null]
```


Таблицы

Создание таблиц

1. #table

С помощью #table можно создать таблицу, указав список имен заголовков и список строк. Например:

```
#table(  
  {"Подразделение", "Продажи"},  
  {"Москва", 1000}, {"Омск", 200} )
```

	ABC 123 Подразделение	ABC 123 Продажи
1	Москва	1000
2	Омск	200

С помощью #table также можно задать типы данных:

```
#table(  
  type table [Подразделение = text, Продажи = number],  
  {"Москва", 1000}, {"Омск", 200} )
```

	A ^B C Подразделение	1.2 Продажи
1	Москва	1000
2	Омск	200

2. Table.FromRows

Создает таблицу из списка с элементами значений для одной строки. Дополнительно можно указать необязательные наименования столбцов, тип таблицы или число столбцов.

Пример:

```
Table.FromRows(  
{  
    { 1, "Покупатель 1", "123-4567" },  
    { 2, "Покупатель2", "987-6543" }  
},  
    {"CustomerID", "Name", "Phone"}  
)
```

	ABC 123 CustomerID	ABC 123 Name	ABC 123 Phone
1	1	Покупатель 1	123-4567
2	2	Покупатель2	987-6543

3. Table.FromRecords

Создает таблицу из списка записей. Пример:

```
Table.FromRecords(  
{  
    [CustomerID = 1, Name = "Bob", Phone = "123-4567"],  
    [CustomerID = 2, Name = "Jim", Phone = "987-6543"],  
    [CustomerID = 3, Name = "Paul", Phone = "543-7890"]  
}  
)
```

Дополнительно можно указать тип данных:

```
Table.FromRecords(  
    { [CustomerID=1, Name="Bob"] },  
    type table[ CustomerID = Number.Type, Name = Text.Type ]  
)
```

Обращение к данным в таблицах

Формула	Описание
Excel.CurrentWorkbook	Дает доступ к объектам в текущем файле: форматированные таблицы excel, именованные диапазоны, связи рабочей книги.
Excel.CurrentWorkbook() {[Name="Sales"]}[Content]	обращается к содержимому таблицы SalesTable в текущей книге.
Excel.Workbook	Выводит запись из листов в книге excel.

Формулы для работы с данными в таблицах

Функция	Описание
Table.RowCount	Определяет количество строк в таблице.
Table.ColumnCount	Определяет количество столбцов в таблице.
Table.Group	Группирует строк таблицы по значениям в указанном ключевом столбце для каждой строки. Для каждой группы создается запись, содержащая ключевые столбцы и их значения вместе со всеми агрегированными столбцами.
Table.Sort	Сортирует таблицу, используя список из одного или нескольких имен столбцов и необязательного параметра comparisonCriteria
Table.FindText	Выбирает из таблицы строки с указанным текстом. Если текст не найден, возвращается пустая таблица.
Table.First	Возвращает первую строку из таблицы или необязательное значение, если таблица пуста.
Table.LastN	Возвращает последние строки таблицы в зависимости от указанного значения. Если значение – число, то будет возвращено соответствующее число строк. Если указано условие, то строки возвращаются в порядке возрастания до тех пор, пока не будет встречена строка, не соответствующая условию.
Table.Repeat	Возвращает таблицу со строками, повторенными указанное число раз.
Table.Column	Возвращает указанный столбец данных из таблицы в виде списка.
Table.PromoteHeaders	Назначает первую строку таблицы в качестве новых заголовков столбцов.
Table.Max	Определяет наибольшую строку в таблице, исходя из критериев.
Table.Min	Определяет наименьшую строку в таблице, исходя из критериев.
Table.Buffer	Помещает таблицу в буфер памяти, изолируя ее от внешних изменений во время оценки.

Пользовательские функции в Power Query

Функция (function) преобразует набор входящих значений в одно результирующее.

Функция записывается путем перечисления параметров функции в круглых скобках (), за которыми следует знак перехода => далее указывают выражение, определяющее функцию.

Для выполнения функции ей передают набор входящих параметров. Например:

```
[  
    MyFunction = (x, y, z) => x + y + z, // записали функцию,  
    которая складывает три числа  
    Result = MyFunction(1, 2, 3) // передали функции цифры  
    1,2,3 для суммирования  
]
```

```
let  
    MyFunction = (x as number, y as number, z as number) as number  
    => x + y + z,  
    Result = MyFunction(1, 2, 3)  
in  
    Result
```

each

let

```
Источник = Excel.CurrentWorkbook(){[Name= "продажи"]}[Content],  
Суммирование = Table.AddColumn(Источник, "сумма",  
each MyFunction([1 квартал],[2 квартал],[3 квартал]))
```

in

Суммирование

Рекурсивные функции @

Это функции, вызывающие сами себя в тексте с помощью знака @. Например, с помощью рекурсивной функции рассчитан факториал:

```
let
    Factorial = (x) => If x = 0 then 1 else x * @Factorial(x - 1),
    Result = Factorial(3) // Факториал 3! = 1 · 2 · 3 = 6
in
    Result
```


Спасибо за внимание!



Шевцов Василий Викторович

shevtsov_vv@rudn.university
+7(903)144-53-57