Составитель: Нуруллин А.Р., волшебник 0,5 ст.

Лабораторная работа №
Таблица строк (StringGrid).

Часть 1. Таблица строк

Таблица строк – компонент позволяющий работать с текстовой информацией в двумерной таблице, имеющей столбцы и строки.

Основные свойства:

1. Cells - двумерный массив, позволяющий обращаться к содержимому ячеек и изменять их содержимое. Первое измерение – номер столбца, второе – номер строки (*НУМЕРАЦИЯ НАЧИНАЕТСЯ С НУЛЯ*).

Например:

StringGrid1.Cells[0 , 2] := ‘ ТЕКСТ ’; *// ячейке, находящейся в нулевом столбце и второй строке присваиваем значение ‘текст’*

2. Height – высота области выделенной под таблицу (не самой таблицы)

3. Width - ширина области выделенной под таблицу (не самой таблицы)

4. ColCount – число столбцов таблицы

5. RowCount – число строк таблицы

6. FixedCols – число фиксированных (серых) столбцов

7. FixedRows – число фиксированных (серых) строк

6. DefaultColWidth – ширина столбцов по умолчанию

7. DefaultRowHeight – высота строк по умолчанию

8. GridLineWidth – ширина линии-разделителя ячеек

9. Options – свойство включающее в себя ряд других, направленных на работу с таблице.

Например:

Options.goEditing – возможность редактирования содержимого ячеек (**True** - можно)

Options.goRowSizing – возможность изменения высоты строк

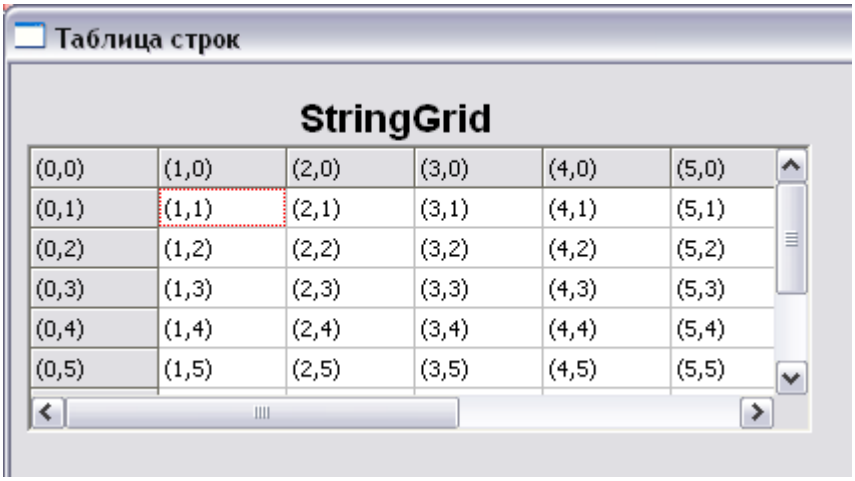
Options.goColSizing – возможность изменения ширины столбцов

(…ДАЛЕЕ ВЫПОЛНЯЕМ РИСУНОК С ДОСКИ…)

Пример:

Следующий код программы изменяет размер таблицы, помещенной на форму в режиме проектирования (по умолчанию размер 5x5), на размер 10x10 ячеек и заполняет строками, содержащими координаты ячеек (см. рис.). (свойство **GridLineWidth** по умолчанию установите равным **1**)

```
procedure TForm1.FormCreate(Sender: TObject);
var Col,Row: integer; // Col – колонка (столбец), Row - строка
begin
   StringGrid1.ColCount:=10;
   StringGrid1.RowCount:=10;
   for Col:= 0 to StringGrid1.ColCount - 1 do
      for Row:=0 to StringGrid1.RowCount - 1 do
         StringGrid1.Cells[ Col , Row ]:=' ( ' + IntToStr(Col) + ' , ' + IntToStr(Row) + ' ) ';
end;
```



Вы можете заметить, что на рис. размеры таблицы превышают размеры области, выделенной под таблицу, в результате чего появляются полосы прокрутки. Уберите полосы прокрутки, задав значение свойства ***ScrollBars = ssNone***.

Для того чтобы область, выделенная под таблицу автоматически становилась примерно равной размерам самой таблицы необходимо произвести вычисление ее размеров (**Width** и **Height**).

Размеры самой таблицы, например ширина, складывается из следующих величин:

(ширина столбца) **x** ***(кол-во столбцов)*** **+** ***(толщина линии разделителя)*** **x** ***(кол-во столбцов + 1)***

Т.е.

StringGrid1.Width : = StringGrid1.DefaultColWidth * StringGrid1.ColCount +
+ StringGrid1.GridLineWidth * (StringGrid1.ColCount + 1);

Добавив эту строку в программу ширина области, выделенной под таблицу будет примерно такой же как и сама таблица.

Задание 1

Самостоятельно дополните программу кодом, который автоматически высчитывает высоту области, выделенной под таблицу, делая ее такой же, как и сама таблица.

Часть 2. Выделение областей таблицы

По опыту работы в табличных процессорах (MS Excel, Open Office Calc, и т.д.) мы знаем, чтобы обрабатывать данные, находящиеся в таблице, необходимо выделять некоторые области. Поддержку возможности выделения областей таблицы обеспечивает свойство ***goRangeSelect*** таблицы строк (придайте ему значение **True**). Т.о. мы получим возможность выделять ячейки (см.рис.)

(0,0)	(1,0)	(2,0)	(3,0)	(4,0)	(5,0)
(0,1)	(1,1)	(2,1)	(3,1)	(4,1)	(5,1)
(0,2)	(1,2)	(2,2)	(3,2)	(4,2)	(5,2)
(0,3)	(1,3)	(2,3)	(3,3)	(4,3)	(5,3)
(0,4)	(1,4)	(2,4)	(3,4)	(4,4)	(5,4)
(0,5)	(1,5)	(2,5)	(3,5)	(4,5)	(5,5)

На данном рисунке выделена область ячеек с левой верхней (2-ой столбец, 2-ая строка) до правой нижней (4-ый столбец, 4-ая строка).

Программно обратиться к выделенной области (например, чтобы посчитать сумму чисел находящихся в выделенных ячейках) можно с помощью свойства **Selection,** которое имеет сложный тип и состоит из ряда подсвойств ***Left, Top, Right, Bottom***.

Left – номер левого столбца выделения
Top – номер верхней строки выделения
Right – номер правого столбца выделения
Bottom – номер нижней строки выделения

На рис. приведенном ниже эти свойства принимают следующие значения
StringGrid1.Selection.Left = 1, StringGrid1.Selection.Top = 2 – координаты верхней левой ячейки
StringGrid1.Selection.Right = 3, StringGrid1.Selection.Bottom = 4 – правая нижняя ячейка

(0,0)	(1,0)	(2,0)	(3,0)	(4,0)	(5,0)
(0,1)	(1,1)	(2,1)	(3,1)	(4,1)	(5,1)
(0,2)	(1,2)	(2,2)	(3,2)	(4,2)	(5,2)
(0,3)	(1,3)	(2,3)	(3,3)	(4,3)	(5,3)
(0,4)	(1,4)	(2,4)	(3,4)	(4,4)	(5,4)
(0,5)	(1,5)	(2,5)	(3,5)	(4,5)	(5,5)

Задание 2

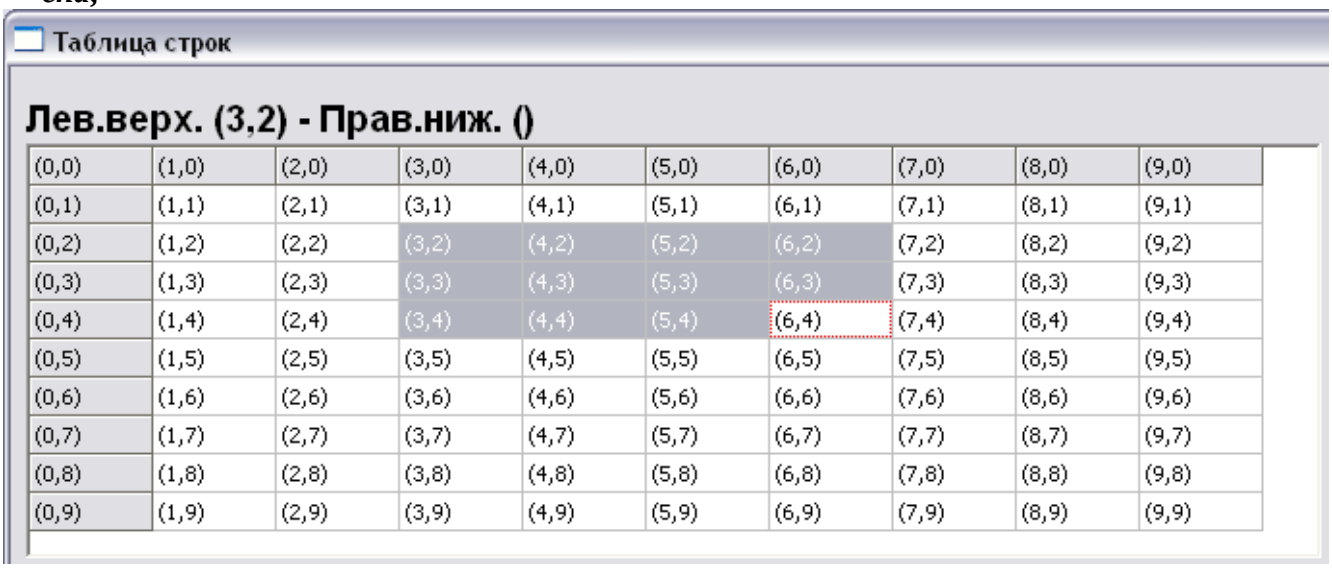
Написать программу, которая по событию отпускания кнопки мыши onMouseUp, выводит на форму координаты крайних ячеек выделенной области.

Событие onMouseUp используется именно потому, что оно происходит как раз тогда, когда мы заканчиваем выделять область и отпускаем кнопку мыши.

Например, следующий код в процедуре **onMouseUp** выводит координаты левой верхней ячейки из выделенных.

```
procedure TForm1.StringGrid1MouseUp(Sender: TOBject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
```

```
begin
  Label1.Caption:=' Лев.верх. ( ' + IntToStr( StringGrid1.Selection.Left ) + ' , ' +
                                  IntToStr( StringGrid1.Selection.Top ) + ' ) – Прав.ниж.. ( ' + ' )';
end;
```



(0,0)	(1,0)	(2,0)	(3,0)	(4,0)	(5,0)	(6,0)	(7,0)	(8,0)	(9,0)
(0,1)	(1,1)	(2,1)	(3,1)	(4,1)	(5,1)	(6,1)	(7,1)	(8,1)	(9,1)
(0,2)	(1,2)	(2,2)	(3,2)	(4,2)	(5,2)	(6,2)	(7,2)	(8,2)	(9,2)
(0,3)	(1,3)	(2,3)	(3,3)	(4,3)	(5,3)	(6,3)	(7,3)	(8,3)	(9,3)
(0,4)	(1,4)	(2,4)	(3,4)	(4,4)	(5,4)	(6,4)	(7,4)	(8,4)	(9,4)
(0,5)	(1,5)	(2,5)	(3,5)	(4,5)	(5,5)	(6,5)	(7,5)	(8,5)	(9,5)
(0,6)	(1,6)	(2,6)	(3,6)	(4,6)	(5,6)	(6,6)	(7,6)	(8,6)	(9,6)
(0,7)	(1,7)	(2,7)	(3,7)	(4,7)	(5,7)	(6,7)	(7,7)	(8,7)	(9,7)
(0,8)	(1,8)	(2,8)	(3,8)	(4,8)	(5,8)	(6,8)	(7,8)	(8,8)	(9,8)
(0,9)	(1,9)	(2,9)	(3,9)	(4,9)	(5,9)	(6,9)	(7,9)	(8,9)	(9,9)

Заметьте, если выделить всего одну ячейку, т.е. попросту щелкнуть на ней, координаты левой верхней ячейки выделенной области (**Selection.Left** и **Selection.Top**) и координаты правой нижней ячейки (**Selection.Right** и **Selection.Bottom**) совпадут.

Часть 3. Вычислительные возможности

Поскольку мы можем программно обращаться к выделенным областям, то мы теперь можем обрабатывать каким-либо образом выделенные данные, например, подсчитывать их сумму, среднее значение в некотором диапазоне и т.д.

Задайте объекту **«таблица строк»** короткое имя, например **SG** (**S**tring**G**grid),

Свойству **Cursor** задайте значение **crHandPoint.** (вид курсора при наведении на таблицу)

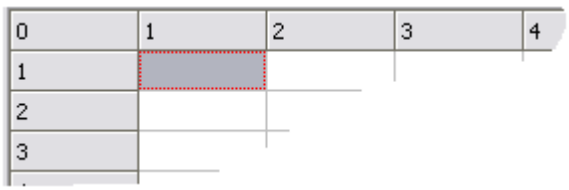
Свойству **Options.goDrawFocusSelected = True** (при выеделении последняя, активная ячейка также будет выделенной)

Задание 3

Требуется написать простейший табличный процессор, работающий на подобии программ MS Excel, Calc и т.д. Программа должна поддерживать возможности подсчета суммы, среднего значения и нахождения минимального элемента в выделенных строках и столбцах.

Шаг 1

Нумеруем заголовки строк и столбцов их номерами (фиксированные ячейки) в процедуре **FormCreate.**



0	1	2	3	4
1				
2				
3				

```
procedure TForm1.FormCreate(Sender: TObject);
var Col, Row: integer;
begin
SG.ColCount:=10; // по умолчанию 10 колонок
 SG.RowCount:=10; // по умолчанию 10 строк
for Col:= 0 to SG.ColCount - 1 do SG.Cells[Col ,0]:=
                                        IntToStr( Col  );
end;
// нумерация колонок готова, нумерацию строк сделать самостоятельно
```

Шаг 2

Приводим форму к виду, как на рис. ниже (размеры таблицы задаются автоматический в процедуре **FormCreate** после запуска проекта, этот код можно взять из частей 1 и 2 лабораторной работы).

Создаем обработчик события по нажатию на кнопку СУММА (по столбцам)

Здесь алгоритм прост – считаем суму значений ячеек в диапазоне от верхней выделенной (**Options.Top**) до нижней выделенной (**Options.Bottom**), все это происходит в текущей выделенной колонке (**Selection.Left** или **Selection.Right** если выделена всего одна колонка). Т.о. после выделения колонки и нажатия на кнопку СУММА в ячейке которая находится ниже выделенной области появляется значение суммы (см.рис.)

	3,7
	3,1
	5,63
Сумма:	12,43

```
procedure TForm1.Button1Click(Sender: TObject);
var Col, Row: integer;
        S :real;
begin
    Col := SG.Selection.Left;
    for Row := SG.Selection.Top to SG.Selection.Bottom do S:=S + StrToFloat( SG.Cells[ Col , Row ]);
    SG.Cells[ Col , SG.Selection.Bottom + 1 ]:=FloatToStr( S );
end;
```

0	1	2	3	4	5	6	7	8	9
1									
2									
3									
4		3,7		78		11		92,7	<<-- Сумма
5		3,1		5		9,3		17,4	<<--Миним
6		5,63		45		3		53,63	<<--Средн.
7	Сумма:	12,43	Миним:	5	Среднее:	7,76			
8									
9									

Шаг 3

Задача усложняется, необходимо чтобы по нажатию на кнопку СУММА (По столбцам) считалась сумма в нескольких выделенных столбцах (используйте для этого свойства ***Selection.Left*** и ***Selection.Right***)

	3,7	2,3	78
	3,1	4,3	5
	5,63	1,2	45
Сумма:	12,43	7,8	128

			Сумма:
3,7	2,3	78	84
3,1	4,3	5	12,4
5,63	1,2	45	51,83

Шаг 4

Запрограммируйте аналогичным образом кнопку СУММА (По строкам)

Шаг 5

После того, как кнопки СУММА готовы, можно приступать к другим кнопкам минимального значения и среднего значения. Они работают по тем же принципам, только алгоритмы вычисления результата соответствуют названию кнопки.