Практическая работа №5, ВАШ ВЕС

Постановка задачи

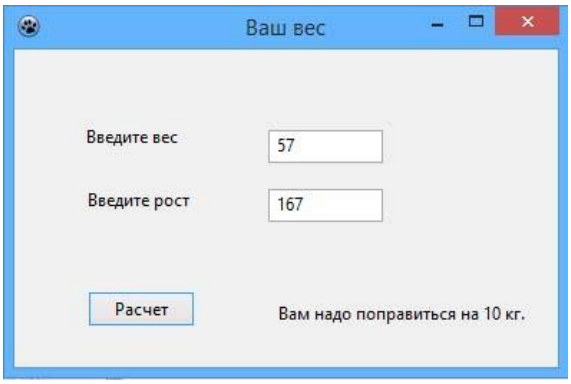

Рис.5.1.

Пусть оптимальный вес человека определяется как рост минус 100 см. Если фактический вес человека меньше оптимального, то будем считать его худым, если больше, то полным.

Создать программу, выполняющую следующие действия.

Введя рост и фактический вес, нажав кнопку «Расчет», пользователь может определить, худой он или полный и на сколько килограммов надо поправиться или похудеть (рис.5.1.).

Новым в этой работе являются:

- использование числовых типов переменных – целочисленного и действительного (Integer и Real);
- преобразование строковых данных в числовой тип и числовые в строковые с помощью функций **StrToInt**, **StrToFloat**, **IntToStr FloatToStr**;
- обработка исключительных ситуаций с помощью оператора **Try – except – end;**
- использование процедуры **ShowMessage** для вывода сообщения в отдельном окне.

План разработки программы

1. Откройте новый проект.

2. Разместите в форме экземпляры компонентов в соответствии с рис.5.2. В поле **Edit1** будем вводить вес в килограммах, а в **Edit2** – рост в сантиметрах.

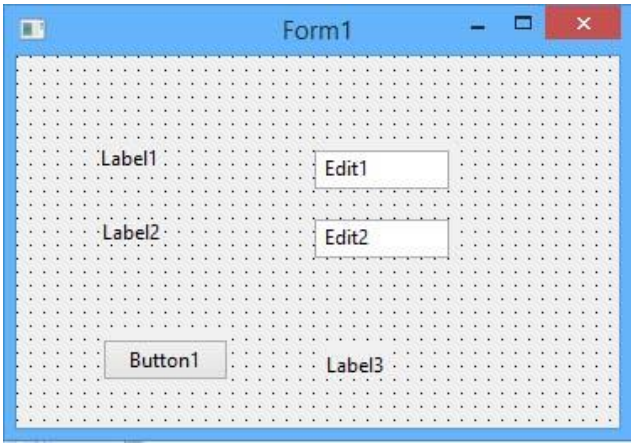

Рис.5.2.

3. Сохраните код программы и проект под именами, например, **Main.pas** и **Weight.lpr**.

4. Для сохранения результатов расчета введем переменные:

faktW – фактический вес,
optW – оптимальный вес,
Rost – рост,
Delta – разница между оптимальным весом и фактическим.

В начале будем считать, что все данные у нас целые числа.

В блоке реализации после слова **implementation** разместите описание переменных:

```
VAR
factW, optW, Rost, Delta : Integer;
```

5. Выполните следующие действия:

Выделенный объект	Вкладка окна **Инспектор объектов**	Имя свойства/ Имя события	Значение/Действие
Label1	Свойства	Caption	Введите вес
Label2	Свойства	Caption	Введите рост
Label3	Свойства	Caption	Удалить название объекта
Edit1	Свойства	Text	Удалить название объекта
	События	OnKeyPress	`if key=#13 then Form1.ActiveControl := Edit2;` **Комментарий** Делает активным объект **Edit2**, т.е. после окончания ввода фокус перейдет в окно ввода **Edit2**.
Edit2	Свойства	Text	Удалить название компонента
	События	OnKeyPress	`if key=#13 then Button1.SetFocus;` **Комментарий** Перемещает курсор на компонент **Button1**.

Button1	События	OnClick	`factW := StrToInt(Edit1.text);`
			`Rost := StrToInt(Edit2.Text);`
			`OptW :=Rost - 100;`
			`Delta := abs(factW - OptW);`
			`if OptW = factW then`
			`Label3.caption := 'Ваш вес идеален! '`
			`else`
			`if OptW > factW then`
			`Label3.caption:= 'Вам надо поправиться на '`
			`+IntToStr(Delta)+ ' кг. '`
			`else`
			`Label3.caption:= 'Вам надо похудеть на '`
			`+IntToStr(Delta)+ ' кг. '`

Комментарий

а) Компонента **Edit** содержит информацию строкового типа, поэтому нам необходимо для выполнения вычислений перевести ее в числовой вид.

б) После выполнения арифметических действий результат вычислений нужно будет разместить на форме в компоненте **Label**, которая так же может содержать только информацию строкового типа.

в) Функция **StrToInt** преобразует строку символов в целое число, функция **IntToStr** выполняет обратное действие – целое число преобразует в строку символов.

6. Сохраните проект, запустите и протестируйте его.

7. Усовершенствуйте программу так, чтобы можно было вводить любые десятичные величины. Для этого необходимо использовать вещественный тип переменных **Real**:

```
VAR
factW, optW, Rost, Delta : Real;
```

Комментарий

Преобразование действительных чисел в строковый тип и строковый тип в действительное число выполняется с помощью функций: **FloatToStr** и **StrToFloat**.

Внесите соответствующие изменения в обработку события **OnClick** компонента **Button1**.

8. Сохраните проект окончательно, запустите и протестируйте его.

Немного теории

В случае преобразования строкового типа в числовой тип может возникнуть ситуация появления ошибки, если введены недопустимые символы. Если функции **StrToInt** или **StrToFloat** обнаружат ошибку в записи числа, они инициируют так называемую исключительную ситуацию, которая обычно приводит к аварийному завершению работы программы. Но существует возможность не допустить аварийное завершение программы. Для этого используется *обработчик исключительных ситуаций*:

Try
<защищенный блок операторов>
except
<обработка исключительной ситуации>
end;

Если при выполнении операторов из защищенного блока возникнет исключительная ситуация, управление будет передано в блок операторов, располагающийся между ключевыми словами **except** и **end**. Но если обработка пройдет без ошибок, блок обработки исключительной ситуации игнорируется и управление передается оператору, следующему за ключевым словом **end**.

Пример использования обработки исключительной ситуации для процедуры **Edit1KeyPressed** может выглядеть так:

```
try
 FactW:=StrToInt(Edit1.Text);
except
 ShowMessage('Ошибочная запись числа: ' + Edit1.Text);
 Edit1.SetFocus;
 Exit;
end;
```

В результате выполнения оператора

```
FactW:=StrToInt(Edit1.Text);
```

если возникнет исключительная ситуация, то процедура **ShowMessage** выведет на экран простое диалоговое окно, содержащее строку с текстом и кнопку «OK». Эта процедура используется для сообщения пользователю какой-либо информации и не требует принимать каких-либо решений. После появления окна работа программы

приостановится в ожидании реакции пользователя.

При вызове стандартной процедуры **Exit** снова активизируется окно редактора (компонента **Edit1**), в котором обнаружен ошибочный текст.

9. Внесите необходимые изменения для обработки исключительных ситуаций, возникающих при вводе чисел.

Задание для самостоятельного выполнения

1. Усовершенствуйте проект:

a) сделайте к программе заголовок,

b) сделайте шрифт выводимой реплики отличным от стандартного по виду, цвету и размеру,

c) вставьте кнопку выхода из программы,

d) предусмотрите возможность повторного запуска программы (см. Проект № 3 «Диалог»).

2. Сделайте так, чтобы в начале программы и после повторного запуска объект **Button1** был не доступен и только после того, как будет введен вес, появлялась возможность нажать на кнопку «Расчет».

Подсказка. Изменение свойства доступности компонента для пользователя – **Enabled**. Если свойство имеет значение **True**, то компонент доступен, а если значение **False**, то – не доступен (см. Проект № 3 «Диалог»).

3. Сделайте так, чтобы в начале программы и после повторного запуска объекты **Label2** и **Edit2** были не видны и появлялись на экране только после того, как будет введен вес.

Подсказка. Изменение свойства видимости компонента – **Visible.** Если свойство имеет значение **True**, то компонент виден, а если значение **False**, то – не виден.

4. Предусмотрите невозможность ввода отрицательных значений веса и роста.

5. Измените алгоритм расчета с учетом *Индекса массы тела*.

Вес – X, Рост – Y.

Индекс массы тела – A, где $A = X / Y^2$ (кг/м^2)

Результат определяется по таблице:

№ п/п	Значение индекса	Результат (сообщение, которое надо вывести)
1.	A < 18	Большой недовес
2.	18 <= A < 20	Маловато и небезопасно, можно получить истощение
3.	20 <= A <= 25	Идеально
4.	26 <=A <=30	Легкий недобор
5.	30 < A	Срочно нужно худеть

Внимание

Если вы измените свойство Name объекта Form1 на fMain, то необходимо внести изменение в событие OnKeyPress объекта Edit1:

```
if key=#13 then fMain.ActiveControl := Edit2;
```