|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Мытищинский филиал**  **федерального государственного бюджетного образовательного**  **учреждения высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МФ МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ: Космический

КАФЕДРА: «Прикладная математика, информатика и вычислительная техника» К3-МФ

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОЙ РАБОТЕ***

***НА ТЕМУ:***

**Конвертер валют**

Студент К3-45Б \_\_\_\_\_\_\_\_\_\_\_\_\_\_ **А. М. Шехаб**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсовой работы  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ **А. М. Ветошкин**

(Подпись, дата) (И.О.Фамилия)

*2021 г.*

**Министерство науки и высшего образования Российской Федерации**

**Мытищинский филиал**

**федерального государственного бюджетного образовательного**

**учреждения высшего образования**

**«Московский государственный технический университет**

**имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МФ МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ

Заведующий кафедрой \_\_\_\_\_\_\_\_\_\_

(Индекс)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(И.О.Фамилия)

« \_\_\_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 20 \_\_\_\_ г.

**ЗАДАНИЕ**

**на выполнение курсовой работы**

по дисциплине: Методы разработки ПО (программного обеспечения)

Студент группы: К3-45Б

Шехаб Адам Мохамад

(Фамилия, имя, отчество)

Тема курсовой работы: Конвертер валют

Индивидуальное задание: Создать окно перевода из одной валюты в другую, используя библиотеку Turbo Vision.

***Оформление курсовой работы:***

Расчетно-пояснительная записка на 22 листах формата А4.

Дата выдачи задания « \_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

**Руководитель курсовой работы**  **А. М. Ветошкин**

(И.О.Фамилия)

**Студент**  **А. М. Шехаб**

(И.О.Фамилия)

Оглавление

[**Введение** 4](#_Toc75133461)

[**Описание принципа работы программы** 5](#_Toc75133462)

[1. Main (Converter.pas) 5](#_Toc75133463)

[2. ModuleApp (ModuleApp.pas) 6](#_Toc75133464)

[3. ModuleTasks (ModuleTasks.pas) 10](#_Toc75133465)

[**Заключение** 15](#_Toc75133466)

[**Литература:** 16](#_Toc75133467)

[**Ссылки на источники:** 16](#_Toc75133468)

[**Приложения (Скриншоты работы и код программы)** 17](#_Toc75133469)

# **Введение**

Цель работы: написать программу, которая конвертирует валюту. Написать программу нужно на языке программирования Turbo Pascal с использованием библиотеки Turbo Vision.

Для перевода я выбрал родную валюту – рубли (RUB), а также американский доллар (USD) и ливанскую лиру (LBP).

Программа получает на входе первоначальную валюту и само число. Выводиться будет денежный эквивалент этой же валюты и двух других. В своей курсовой работе я использовал наследование и виртуальные методы.

# **Описание принципа работы программы**

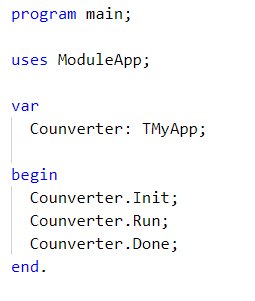
## Main (Converter.pas)

В основной программе компилятора я использую модуль **ModuleApp**, который описывается ниже.

Далее мы применяем инкапсуляцию (инкапсулируем все данные и функции-члены вместе, чтобы сформировать объект) полей и алгоритмов, которую применяем не только к графическим объектам, но и ко всей программе в целом.

Мы создаем объект-программу и даем ей 3 основных действия: инициации (**Init**), выполнению основной работы (**Run**) и завершению (**Done**).

* На этапе инициации (**Init**) экран переводится в графический режим работы и создаются, и отображаются графические объекты.
* На этапе **Run** осуществляется сканирование клавиатуры и перемещение графических объектов.
* На этапе **Done** экран переводится в текстовый режим и завершается работа всей программы.



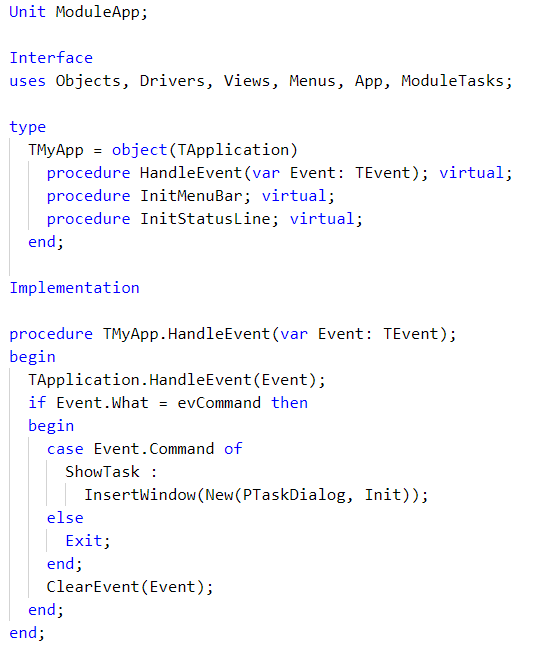
## ModuleApp (ModuleApp.pas)

Первый из двух модулей это **ModuleApp**. Он отвечает за создание и инициализацию нашего окна и добавление кнопок меню.

Окно Turbo Vision – это объект со встроенной в него возможностью реагировать на ввод пользователя без написания специального кода. Окна Turbo Vision уже знают, как открываться, изменять размер, перемещаться и закрываться. Окно Turbo Vision содержит то, что содержат и чем управляют другие объекты: эти объекты отображают себя на экране. Окно управляет видимыми элементами и функциями программы – это видимые элементы, которые окно содержит, и которыми оно управляет.

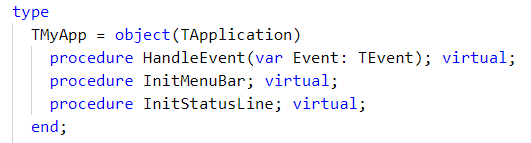
**Библиотеки**

Подключаем библиотеки дополнительных процедур и функций для управления экраном: **Objects**, **Drivers**, **Views**, **Menus**, **App**, **ModuleTasks**.

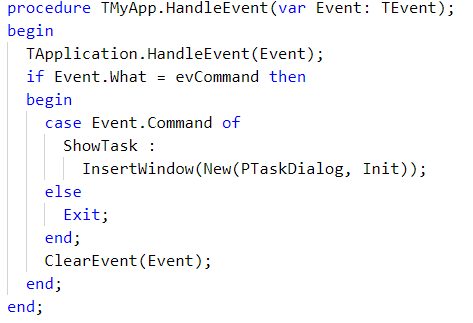


**Инициализация окна**

В наших процедурах мы будем использовать виртуальные методы (**virtual**). Виртуальный метод — это метод, чей вызов встроен в его код во время выполнения, процессом, называемым процессом поздней связи. Объявление метода виртуальным делает возможным для методов с тем же именем выполнение различными способами внутри иерархии объектных типов.

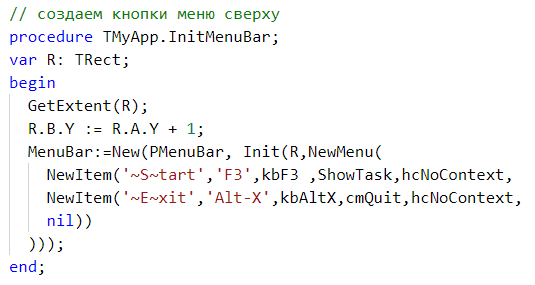


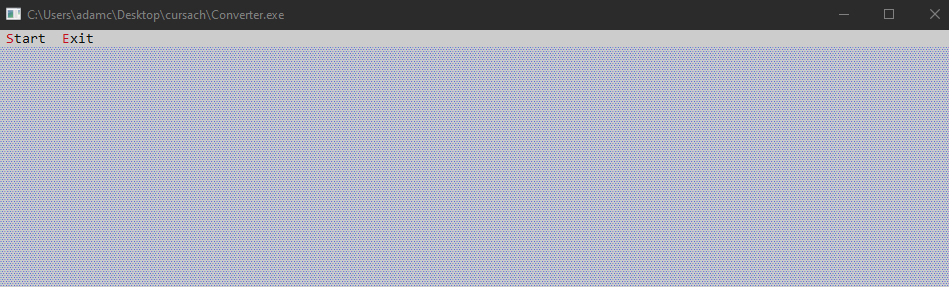
Первая процедура **TMyApp.HandleEvent** ответственна за создание окна. **TApplication.HandleEvent(Event)** в данном случае действует как предок, но откликается на дополнительные команды. **ClearEvent(Event)** очищает событие после обработки. Данный код инициализирует окно и подсоединяет его к панели экрана.

****

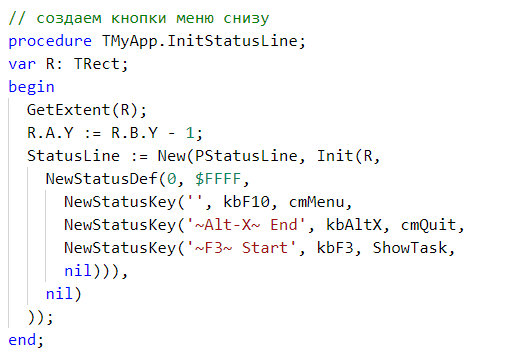
**Кнопки меню**

Добавим кнопки меню сверху. Для этого заведем другую процедуру **TMyApp.InitMenuBar**. Вверху окна мы создаем две кнопки: **Start** и **Exit**. Так же мы привязываем к ним кнопку «**F3**» и комбинацию клавиш «**Alt+X**» соответственно. Теперь при нажатии на кнопки или на клавиши у нас будет запускаться и закрываться наша программа.





Аналогично создаем такие же кнопки меню снизу. Делаем это с помощью процедуры **procedure TMyApp.InitStatusLine**.





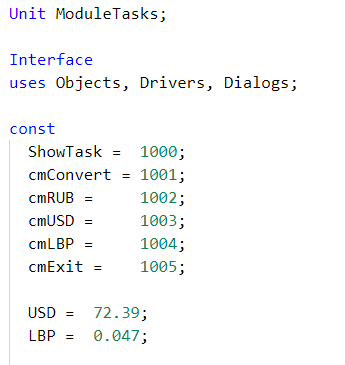
В результате у нас получилось инициализировать окно с кнопками (После того как мы зададим размеры окна во втором модуле **ModuleTasks**.

## ModuleTasks (ModuleTasks.pas)

Второй модуль **ModuleTasks** отвечает за размер окон, их положение, а также за расчет и вывод валют.

**Библиотеки и константы**

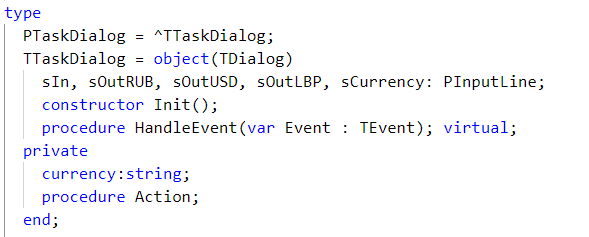
Вначале подключаем 3 библиотеки: **Objects**, **Drivers** и **Dialogs**. Задаем команды для обработчиков событий: ShowTask, cmConvert, cmRUB, cmUSD, cmLBP, cmExit. А также указываем курсы валют.



**Наследование**

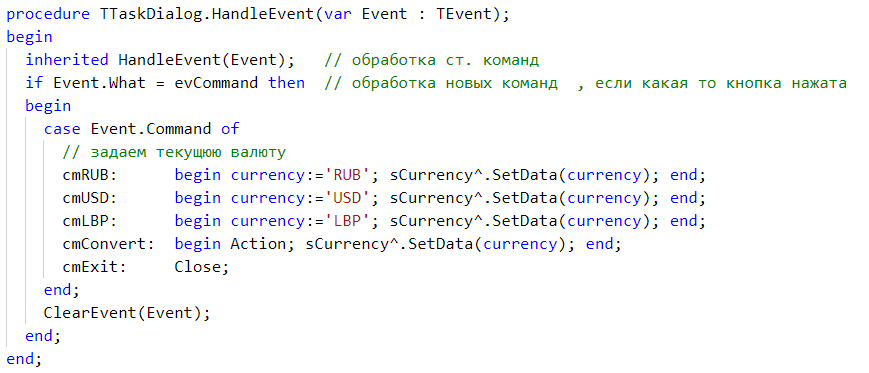
Если методы сделать виртуальными и инкапсулировать в объект-предок, каждый из его потомков сможет осуществить нужные действия одинаковым способом. Поэтому мы будем пользоваться предками.

Например, в нашем случае объект **TTaskDialog** наследуется от **TDialog**.



**Выбор валюты**

Процедура **procedure TTaskDialog.HandleEvent** отвечает за обработку команд и выбор валюты.



**Inherited** используется для того, чтобы обозначить непосредственного предка вложенного метода объектного типа. В данном случае мы обрабатываем старые команды.

Условие  обрабатывает новые команды, если нажата какая-либо кнопка. При нажатии кнопки (в зависимости на какую кнопку мы нажали) мы либо выбираем текущую валюту, либо конвертируем валюту, либо закрываем окно конвертатора.

При нажатии на кнопку какой-либо валюты, мы выбираем текущую валюту.

При нажатии на кнопку **Convert**, мы переводим из данной валюты в остальные.

При нажатии на кнопку **Close**,мызакрываем окно конвертатора.

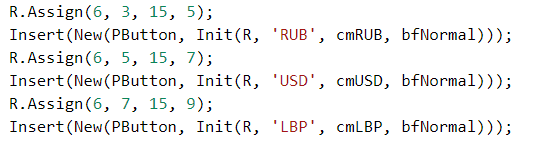
**Конструктор**

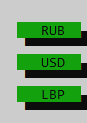
Далее мы вводим конструктор **constructor TTaskDialog.Init**. Конструктор – это специальный метод, который инициализирует объект, содержащий виртуальные методы. Тут мы будем описывать размеры и положение кнопок и окон.

Вначале мы вводим размер основного окна конвертера и пишем наверху «Converter». (Первые две цифры – координаты верхней левой точки, последние две цифры – координаты нижней правой)

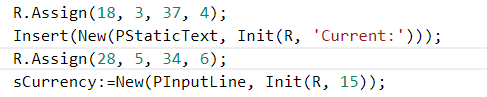


Ниже описываем положение и размеры трех кнопок выбора валют.



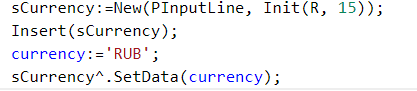


Далее вводим положение фразы «Current». Так же указываем координаты строки, в которой написана данная валюта (из которой мы переводим).

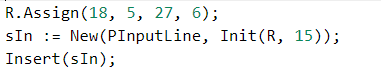




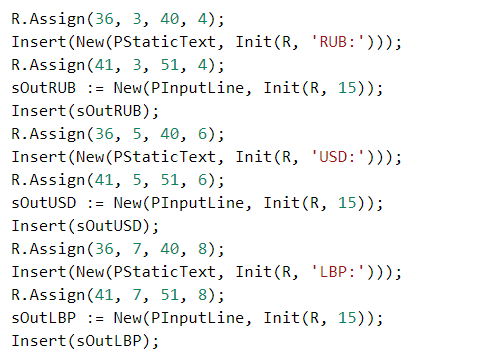
Пусть первоначальная валюта всегда будет равна рублям.

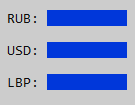


Потом идут координаты строки ввода.

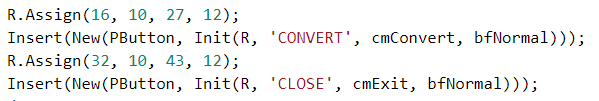


Ниже описаны окна, в которые программа будет передавать значения переведенных валют.



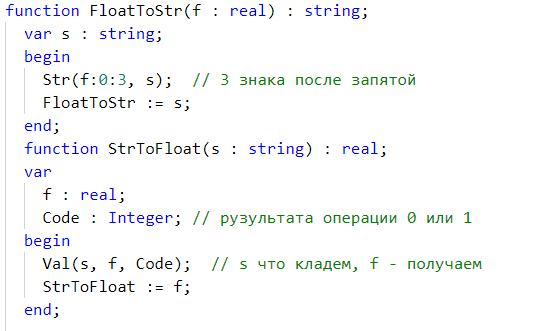


Аналогично с кнопками «**Close**» и «**Convert**», при нажатии на которые будет запускаться событие описанные выше.



**Перевод типов данных**

Описываем процедуру **procedure TTaskDialog.Action**. В ней находится две функции: **function FloatToStr** переводит число в строку, а **function StrToFloat** переводит строку в число. Нам нужны эти функции, чтобы взаимодействовать с данными, потому что, когда мы вписываем в диалоговое окно число для перевода, оно считается не как число, а как строка. Поэтому для того, чтобы выполнить какие-либо математические действия нам надо перевести эту строку в число. А когда мы все сделали, надо перевести его обратно в строку, чтобы вывести на экран. В функции **function FloatToStr** мы так же указываем количество знаков после запятой (в данный момент 3 знака после запятой).

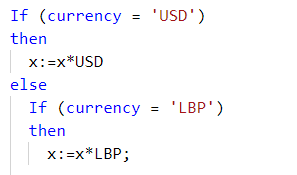


**Перевод валют**

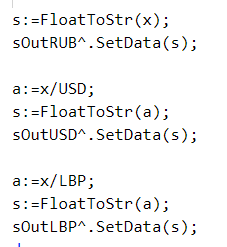
Далее идет тело программы. Мы получаем на вход строку для перевода и переводим его в число.



Так как мы считаем все ориентируясь на рубли, то сначала определяем валюту, а потом переводим все в рубли.



Потом мы выводим рубли и считаем другие валюты. Делим рубли на курс и получаем доллары и ливанскую лиру.



# **Заключение**

В своей курсовой работе я решил все поставленные перед собой задачи. Мною был создан простой в оформлении, но очень эффективный конвертер валют. Он имеет такие функции, как выбор валюты, ввод числа, кнопка конвертации и кнопка закрытия окна. С помощью библиотеки Turbo Vision я смог реализовать все методы, которые хотел.

# **Литература:**

1. Фаронов В.В.Ф24 Турбо Паскаль 7.0. Начальный курс. Учебное пособие. -М.: Издательство «ОМД Групп», 2003. -616 с

# **Ссылки на источники:**

1. http://citforum.ru/programming/tv\_doc/

2. http://www.pascal.helpov.net/index/

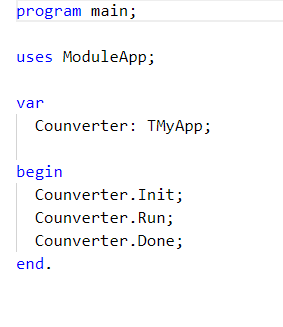
3. https://ru.wikipedia.org/wiki/Turbo\_Pascal

4. http://mif.vspu.ru/books/pascal/

# **Приложения (Скриншоты работы и код программы)**

****

**Converter.pas**

****

**ModuleApps.pas**

Unit ModuleApp;

Interface

uses Objects, Drivers, Views, Menus, App, ModuleTasks;

type

  TMyApp = object(TApplication)

    procedure HandleEvent(var Event: TEvent); virtual;

    procedure InitMenuBar; virtual;

    procedure InitStatusLine; virtual;

  end;

Implementation

procedure TMyApp.HandleEvent(var Event: TEvent);

begin

  TApplication.HandleEvent(Event);

  if Event.What = evCommand then

  begin

    case Event.Command of

      ShowTask :

        InsertWindow(New(PTaskDialog, Init));

    else

      Exit;

    end;

    ClearEvent(Event);

  end;

end;

procedure TMyApp.InitMenuBar;

var R: TRect;

begin

  GetExtent(R);

  R.B.Y := R.A.Y + 1;

  MenuBar:=New(PMenuBar, Init(R,NewMenu(

    NewItem('~S~tart','F3',kbF3 ,ShowTask,hcNoContext,

    NewItem('~E~xit','Alt-X',kbAltX,cmQuit,hcNoContext,

    nil))

  )));

end;

procedure TMyApp.InitStatusLine;

var R: TRect;

begin

  GetExtent(R);

  R.A.Y := R.B.Y - 1;

  StatusLine := New(PStatusLine, Init(R,

    NewStatusDef(0, $FFFF,

      NewStatusKey('', kbF10, cmMenu,

      NewStatusKey('~Alt-X~ End', kbAltX, cmQuit,

      NewStatusKey('~F3~ Start', kbF3, ShowTask,

      nil))),

    nil)

  ));

end;

end.

**ModuleTasks.pas**

Unit ModuleTasks;

Interface

uses Objects, Drivers, Dialogs;

const

  ShowTask =  1000;

  cmConvert = 1001;

  cmRUB =     1002;

  cmUSD =     1003;

  cmLBP =     1004;

  cmExit =    1005;

  USD =  72.39;

  LBP =  0.047;

type

  PTaskDialog = ^TTaskDialog;

  TTaskDialog = object(TDialog)

    sIn, sOutRUB, sOutUSD, sOutLBP, sCurrency: PInputLine;

    constructor Init();

    procedure HandleEvent(var Event : TEvent); virtual;

  private

    currency:string;

    procedure Action;

  end;

Implementation

procedure TTaskDialog.HandleEvent(var Event : TEvent);

begin

  inherited HandleEvent(Event);

  if Event.What = evCommand then

  begin

    case Event.Command of

      cmRUB:      begin currency:='RUB'; sCurrency^.SetData(currency); end;

      cmUSD:      begin currency:='USD'; sCurrency^.SetData(currency); end;

      cmLBP:      begin currency:='LBP'; sCurrency^.SetData(currency); end;

      cmConvert:  begin Action; sCurrency^.SetData(currency); end;

      cmExit:     Close;

    end;

    ClearEvent(Event);

  end;

end;

constructor TTaskDialog.Init;

var

  R : TRect;

begin

  R.Assign(15, 5, 73, 18);

  inherited Init(R, 'CONVERTER');

  R.Assign(6, 3, 15, 5);

  Insert(New(PButton, Init(R, 'RUB', cmRUB, bfNormal)));

  R.Assign(6, 5, 15, 7);

  Insert(New(PButton, Init(R, 'USD', cmUSD, bfNormal)));

  R.Assign(6, 7, 15, 9);

  Insert(New(PButton, Init(R, 'LBP', cmLBP, bfNormal)));

  R.Assign(18, 3, 37, 4);

  Insert(New(PStaticText, Init(R, 'Current:')));

  R.Assign(28, 5, 34, 6);

  sCurrency:=New(PInputLine, Init(R, 15));

  Insert(sCurrency);

  currency:='RUB';

  sCurrency^.SetData(currency);

  R.Assign(18, 5, 27, 6);

  sIn := New(PInputLine, Init(R, 15));

  Insert(sIn);

  R.Assign(36, 3, 40, 4);

  Insert(New(PStaticText, Init(R, 'RUB:')));

  R.Assign(41, 3, 51, 4);

  sOutRUB := New(PInputLine, Init(R, 15));

  Insert(sOutRUB);

  R.Assign(36, 5, 40, 6);

  Insert(New(PStaticText, Init(R, 'USD:')));

  R.Assign(41, 5, 51, 6);

  sOutUSD := New(PInputLine, Init(R, 15));

  Insert(sOutUSD);

  R.Assign(36, 7, 40, 8);

  Insert(New(PStaticText, Init(R, 'LBP:')));

  R.Assign(41, 7, 51, 8);

  sOutLBP := New(PInputLine, Init(R, 15));

  Insert(sOutLBP);

  R.Assign(16, 10, 27, 12);

  Insert(New(PButton, Init(R, 'CONVERT', cmConvert, bfNormal)));

  R.Assign(32, 10, 43, 12);

  Insert(New(PButton, Init(R, 'CLOSE', cmExit, bfNormal)));

end;

procedure TTaskDialog.Action;

function FloatToStr(f : real) : string;

  var s : string;

  begin

    Str(f:0:3, s);

    FloatToStr := s;

  end;

  function StrToFloat(s : string) : real;

  var

    f : real;

    Code : Integer;

  begin

    Val(s, f, Code);

    StrToFloat := f;

  end;

var

  x,a:real;

  s:string;

begin

  sIn^.GetData(s);

  x:=StrToFloat(s);

  If (currency = 'USD')

  then

    x:=x\*USD

  else

    If (currency = 'LBP')

    then

      x:=x\*LBP;

  s:=FloatToStr(x);

  sOutRUB^.SetData(s);

  a:=x/USD;

  s:=FloatToStr(a);

  sOutUSD^.SetData(s);

  a:=x/LBP;

  s:=FloatToStr(a);

  sOutLBP^.SetData(s);

end;

end.