Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Основы алгоритмизации и программирования (ОАиП)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему

Программное средство анализа продаж товаров и построения прогнозов с выводом графиков

БГУИР КР 1-40 01 01 218 ПЗ

Студент: гр. 951002 Махнач А.О.

Руководитель:

асс. Фадеева Е.Е.

Минск 2020

# **СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 5](#_Toc42049521)

[1. АНАЛИЗ ПРОТОТИПОВ, ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОМУ ПРОГРАММНОМУ СРЕДСТВУ 6](#_Toc42049522)

[1.1 Анализ существующих программных средств 6](#_Toc42049523)

[1.2 Формирование требований к проектируемому программному средству 9](#_Toc42049524)

[2. АНАЛИЗ ТРЕБОВАНИЙ К ПС И РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ ТРЕБОВАНИЙ 11](#_Toc42049525)

[2.1 Описание используемых в курсовой работе методов анализа и прогнозирования 11](#_Toc42049526)

[2.2 Описание функциональности программного средства 15](#_Toc42049527)

[2.3 Спецификация функциональных требований 16](#_Toc42049528)

[3. ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА 19](#_Toc42049529)

[3.1 Обобщённый алгоритм работы программы 19](#_Toc42049530)

[3.2 Проектирование структур для данных в программе 19](#_Toc42049531)

[3.3 Проектирование общих структур данных для методов прогнозирования и анализа 20](#_Toc42049532)

[3.4 Алгоритм прогнозирования с построением графика 20](#_Toc42049533)

[3.5 Алгоритм построения графиков 21](#_Toc42049534)

[3.6 Алгоритмы прогнозирования и анализа 22](#_Toc42049535)

[4 СОЗДАНИЕ (КОНСТРУИРОВАНИЕ) ПРОГРАММНОГО СРЕДСТВА 28](#_Toc42049536)

[4.1 Подпрограммы модуля MainUnit 28](#_Toc42049537)

[4.2 Подпрограммы модуля AddForm 35](#_Toc42049538)

[4.3 Подпрограммы модуля Graphic 37](#_Toc42049539)

[5. ТЕСТИРОВАНИЕ, ПРОВЕРКА РАБОТОСПОСОБНОСТИ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ 40](#_Toc42049540)

[6. РУКОВОДСТВО ПО УСТАНОВКЕ И ИСПОЛЬЗОВАНИЮ 48](#_Toc42049541)

[ЗАКЛЮЧЕНИЕ 57](#_Toc42049542)

[СПИСОК ЛИТЕРАТУРЫ 58](#_Toc42049543)

[ПРИЛОЖЕНИЕ 1 59](#_Toc42049544)

[ПРИЛОЖЕНИЕ 2 60](#_Toc42049545)

# **ВВЕДЕНИЕ**

Анализ продаж является важной составляющей в области торговли и экономики в целом как для бизнеса различных категорий и размеров, так и для государственных структур (например, для учёта возможных изменений в бюджете из-за изменения уровня продаж, опять же, предприятий). Он позволяет определять тенденции рынка и корректность текущей экономической политики, грамотно распределять новые ресурсы и перераспределять имеющиеся, что несомненно благоприятно влияет на дальнейшее развитие предприятия.

Прогнозы, построенные различными методами на основе данных, полученных после проведения анализа продаж, позволяют упростить принятие управленческих решений за счёт расчёта наиболее вероятного движения продаж в дальнейшем. Для большей наглядности и доступности нередко прогнозирование и анализ данных подкрепляют графиками, что позволяет использовать результаты также вне специализированных рамок.

Целью работы является создание программного средства, которое позволит на основе имеющихся данных проводить определённого рода анализ, а также строить графики с элементами прогнозирования.

В дальнейших разделах будут рассмотрены преимущества и недостатки каждого из методов. Анализ же будет проводиться в виде поиска наиболее и наименее популярных представителей в определённых категориях данных.

Данная пояснительная записка содержит следующие основные разделы. В первом разделе выполнен анализ прототипов, литературных источников. Во втором разделе сформированы функциональные требования к проектируемому программному средству. Третий раздел посвящен проектированию программного средства и содержит схемы работы модулей программного средства. Четвертый раздел описывает этапы создания программного средства. Пятый раздел содержит набор тестов и сценарии тестирования. В шестом разделе описано руководство по установке и использованию. Завершающий раздел содержит собранные статистические данные работы анализатора при различных условиях и выводы.

# **АНАЛИЗ ПРОТОТИПОВ, ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОМУ ПРОГРАММНОМУ СРЕДСТВУ**

# **Анализ существующих программных средств**

В зависимости от масштабности и направленности предприятия могут применяться различные средства анализа продаж, построения графиков с прогнозами, однако зачастую они не направлены исключительно на единственную конкретную задачу, а являются совмещёнными. Многие средства в данной области имеют схожую структуру: данные представляются в виде таблиц или списков, предусмотрены возможности редактирования, добавления и удаления данных, а для их анализа, а также построения графиков и прогнозов представляется ряд функций. Некоторые средства, будучи более широко направленными (например, Microsoft Excel), предоставляют пользователю большую свободу действий, другие выполняют строго установленные, необходимые конкретному типу пользователей, задачи.

* + 1. **Microsoft Excel**

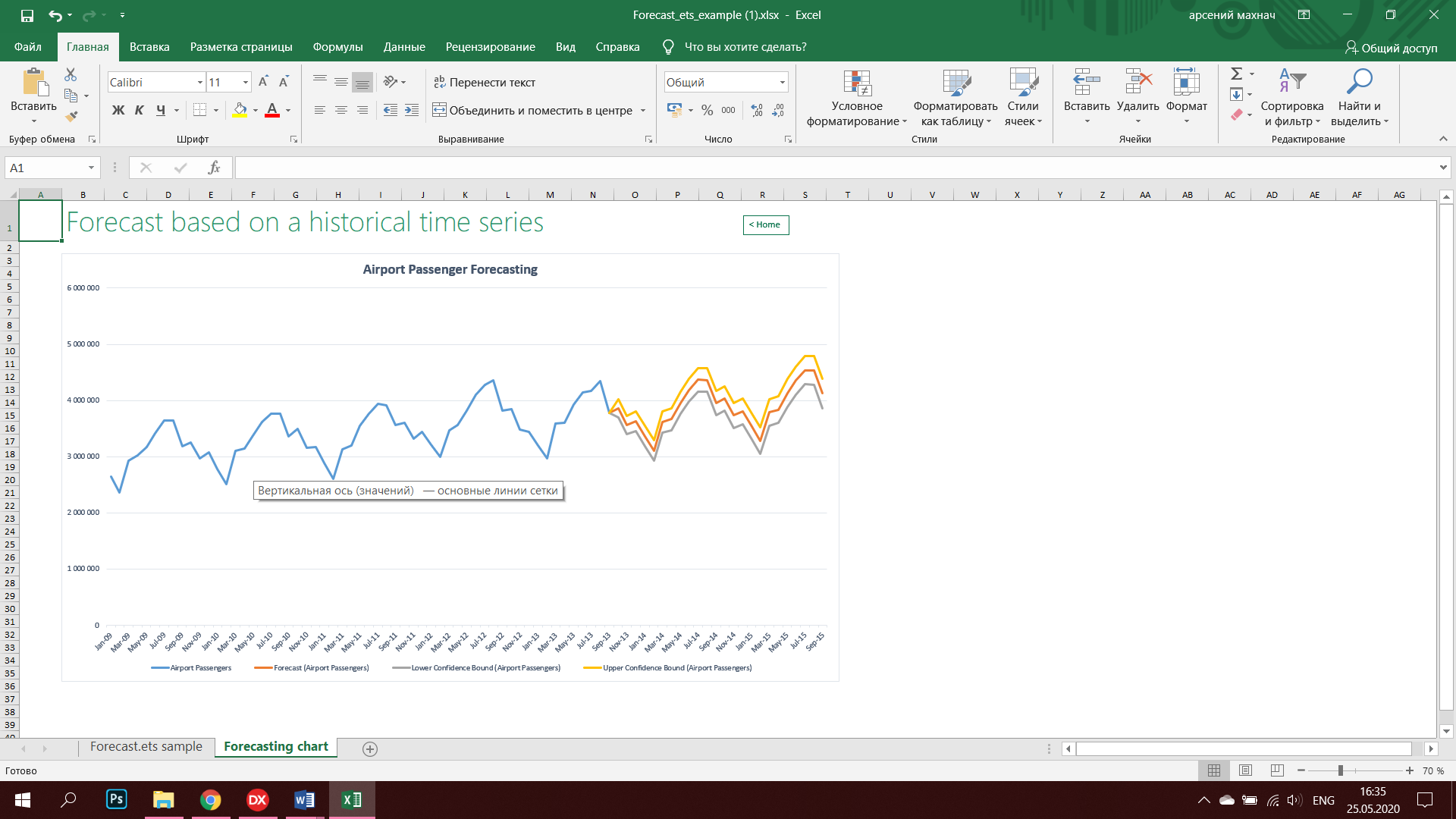
 Программных инструментов для планирования продаж в последнее время появилось множество — при этом большинство из них либо дороги для малого бизнеса, либо требуют значительных затрат на оплату рабочего времени специалистов. Приложение Microsoft Excel позволяет достаточно быстро получить прогноз как по продажам только что запускаемого товара, так и по уже известным продуктам.

Рис. 1.1 – Интерфейс Microsoft Excel

В Excel могут быть реализованы следующие методы для прогнозирования:

* наименьших квадратов;
* скользящей средней;
* экспоненциального сглаживания;
* сезонных колебаний.

Благодаря простоте работы Excel и большому количеству полезных встроенных функций и процедур, он позволяет решать многие простые прикладные задачи, связанные с обработкой данных. Однако потребность в серьезных методах прикладной статистики и анализа данных у пользователей MS Excel при этом остается неудовлетворенной [1].

* + 1. **Statistica**

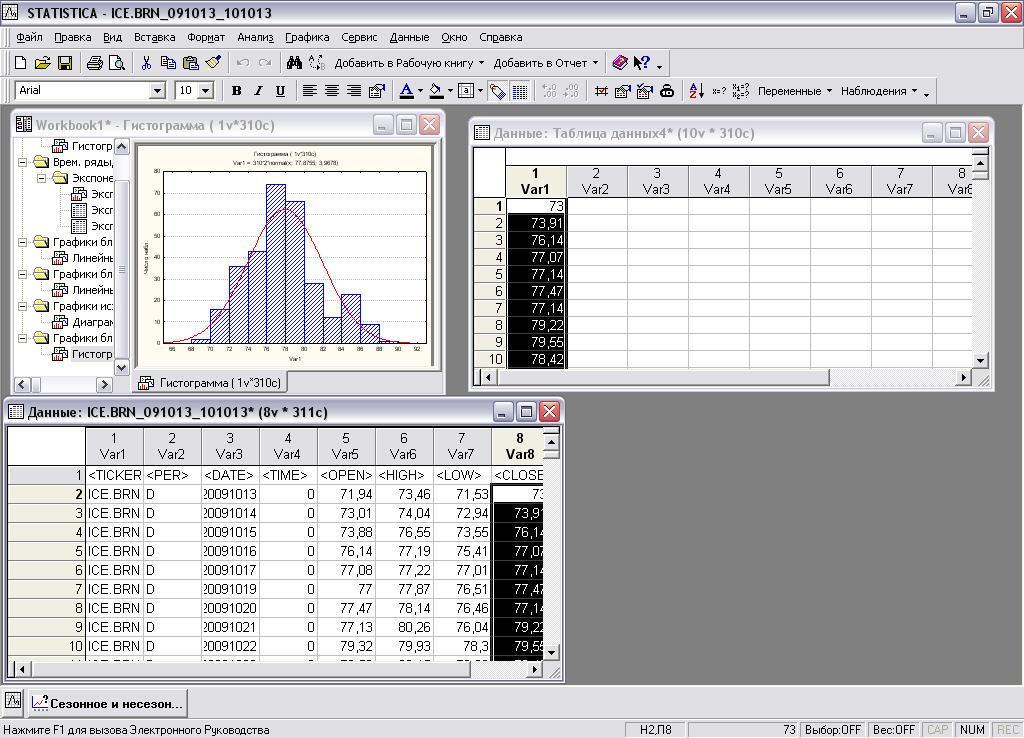
 Statistica — это интегрированная система для анализа данных, графики и управления данными, инструмент разработки пользовательских приложений в бизнесе, экономике, финансах. Statistica легка в освоении и использовании. Все имеющиеся в системе аналитические инструменты доступны пользователю и могут быть выбраны с помощью альтернативного пользовательского интерфейса.

Рис. 1.2 – Интерфейс Statistica

В пакете Statistica реализованы следующие методы прогнозирования продаж:

* регрессионный анализ;
* ARIMA;
* экспоненциальное сглаживание;
* спектральный анализ;
* сезонная декомпозиция;
* нейронные сети;

Процедуры Statistica имеют высокую скорость и точность вычислений. Гибкая и мощная технология доступа к данным позволяет эффективно работать как с таблицами данных на локальном диске, так и с удаленными хранилищами данных. Statistica поддерживает Web-форматы: HTML, JPEG, PNG [1].

* + 1. **ForexSal**

ForexSal является специализированным программным обеспечением для прогнозирования продаж. Уникальные возможности программы обусловлены тем, что в ней реализованы методы прогнозирования временных рядов, предназначенные как для построения прогнозов отдельных независимых товаров, так и одновременно для многих товаров с учетом взаимного воздействия товаров друг на друга и влияния внешних факторов. Прогнозирование в программе осуществляется классическими методами построения прогнозов. В программе реализованы трендовые и сезонные методы.

Трендовые методы:

* ARIMA;
* скользящего среднего;
* Хольта — Уинтерса;
* экспоненциального сглаживания.

Для каждого ряда (статистики продаж отдельного товара) возможно нахождение оптимального метода построения прогноза, который дает наиболее точные оценки прогноза. Разработаны специализированные методы совместного прогнозирования многих временных рядов (товаров и факторов), например, продажи товара, цены и рекламные затраты.

Для сезонных продаж реализован метод построения прогноза для товаров, подверженных сезонному спросу (метод Хольта — Уинтерса). В программе реализовано девять методов прогнозирования для сезонных рядов с различными особенностями поведения: различные типы трендов (линейные, квадратичные, кубические), различные виды динамики амплитуд периодов: сходящиеся амплитуды, расходящиеся, равнозначные периоды и т. д.

На вход программы подаются данные о продажах и поведении внешних факторов (если таковые необходимо анализировать). Возможен ручной ввод с клавиатуры или в режиме рисования кривой поведения ряда [1].

* + 1. **Выводы**

Как можно установить из используемых в рассмотренных программных средствах методов анализа и прогнозирования, основными являются:

* метод экспоненциального сглаживания
* метод скользящего среднего
* метод регрессивного анализа

В более специальных ситуациях из-за недостатков каждого из методов, часто применяются иные методы, которые учитывают тренды в более совершенной форме, а также сезонность. Однако зачастую они основаны на одном из приведённых методов (напр. ARIMA), или же сразу на нескольких, совмещая их.

Прогнозирование в рассмотренных программах сопровождается возможностями построения графиков на основе данных, полученных в результате прогнозирования, а также графиков на основе исключительно имеющихся (не предсказанных) данных, что также является необходимой задачей, поставленной курсовой работой.

# **Формирование требований к проектируемому программному средству**

Согласно с темой курсовой работы, целью работы является создание программного средства, обеспечивающего возможности анализа продаж и построения прогнозов, сопровождаемых графиками. После рассмотрения существующих решений на данную тему я сформировал следующие требования к возможностям разрабатываемого программного средства:

* Отображение данных в программе в табличной (или близкой к ней) форме с разделением на поля;
* Работа с двумя различными массивами данных
* Возможность изменения данных (добавление, редактирование, удаление) непосредственно в программе;
* Возможность сохранения данных в файлы форматов .table и .txt и импорта данных из файлов формата .table в программу (при условии их корректности);
* Возможности сортировки данных, а также поиска конкретных данных среди имеющихся;
* Примитивные возможности анализа (основанные на функциях поиска);
* Наличие меню, представляющего доступ к большинству функций программы
* Возможности анализа и прогнозирования с возможностью выбора различных методов и данных, на основе которых анализ и прогнозирование производится, а также графическое представление результатов в виде графика
* Возможность сохранения графика, построенного в программе в файлы форматов .jpg и .bmp;
* Обработка ошибок, связанных с вводом данных, с выводом соответствующих сообщений, а также вывод сообщений, предупреждающих определённые действия (например, выход).

В качестве методов анализа и прогнозирования я буду использовать метод простого экспоненциального сглаживания, метод наименьших квадратов и метод простого скользящего среднего, как представления экспоненциального, регрессивного и скользящего среднего методов соответственно (как наиболее базовых).

В качестве языка программирования для реализации поставленной задачи предложено использовать Delphi.

# **АНАЛИЗ ТРЕБОВАНИЙ К ПС И РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ ТРЕБОВАНИЙ**

# **Описание используемых в курсовой работе методов анализа и прогнозирования**

* + 1. **Метод простого экспоненциального сглаживания**

формула*Метод простого экспоненциального сглаживания* наиболее эффективен при разработке среднесрочных прогнозов. Он приемлем при прогнозировании только на один период вперед. Его основные достоинства простота процедуры вычислений и возможность учета весов исходной информации. Рабочая формула метода экспоненциального сглаживания:

(2.1)

В формуле (2.1) t – период, предшествующий прогнозному; t+1 – прогнозный период; Ut+1 - прогнозируемый показатель; α - параметр сглаживания; Уt - фактическое значение исследуемого показателя за период, предшествующий прогнозному; Ut - экспоненциально взвешенная средняя для периода, предшествующего прогнозному[2]. При прогнозировании данным методом возникает два затруднения:

* выбор значения параметра сглаживания α;
* определение начального значения Uo.

От величины α зависит, как быстро снижается вес влияния предшествующих наблюдений. Чем больше α, тем меньше сказывается влияние предшествующих лет. Если значение α близко к единице, то это приводит к учету при прогнозе в основном влияния лишь последних наблюдений. Если значение α близко к нулю, то веса, по которым взвешиваются уровни временного ряда, убывают медленно, т.е. при прогнозе учитываются все (или почти все) прошлые наблюдения. Точного метода для выбора оптимальной величины параметра сглаживания α нет[2].

Задача выбора Uo (экспоненциально взвешенного среднего начального) решается следующими способами:

* если есть данные о развитии явления в прошлом, то можно воспользоваться средней арифметической и приравнять к ней Uo;
* если таких сведений нет, то в качестве Uo используют исходное первое значение базы прогноза У1.

Для демонстрации алгоритма данного метода я рассмотрю пример прогнозирования с его использованием. В качестве исходных данных я возьму данные, характеризующие ежемесячный уровень прибыли за промежуток с января по июнь. Соответственно прогноз будет выполняться на июль.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Январь | Февраль | Март | Апрель | Май | Июнь |
| 138 | 126 | 158 | 182 | 140 | 210 |

Таблица 2.1 – Исходные данные

Шаг 1. Выполняю выбор значения параметра сглаживания α.

Поскольку у меня не имеется данных для оправданного выбора конкретного значения, в данном случае я рассмотрю α = 0.3. В случае разрабатываемого мной программного средства пользователю будет предоставлена возможность самостоятельного выбора.

Шаг 2. Определяю начальное значение Uo.

Так как у меня имеются данные за предшествующие прогнозному периоды, я нахожу Uo как среднее арифметическое фактических данных. Uo=(138+126+158+182+140+210)/6=159

Шаг 3. Рассчитываю экспоненциально взвешенную среднюю для каждого периода по формуле.

Например, Uфев = 138 \*0,3 +(1-0,3) \* 159 = 152,7; Uмарт = 126\*0,3+(1-0,3) \* 152,7 = 144,69; Uапр = 158\*0,3+(1-0,3) \* 144,69 = 148,68 и т.д. В конечном итоге получаю следующие данные:

Таблица 2.2 – Исходные данные совместно с данными по экспоненциально взвешенным средним

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Январь | Февраль | Март | Апрель | Май | Июнь |
| Фактическое значение | 138 | 126 | 158 | 182 | 140 | 210 |
| Экспонециально взвешенное | 159 | 152,7 | 144,69 | 148,68 | 158,68 | 153,08 |

Шаг 4. Рассчитываю значение экспоненциально взвешенной средней для прогнозного периода.

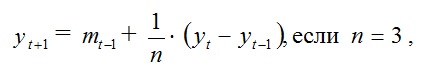
Uиюль=210\*0.3+0.7\*153.08=170,16. Это значение и является прогнозом на июль согласно методу простого экспоненциального сглаживания.

* + 1. **Метод простого скользящего среднего**

*Метод скользящих средних* является одним из широко известных методов сглаживания временных рядов. Применяя этот метод, можно элиминировать случайные колебания и получить значения, соответствующие влиянию главных факторов.

Сглаживание с помощью скользящих средних основано на том, что в средних величинах взаимно погашаются случайные отклонения. Это происходит вследствие замены первоначальных уровней временного ряда средней арифметической величиной внутри выбранного интервала времени. Полученное значение относится к середине выбранного интервала времени (периода). Затем период сдвигается на одно наблюдение, и расчет средней повторяется. При этом периоды определения средней берутся все время одинаковыми. Таким образом, в каждом рассматриваемом случае средняя центрирована, т.е. отнесена к серединной точке интервала сглаживания и представляет собой уровень для этой точки.

При сглаживании временного ряда скользящими средними в расчетах участвуют все уровни ряда. Чем шире интервал сглаживания, тем более плавным получается тренд. Сглаженный ряд короче первоначального на (n-1) наблюдений, где n – величина интервала сглаживания. При больших значениях n колеблемость сглаженного ряда значительно снижается. Одновременно заметно сокращается количество наблюдений, что создает трудности[3].

 Данный метод наиболее эффективен при краткосрочном прогнозировании. Также, в отличие от метода простого экспоненциального сглаживания, метод простого скользящего среднего не имеет ограничения на количество прогнозируемых периодов. Его рабочая формула:

(2.2)

В формуле (2.2) t + 1 – прогнозный период; t – период, предшествующий прогнозному периоду (год, месяц и т.д.); yt+1 – прогнозируемый показатель; mt-1 – скользящая средняя за два периода до прогнозного (при n=3); n – число уровней, входящих в интервал сглаживания; yt – фактическое значение исследуемого явления за предшествующий период; Уt-1 – фактическое значение исследуемого явления за два периода (при n=3), предшествующих прогнозному[3].

Для рассмотрения алгоритма данного метода я использую те же исходные данные и задачу, что и для метода простого экспоненциального сглаживания (таблица 2.3).

Таблица 2.3 – Исходные данные

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Январь | Февраль | Март | Апрель | Май | Июнь |
| 138 | 126 | 158 | 182 | 140 | 210 |

Шаг 1. Определяю величину интервала сглаживания n

Как пример, я возьму n=3 (в моём программном средстве будет использована эта же величина).

Шаг 2. Рассчитываю скользящую среднюю для последних n периодов

Расчёт производится путём нахождения среднего арифметического фактических значений элементов периода. Полученное значение считается скользящей средней для середины взятого периода. Т.е. в данном случае mмай=(182+140+210)/3=177,3. Данные после выполнения данного шага представлены в таблице 2.4.

Шаг 3. Рассчитываю прогнозное значение на искомый период по формуле

yиюль=177,3+(210-140)/3=154. Соответственно это и есть прогнозируемое значение на июль по методу скользящего среднего.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Январь | Февраль | Март | Апрель | Май | Июнь |
| Фактическое значение | 138 | 126 | 158 | 182 | 140 | 210 |
| Скользящая средняя | - | - | - | - | 177,3 | - |

Таблица 2.4 – Исходные данные и скользящая средняя

В случае, если прогноз выполняется на несколько периодов вперёд, Шаг 2 и Шаг 3 повторяются в соответствии с количеством периодов для прогноза. При этом значение, полученное прогнозом считается фактическим, т.е используется при расчётах скользящего среднего.

* + 1. **Метод наименьших квадратов**

Сущность *метода наименьших квадратов* состоит в минимизации суммы квадратических отклонений между наблюдаемыми и расчетными величинами. Расчетные величины находятся по подобранному уравнению – уравнению регрессии. Чем меньше расстояние между фактическими значениями и расчетными, тем более точен прогноз, построенный на основе уравнения регрессии.

*Рабочая формула метода наименьших квадратов*:

 У t+1 = а\*Х + b (2.3)

В формуле (2.3) t + 1 – прогнозный период; Уt+1 – прогнозируемый показатель; a и b - коэффициенты; Х - условное обозначение времени.

Расчет коэффициентов a и b осуществляется по следующим формулам:

(2.5)

(2.4)

|  |
| --- |
| формула |

|  |
| --- |
| формула |

В формулах (2.4) и (2.5) Уф – фактические значения ряда динамики; n – число уровней временного ряда; Х - условное обозначение времени[4].

Сглаживание временных рядов методом наименьших квадратов служит для отражения закономерности развития изучаемого явления. В аналитическом выражении тренда время рассматривается как независимая переменная, а уровни ряда выступают как функция этой независимой переменной. Развитие явления зависит не от того, сколько лет прошло с отправного момента, а от того, какие факторы влияли на его развитие, в каком направлении и с какой интенсивностью. Отсюда ясно, что развитие явления во времени выступает как результат действия этих факторов.

Недостатком метода наименьших квадратов является то, что при попытке описать изучаемое экономическое явление с помощью математического уравнения, прогноз будет точен для небольшого периода времени и уравнение регрессии следует пересчитывать по мере поступления новой информации[4].

Алгоритм прогнозирования методом наименьших квадратов я буду рассматривать на примере той же задачи, что и для предыдущих методов.

Таблица 2.4 - Исходные данные

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Январь (X=1) | Февраль (X=2) | Март (X=3) | Апрель (X=4) | Май (X=5) | Июнь (X=6) |
| 138 | 126 | 158 | 182 | 140 | 210 |

Шаг 1. Расчёт коэффициентов a и b по формулам

Для приведённого примера a=12,17; b=116,4.

Шаг 2. Определение прогнозного значения

yиюль=12,17\*7+116,4=201,59.

# **Описание функциональности программного средства**

Посредством выбора определённого пункта меню программы представляются различные возможности анализа имеющихся данных. При выборе пункта меню «Графики» (нажатии на него) будет открыто окно, которое предназначено для построения графиков с прогнозом. Выбор данного пункта недоступен при несоблюдении определённых условий (вызывающих невозможность построения корректного графика). Построение графика осуществляется при нажатии на кнопку «Построить». В данном окне имеются возможности настройки графика перед его построением, а именно: выбор анализируемых данных и метода анализа и прогнозирования посредством выбора пункта из комбинированного списка; в случае, если выбран метод экспоненциального сглаживания, изменение параметра сглаживания. После построения графика доступно его сохранение в файл формата .jpg или .bmp, сохранение выполняется посредством нажатия на кнопку «Сохранить график».

Также во вкладке «Анализ» меню представляется вариативность более примитивных возможностей анализа, таких как идентификация лучшего и худшего наименования среди имеющихся данных (по данным о прибыли по данному товару или по данным о количестве продаж по данному товару), идентификация наиболее популярного производителя (исходя из данных о том, каково количество наименований с определённым производителем), идентификация лучших товаров по производителям (исходя из данных о количестве продаж или прибыли определённых товаров у определённых производителей), расчёт общей прибыли, трат и итогов за имеющийся период, обнаружение лучшей даты в периоде (исходя из данных о прибыли, тратах или итогах в конкретный день\месяц). О результатах приведённых действий пользователь информируется посредством всплывающих окон-сообщений.

# **Спецификация функциональных требований**

* Отображение данных в программе в табличной (или близкой к ней) форме с разделением на поля и работа с двумя различными массивами данных;

Программа представляет доступ к двум различным массивам данных, данные одного из которых отображаются в виде списка в программе. При открытии программы отображается список, демонстрирующий данные из массива данных о товарах, который изначально пуст. При нажатии на кнопку «таблица итогов» отображение переключается на массив данных о периодах.

В случае отображения массива с данными о товарах отображается следующая информация, разделённая на колонки с соответствующими заголовками: код, наименование, производитель, количество, закупочная цена, НДС, цена с НДС, цена продажи, количество продаж, прибыль, а также номер позиции в списке. В случае отображения массива с данными о периодах будут отображаться следующая информация, разделённая на заголовки с соответствующими названиями: дата(формата дд.мм.гггг), прибыль, траты и итог.

* Возможность изменения данных (добавление, редактирование, удаление) непосредственно в программе;

При любом из вариантов отображения реализованы функции добавления, изменения, удаления и очистки, осуществляемые при нажатии кнопок «Добавить», «Изменить», «Удалить» и «Очистить список» соответственно.

Добавление и изменение данных реализовано в отдельном окне посредством ввода данных в предназначенные для этого компоненты или выбора из определённого списка значений (например, выбор даты в календаре). При желании отмены выполнения добавления или изменения предусмотрена кнопка «Отменить», которая закроет окно без изменений в массиве данных.

В случае с отображением данных о товарах при нажатии на кнопку «Добавить» появится окно, в котором необходимо заполнить информацию о добавляемом товаре.

В случае с отображением данных об итогах в появившемся окне необходимо будет выбрать период (его начало и конец), после чего будут добавлены все даты от начальной точки периода до конечной. После этого при нажатии на кнопку «Добавить» будет происходить добавление следующего единичного периода относительно последнего на момент нажатия. Функции «Добавить» будут возвращена к исходному состоянию по очистке данных о периодах.

Функция удаления осуществляет удаление конкретной выбранной позиции, функция очистки производит удаление всех данных из массива данных, который сейчас отображается.

* Возможность сохранения данных в файлы форматов .table и .txt и импорта данных из файлов формата .table в программу (при условии их корректности);

Во вкладке меню «Файл» находятся пункты «Открыть», «Сохранить» и «Сохранить всё», отвечающие за импорт данных в программу и сохранение данных из программы.

При выборе пункта «Открыть» (горячая клавиша Ctrl+O) будет открыто диалоговое окно, где можно будет выбрать путь к файлу с расширением .table или ввести его вручную. Имеется обработка ошибок на ввод пути к несуществующему файлу и обработка случая, когда пользователь не ввел ничего.

При выборе пункта «Сохранить» (горячая клавиша Ctrl+S) будет открыто диалоговое окно, в котором необходимо ввести название сохраняемого файла и выбрать расширение (.table или .txt). Предусмотрена обработка случая, когда название не введено, а также вывод диалогового окна (с предупреждением о возможной перезаписи) в случае, если файл с введённым именем и расширением уже существует. В данном случае происходит сохранение того массива данных, отображение которого сейчас представлено в программе в виде списка. При выборе пункта «Сохранить всё» (горячая клавиша Shift+Ctrl+S) будет последовательно выполнено сохранение каждого из двух массивов данных. В случае, если сохраняемые массивы пусты (при любом из видов сохранения), будет выведено соответствующее сообщение, не допускающее сохранение.

* Возможности сортировки данных, а также поиска конкретных данных среди имеющихся;

В случае, если отображается массив данных о товарах, при нажатии на заголовок столбца списка (отображения) будет произведена сортировка данных массива по возрастанию в соответствии с нажатым заголовком (при нажатии на колонку с заголовком «Код» данные будут отсортированы относительно кодов товаров), после чего будет обновлено отображение. Если отображается массив данных о периодах, то при нажатии на заголовок столбца будет выведено сообщение о невозможности выполнения сортировки (такое решение принято из-за необходимости отсортированности данных по периодам для построения корректных прогнозов и соответственно графиков).

Поиск данных я реализую посредством прохода последовательно по элементам массива данных, выбор данных для поиска осуществляется через комбо бокс (выбор колонки) и поле ввода (непосредственно искомая информация). Поиск не строгий, т.е., чтобы элемент считался найденным необходимо частичное совпадение (например, поиск «7» найдёт и «78»). Для поиска выбран последовательный перебор, поскольку гарантии отсортированности массива перед поиском нет, а траты ресурсов на выполнение предварительной сортировки перед поиском не оправдано. После окончания поиска в зависимости от результата будет либо обновлено отображение (в соответствии с найденными элементами), либо, если искомый текст не найден, выведено соответствующее сообщение. Возврат отображения к полному осуществляется через кнопку «Х».

* Возможности анализа и прогнозирования.

Доступ к ним представляется посредством пункта меню «Графики» (и соответствующего открывающегося окна) и пунктов, содержащихся во вкладке меню «Анализ».

* Возможность сохранения графика, построенного в программе

Осуществляется посредством нажатия кнопки «Сохранить график» в специальном окне, после чего будет вызвано диалоговое окно сохранения, предоставляющее возможность выбора расширения (из .jpg и .bmp) и имени файла. Предусмотрена обработка перезаписи файла с выводом соответствующего сообщения.

* Предупреждающие сообщения

В случае попытки закрытия приложения (кнопкой на рамке основного окна или кнопкой «Выход» меню) будет выведен message box, уточняющий, действительно ли необходимо выполнить закрытие, и предупреждающий о возможной потере данных. В случае попытки закрыть окно добавления\изменения данных (кнопкой на рамке окна или кнопкой «Отменить») будет выведен message box, уточняющий, действительно ли необходимо выполнить закрытие.

Также предусмотрено сообщение, не допускающее построения графика при превышении количества позиций в массиве данных о периодах значения в 366 (с целью корректного отображения и построения графика).

# **ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА**

# **Обобщённый алгоритм работы программы**

Алгоритм работы программы следующий: изначально создаётся главная форма, после чего начинается взаимодействие пользователя с интерфейсом программы, которое, в свою очередь, после обработки вызывает выполнение определённых подпрограмм. В результате выполнения могут быть созданы (или показаны) другие формы, у которых также есть элементы взаимодействия с пользователем. Обобщённая схема алгоритма программы представлена в приложении 1.

# **Проектирование структур для данных в программе**

Таблица 3.1 – Типы структур данных, использованные в программе

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Назначение** | **Тип** | **Состав** |
| TReportPosRec | Хранение данных о единичной позиции товара, а именно её код, наименование, производитель, количество, цена, НДС, цена с НДС, цена продажи, количество продаж и прибыль. | Запись | Code: string[30] |
| Name: string[50] |
| Prod: string[50] |
| Amount: integer |
| Price: real |
| VAT: integer |
| VatPrice: real |
| PriceSold: real |
| AmountSold:integer |
| Income: real |
| TReportPos | Хранение данных о перечне товаров | Динамический массив из записей TReportPosRec | Array of TReportPosRec |
| TReportRec | Хранение данных о единичном отрезке периода (о дате), содержит информацию о дате отрезка, прибыли, тратах и итогах в данную дату. | Запись | Date: TDate |
| Income: real |
| Expense: real |
| Total: real |

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Назначение** | **Тип** | **Состав** |
| TReport | Хранение данных о периоде (перечне его отрезков) | Динамический массив из записей TReportPos | Array of TReportPos |
| TFilePos | Считывание данных о товарах из файла в программу и запись из программы в файл | Файл из записей TReportPosRec | File of TReportPosRec |
| TFileTotal | Считывание данных о периодах из файла в программу и запись из программы в файл | Файл из записей TReportPosRec | File of TReportPosRec |

Продолжение таблицы 3.1

Для хранения данных я выбрал динамические массивы, поскольку их элементы индексированы, что позволяет осуществлять достаточно эффективные алгоритмы сортировки (в программе используется быстрая сортировка), а также в целом упрощает работу с данными (изменение элемента по индексу, например).

# **Проектирование общих структур данных для методов прогнозирования и анализа**

Таблица 3.2 – Типы общих структур данных для методов прогнозирования и анализа

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Назначение** | **Тип** | **Состав** |
| TDots | Хранение данных для построения графика (будет содержать в том числе и данные о прогнозе) | Динамический массив с данными типа real | Array of real |

# **Алгоритм прогнозирования с построением графика**

Посредством компонентов комбо бокс происходит выбор пользователем метода прогнозирования и анализа и анализируемые данные. В зависимости от выбора выполняются процедуры, определяющие данные для построения графика, включая данные прогноза. Затем происходит поиск наибольшего и наименьшего из значения среди этих данных, чтобы при построении графика на основе этого выполнить масштабирование. Затем происходит непосредственно построение графика. Схема прогнозирования с построением графика представлена на рисунке 3.1.

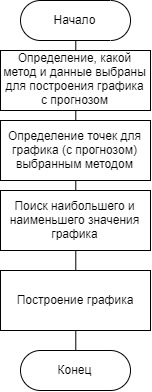


Рис. 3.1 – Схема алгоритма прогнозирования с построением графика

# **Алгоритм построения графиков**

Рисование графика происходит на канве изображения. Перед рисованием, определяется, какой шаг будет между точками относительно координаты x, на основании ширины компонента-изображения и количества точек, которые необходимо построить. Затем, поскольку количество пикселей на изображении ограничено, необходимо масштабировать данные перед рисованием. С этой целью вводится коэффициент масштабирования, который зависит от минимального и максимального значений данных для построения, а также высоты изображения. Далее выполняется отрисовка оси Y, а также на оси отмечается максимальное и минимальное значение данных для построения, после выполняется отрисовка графика. На рисунке 3.2 представлена схема алгоритма построения графиков.

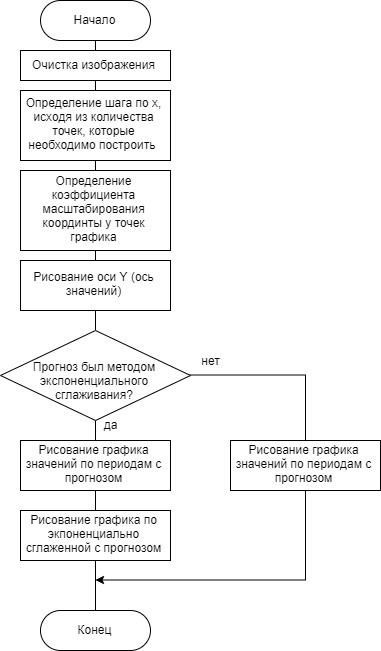


Рис. 3.2 – Схема алгоритма построения графиков

# **Алгоритмы прогнозирования и анализа**

* + 1. **Алгоритм работы метода экспоненциального сглаживания**

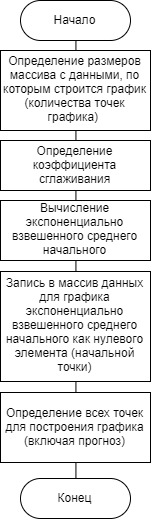
В первую очередь определяется, какого размера будет массив, хранящий данные экспоненциального сглаживания (это же в последующем количество точек графика). Поскольку простое экспоненциальное сглаживание допускает прогноз только на один период, то определяемое значение – это количество анализируемых периодов + 1. Далее на основе выбранного пользователем значения компонента трэк бар определяется коэффициент сглаживания. Затем вычисляется экспоненциально взвешенное среднее начальное на основе анализируемых данных (как среднее арифметическое из них), после чего определяются все значения полученные путём экспоненциального сглаживания и значение на прогнозный период. Более подробное непосредственно теоретическое описание алгоритма представлено в пункте 2.1.1 раздела 2. На рисунке 3.3 представлена схема алгоритма прогнозирования и анализа методом простого экспоненциального сглаживания.

Рис. 3.3 – Схема алгоритма прогнозирования методом простого экспоненциального сглаживания

* + 1. **Алгоритм работы метода простого скользящего среднего**

В таблице 3.3 представлена структура, которая используется при прогнозировании методом простого скользящего среднего (не считая общей структуры для всех методов прогнозирования).

Таблица 3.3 – Структура, использованная вметоде простого скользящего среднего

|  |  |  |
| --- | --- | --- |
| **Название** | **Назначение** | **Тип** |
| AvgArray | Массив значений скользящих средних | Array [0..6] of real |

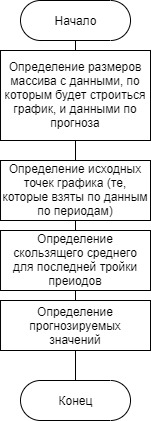
В первую очередь определяется, какого размера будет массив, хранящий исходные (в том числе анализируемые) и прогнозируемые данные (это же в последующем точки графика). Далее, исходя из имеющихся данных о периодах и выбранных ранее данных для анализа, заполняется часть этого массива (исходные данные). Затем определяется значение скользящей средней для последней тройки исходных периодов, после чего выполняется вычисление прогнозируемых значений. Более подробно данный метод с теоретической точки зрения данный метод описан в пункте 2.1.2 раздела 2. На рисунке 3.4 представлена схема алгоритма прогнозирования методом простого скользящего среднего.

Рис. 3.4 – Схема алгоритма метода простого скользящего среднего

* + 1. **Алгоритм работы метода наименьших квадратов**

В первую очередь определяется, какого размера будет массив, хранящий исходные (в том числе анализируемые) и прогнозируемые данные (это же в последующем точки графика). Далее в этот массив на основе массива с данными о периодах вносятся исходные данные, а также вычисляются суммы, используемые в формуле расчёта коэффициентов для прогнозирования данным методом. Затем вычисляются коэффициенты, после чего осуществляется определение прогнозируемых значений. Метод наименьших квадратов более подробно (с теоретической точки зрения) описан и рассмотрен в пункте 2.1.3 раздела 2. На рисунке 3.5 представлена схема алгоритма прогнозирования методом наименьших квадратов.

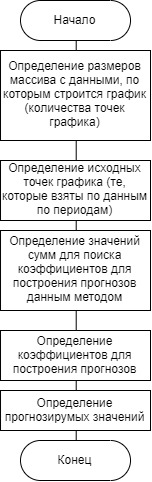
****

Рис. 3.5 – Схема алгоритма метода наименьших квадратов

* + 1. **Алгоритмы иных видов анализа**

Под иными видами анализа имеются ввиду: идентификация лучшего и худшего наименования среди имеющихся (по количеству продаж или по прибыли), определение наиболее популярного производителя (исходя из данных о том, каково количество наименований с определённым производителем), идентификация лучших товаров по производителям (исходя из данных о количестве продаж или прибыли), расчёт общей прибыли, трат и итогов за имеющийся период, обнаружение лучшей даты в периоде (исходя из данных о прибыли, тратах или итогах). Большинство из приведённых действий реализуются проходом по массиву данных с определённого действия (сравнение, сложение) с определённой информацией из массива данных, алгоритмы некоторых из этих действий дополнены предварительной сортировкой (по производителям), а сами действия выполняются через массив-копию массива с данными (например, идентификация лучших товаров по производителям). Предварительная сортировка необходима для упрощения алгоритма (в данном случае осуществляется проход по упорядоченным производителям, что позволяет последовательно определять искомую информацию для каждого из производителей, в противном случае такое было бы невозможно и необходимо было бы осуществлять многократный проход по массиву с сравнениями и каким-либо образом отмечать уже пройденные элементы).

* + 1. **Алгоритмы добавления и изменения данных пользователем в массив данных**

После нажатия на кнопку «Добавить» основного окна (основной формы) откроется отдельное окно (форма) для определения информации для добавления данных в массив данных (один из двух), в этом окне имеются определённые компоненты, которые позволяют определить информацию об элементе (поля ввода, тайм пикеры, чек бокс). Пользователь осуществляет заполнение информации в данных компонентах, после чего необходимо нажать на кнопку «ОК». После этого будет проведена проверка корректности заполненной информации (например, количество не должно быть отрицательным) и, в случае её успешного прохождения на основе заполненной информации в массив данных будет произведено добавление, а затем обновление отображения, если же проверка не будет пройдена, будет выведен message box, информирующий о допущенных ошибках. Также в окне добавления имеется кнопка «Отменить», после нажатия которой будет выведен подтверждающий message box, подтверждение приведёт к закрытию окна без осуществления добавления, те же действия повторяются при нажатии на кнопку закрытия на рамке окна. Алгоритм изменения идентичен алгоритму добавления за исключением того, что в случае изменения в компонентах для заполнения будет находиться текущая информация об изменяемом товаре (периоде).

Общая схема алгоритмов добавления и изменения данных представлена на рисунке 3.6.

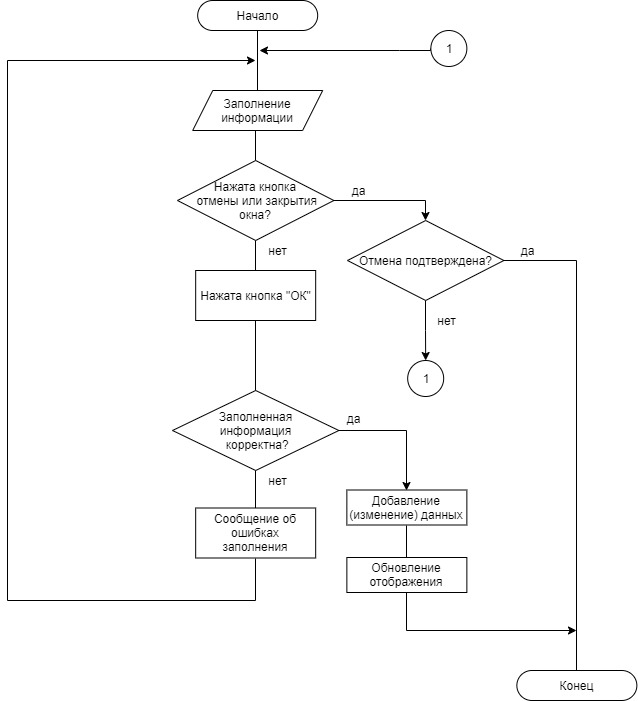


Рис. 3.6 – Общая схема алгоритмов добавления и изменение

# **СОЗДАНИЕ (КОНСТРУИРОВАНИЕ) ПРОГРАММНОГО СРЕДСТВА**

При создании программного средства я использовал три модуля: основной модуль (MainUnit), модуль для добавления и изменения (AddForm) и модуль для прогнозирования и построения графиков (Graphic). Отображение данных одного из массивов данных происходит с помощью listview на форме Form1 из основного модуля. Каждый из них имеет собственную форму.

# **Подпрограммы модуля MainUnit**

В таблице 4.1 представлены подпрограммы модуля MainUnit, обеспечивающие взаимодействие пользователя и программы, в таблице 4.2 – все остальные подпрограммы данного модуля.

Таблица 4.1 – Подпрограммы модуля MainUnit, обеспечивающие взаимодействие пользователя и программы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Назначение** | **Заголовок** | **Имя параметра** | **Назначение параметра** |
| FormCreate | Задание начальной директории сохранения и открытия файлов и разворот программы в полный экран при открытии приложения. | procedure TForm1.FormCreate (Sender: TObject); | Sender | Объект, который сгенерировал событие |
| FormCloseQuery | Подтверджение желания закрытия приложения | procedure TForm1.FormCloseQuery(Sender: TObject; var CanClose: Boolean); | Sender | Объект, который сгенерировал событие |
| CanClose | Показывает, может ли быть закрыто окно |
| actAddExecute | Добавление данных в один из массивов данных (т.е. добавление товара или периода\периодов) | procedure TForm1.actAddExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| actEditExecute | Изменение элемента (выбранного в отображении listview массива данных) | procedure TForm1.actEditExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |

Продолжение таблицы 4.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Назначение** | **Заголовок** | **Имя параметра** | **Назначение параметра** |
| actRemoveExecute | Удаление элемента (выбранного в отображении listview массива данных) | procedure TForm1.actRemoveExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| actClearExecute | Удаление всех элементов (одного из массивов данных) | procedure TForm1.actClearExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| btnTable1Click | Переключение на отображение данных о товарах | procedure TForm1.btnTable1Click(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| btnTable2Click | Переключение на отображение данных о периодах | procedure TForm1.btnTable2Click(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| ComBoxSrchSelect | Выбор в комбо боксе колонки listview для поиска | procedure TForm1.ComBoxSrchSelect(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| EditMaskSrchKeyPress | Обработка нажатия Enter при вводе информации для поиска (для начала поиска) | procedure TForm1.EditMaskSrchKeyPress(Sender: TObject; var Key: Char); | Sender | Объект, который сгенерировал событие |
| Key | Нажатая клавиша клавиатуры |
| lvReportColumnClick | Обработка нажатия на заголовок колонки listview (для начала сортировки или сообщения о её недоступности) | procedure TForm1.lvReportColumnClick(Sender: TObject; Column: TListColumn); | Sender | Объект, который сгенерировал событие |
| Column | Номер колонки с нажатым заголовком |
| actOpenExecute | Открытие файла | procedure TForm1.actOpenExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |

Продолжение таблицы 4.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Назначение** | **Заголовок** | **Имя параметра** | **Назначение параметра** |
| actSaveExecute | Сохранение данных из одного из массивов данных в файл | procedure TForm1.actSaveExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| actSaveAllExecute | Сохранение в файлы данных из обоих массивов данных | procedure TForm1.actSaveAllExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| actExitExecute | Выход из приложения | procedure TForm1.actExitExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| actGraphicExecute | Открытие окна для построения прогнозов с графиками | procedure TForm1.actGraphicExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| actBWSalesAmountExecute | Идентификация лучшего и худшего товара по количеству продаж | procedure TForm1.actBWSalesAmountExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| actBWSalesIncomeExecute | Идентификация лучшего и худшего товара по прибыли | procedure TForm1.actBWSalesIncomeExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| actMPProdExecute | Идентификация самого популярного производителя (по количеству товаров) | procedure TForm1.actMPProdExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| actMPNameProdAmountExecute | Идентификация самых популярных товаров у каждого производителя по количеству продаж | procedure TForm1.actMPNameProdAmountExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| actMPNameProdIncomeExecute | Идентификация самых популярных товаров у каждого производителя по прибыли | procedure TForm1.actMPNameProdIncomeExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |

Продолжение Таблицы 4.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Назначение** | **Заголовок** | **Имя параметра** | **Назначение параметра** |
| actTotalAnExecute | Рассчёт общих прибыли, трат и итогов за периоды | procedure TForm1.actTotalAnExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| actBWIncomeExecute | Определение лучшего и худшего периода по прибыли | procedure TForm1.actBWIncomeExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| actBWExpenseExecute | Определение лучшего и худшего периода по тратам | procedure TForm1.actBWExpenseExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| actBWTotalExecute | Определение лучшего и худшего периода по итогам | procedure TForm1.actBWTotalExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |

Таблица 4.2 – Остальные подпрограммы модуля MainUnit

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Назначение** | **Заголовок** | **Имя параметра** | **Назначение параметра** |
| AnalyzeTable | Идентификация худшего и лучшего (5 вариантов) | procedure TForm1.AnalyzeTable(CaseToAnalyze: byte); | CaseToAnalyze | Определяет, какой из вариантов индентификации выполнять |
| AddTotal | Добавление данных в массив периодов | procedure TForm1.AddTotal(StartDate, EndDate: TDate); | StartDate | Начальный период |
| EndDate | Конечный период |
| RemovePosition | Удаление элемента из массива товаров | procedure RemovePosition(I: integer); | I | Индекс удаляемого элемента |

Продолжение таблицы 4.2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Назначение** | **Заголовок** | **Имя параметра** | **Назначение параметра** |
| AddPosition | Добавление элемента в массив товаров | procedure AddPosition(const Code, Name, Prod: string; Amount: integer; price: Real; VAT: integer; PriceSold: real; AmountSold:integer); | Code | Код |
| Name | Наименование |
| Prod | Производитель |
| Amount | Количество |
| price | Закупочная цена |
| VAT | НДС |
| PriceSold | Цена продажи |
| AmountSold | Количество продаж |
| EditTotal | Изменение элемента в массиве периодов | procedure EditTotal(i: integer; Income, Expense: real); | i | Индекс элемента для изменения |
| Income | Прибыль |
| Expense | Траты |
| EditPosition | Изменение элемента в массиве товаров | procedure EditPosition(I: integer; const Code, Name, Prod: string; Amount: integer; price: Real; VAT: integer; PriceSold:real; amountSold:integer); | I | Индекс изменяемого элемента |
| Code | Код |
| Name | Наименование |
| Prod | Производитель |
| Amount | Количество |
| price | Закупочная цена |
| VAT | НДС |
| PriceSold | Цена продажи |
| amountSold | Количество продаж |
| UpdateView | Обновление отображения | procedure UpdateView(ReportUpdate: TReportPos; ReportUpdate2: TReport); | ReportUpdate | Массив товаров для отображения |
| ReportUpdate2 | Массив периодов для отображения |

Продолжение таблицы 4.2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Назначение** | **Заголовок** | **Имя параметра** | **Назначение параметра** |
| Search | Поиск элемента в массиве | procedure Search(text: string); | text | Искомая информация |
| SearchPosSetItem | Возвращает определённую информацию о товаре в зависимости от того, по чему ведётся поиск | function SearchPosSetItem(index: integer):string; | index | Индекс элемента в массиве товаров |
| SearchTotalSetItem | Возвращает определённую информацию о периоде в зависимости от того, по чему ведётся поиск | function SearchTotalSetItem(index: integer):string; | index | Индекс элемента в массиве периодов |
| SortModel | Сортировка | procedure SortModel(column: integer;var UpdateReport: TReportPos); | column | Колонка, по которой производится сортировка |
| UpdateReport | Сортируемый массив |
| CountXSort | В зависимости от колонки, по которой сортировка, возвращает определённую информацию о товаре | function CountXSort(left, right, ColumnNum: integer;UpdateReport: TReportPos): variant; | left | Текущая левая граница сортировки |
| right | Текущая правая граница сортировки |
| ColumnNum | Номер колонки |
| UpdateReport | Сортируемый массив |
| Check1Sort | Проверка, меньше ли информация о товаре (в зависимости от колонки, по которой сортировка) под текущим индексом текущей записанной | function Check1Sort(i: integer; x: variant; ColumnNum:integer;UpdateReport: TReportPos): boolean; | i | Индекс товара |
| x | Записанная информация о товаре |
| ColumnNum | Номер колонки |
| UpdateReport | Сортируемый массив |

Продолжение таблицы 4.2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Назначение** | **Заголовок** | **Имя параметра** | **Назначение параметра** |
| Check2Sort | Проверка, больше ли информация о товаре (в зависимости от колонки, по которой сортировка) под текущим индексом текущей записанной | function Check2Sort(j: integer; x: variant; ColumnNum:integer;UpdateReport: TReportPos): boolean; | j | Индекс товара |
| x | Записанная информация о товаре |
| ColumnNum | Номер колонки |
| UpdateReport | Сортируемый массив |
| Check3Sort | Сравнение информации между двумя товарами (в зависимости от колонки, по которой сортировка) | function Check3Sort(i, j, ColumnNum:integer;UpdateReport: TReportPos): boolean; | i | Индекс одного из сравниваемых товара |
| j | Индекс одного из сравниваемых товара |
| ColumnNum | Номер колонки |
| UpdateReport | Сортируемый массив |
| MostPopularProd | Определение самого популярного производителя | procedure MostPopularProd; | - | - |
| MostPopularProdSearch | Поиск наиболее популярного производителя | procedure MostPopularProdSearch(BuferReport: TReportPos); | BuferReport | Массив для поиска |
| MostPopularNameProdAmount | Поиск наиболее популярных товаров по производителям (по количеству продаж) | procedure MostPopularNameProdAmount(BuferReport: TReportPos); | BuferReport | Массив для поиска |
| MostPopularNameProdIncome | Поиск наиболее популярных товаров по производителям (по прибыли) | procedure MostPopularNameProdIncome(BuferReport: TReportPos); | BuferReport | Массив для поиска |

# **Подпрограммы модуля AddForm**

В таблице 4.3 представлены подпрограммы модуля AddForm, обеспечивающие взаимодействие пользователя и программы, в таблице 4.4 – все остальные подпрограммы данного модуля.

Таблица 4.3 – Подпрограммы модуля AddForm, обеспечивающие взаимодействие пользователя и программы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Назначение** | **Заголовок** | **Имя параметра** | **Назначение параметра** |
| FormShow | Установка фокуса на определённый компонент по показу формы | procedure TForm2.FormShow(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| FormCloseQuery | Если закрытие формы по кнопке «Ок», то проверка возможных ошибок заполнения, иначе вопрос с подтверждением закрытия | procedure TForm2.FormCloseQuery(Sender: TObject; var CanClose: Boolean); | Sender | Объект, который сгенерировал событие |
| CanClose | Определяет, может ли быть закрыта форма |
| EditLbPriceKeyPress | Допуск в поле для ввода закупочной цены только цифр и одной запятой или точки | procedure TForm2.EditLbPriceKeyPress(Sender: TObject; var Key: Char); | Sender | Объект, который сгенерировал событие |
| Key | Нажатая клавиша клавиатуры |
| EditLbPriceSoldKeyPress | Допуск в поле для ввода цены продажи только цифр и одной запятой или точки | procedure TForm2.EditLbPriceSoldKeyPress(Sender: TObject; var Key: Char); | Sender | Объект, который сгенерировал событие |
| Key | Нажатая клавиша клавиатуры |
| EditLbVATKeyPress | Допуск в поле для ввода НДС только цифр | procedure TForm2.EditLbVATKeyPress(Sender: TObject; var Key: Char); | Sender | Объект, который сгенерировал событие |
| Key | Нажатая клавиша клавиатуры |

Таблица 4.4 – Остальные подпрограммы модуля AddForm

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Назначение** | **Заголовок** | **Имя параметра** | **Назначение параметра** |
| AddNewPosition | Подготовительные действия (для добавления товара) к показу формы и её показ | function AddNewPosition :integer; | - | - |
| AddNewTotal | Подготовительные действия (для добавления периода) к показу формы и её показ | function AddNewTotal: integer; | - | - |
| EditCurTotal | Подготовительные действия (для изменения периода) к показу формы и её показ | function EditCurTotal (Income, Expense: real): integer; | Income | Прибыль при вызове изменения |
| Expense | Траты при вызове изменения |
| EditCurPosition | Подготовительные действия (для изменения товара) к показу формы и её показ | function EditCurPosition(const Code, Name, Prod: string; Amount: integer; price: Real; VAT: integer;PriceSold:real;AmountSold:integer):integer;) | Code | Код |
| Name | Наимнование |
| Prod | Производитель |
| Amount | Количество |
| price | Закупочная цена |
| VAT | НДС |
| PriceSold | Цена продажи |
| AmountSold | Количество продаж |
| GetCode | Возвращает код из поля для его ввода | function GetCode: string; | - | - |
| GetName | Возвращает наименование из поля для его ввода | function GetName: string; | - | - |
| GetProd | Возвращает производителя из поля для его ввода | function GetProd: string; | - | - |

Продолжение таблицы 4.4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Назначение** | **Заголовок** | **Имя параметра** | **Назначение параметра** |
| GetAmount | Возвращает количество из поля для его ввода | function GetAmount: integer; | - | - |
| GetPrice | Возвращает закупочную цену из поля для её ввода | function GetPrice: real; | - | - |
| GetVAT | Возвращает НДС из поля для его ввода | function GetVAT: integer; | - | - |
| GetAmountSold | Возвращает количество продаж из поля для его ввода | function GetAmountSold: integer; | - | - |
| GetPriceSold | Возвращает цену продажи из поля для её ввода | function GetPriceSold: real; | - | - |
| GetStartDate | Возвращает дату начала периода из таймпикера для её выбора | function GetStartDate: TDate; | - | - |
| GetEndDate | Возвращает дату конца периода из таймпикера для её выбора | function GetEndDate: TDate; | - | - |
| GetExpense | Возвращает траты из поля для их ввода | function GetExpense: real; | - | - |
| GetIncome | Возвращает прибыль из поля для её ввода | function GetIncome: real; | - | - |
| GetIsMonth | Возвращает True или False в зависимости от checkbox’а на форме | function GetIsMonth: boolean; | - | - |

# **Подпрограммы модуля Graphic**

В таблице 4.5 представлены подпрограммы модуля Graphic, обеспечивающие взаимодействие пользователя и программы, в таблице 4.6 – все остальные подпрограммы данного модуля.

Таблица 4.5 – Подпрограммы модуля Graphic, обеспечивающие взаимодействие пользователя и программы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Назначение** | **Заголовок** | **Имя параметра** | **Назначение параметра** |
| FormCreate | Установка начальной директории для сохранения графиков по созданию формы | procedure TForm3.FormCreate(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| ComBoxMethodSelect | Активация трэк бара (для выбора коэффициента  Экспоненциального сглаживания), если выбран метод экспоненциального сглаживания | procedure TForm3.ComBoxMethodSelect(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| btnDrawClick | Построение прогноза с графиком по нажатию на кнопку | procedure TForm3.btnDrawClick(Sender: TObject); | Sender | Объект, который сгенерировал событие |

Таблица 4.6 – Остальные подпрограммы модуля Graphic

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Назначение** | **Заголовок** | **Имя параметра** | **Назначение параметра** |
| FindMaxMin | Поиск наибольшего и наименьшего значения точек | procedure FindMaxMin(var HighLim, LowLim:real); | HighLim | Переменная для наибольшего значения |
| LowLim | Переменная для наименьшего значения |
| FindDotsExponential | Построение прогноза методом экпоненциального сглаживания (и экспоненциальной сглаженной) | procedure FindDotsExponential; | - | - |
| FindDotsAverage | Построение прогноза методом скользящей средней | procedure FindDotsAverage; | - | - |

Продолжение таблицы 4.6

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Назначение** | **Заголовок** | **Имя параметра** | **Назначение параметра** |
| FindDotsLowSqr | Построение прогноза методом наименьших квадратов | procedure FindDotsLowSqr; | - | - |
| BuildGraphic | Построение графика | procedure BuildGraphic(HighLim, LowLim:real); | HighLim | Наибольшее значение на графике |
| LowLim | Наименьшее значение на графике |

# **ТЕСТИРОВАНИЕ, ПРОВЕРКА РАБОТОСПОСОБНОСТИ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ**

В таблице 5.1 представлены результаты тестирования созданного программного средства.

Таблица 5.1 – Тестирование возможностей программы

|  |  |  |  |
| --- | --- | --- | --- |
| **Предмет теста** | **Условия и ход тестирования** | **Ожидаемый результат** | **Полученный результат** |
| Добавление товара | Массив товаров пуст. Нажимается кнопка «Добавить», выполняется корректный ввод данных, нажимается кнопка «ОК». | В массив товаров добавлен товар, исходя из введённой информации. Отображение обновлено. | **Тест пройден** |
| Массив товаров пуст. Нажимается кнопка «Добавить», выполняется некорректный ввод данных, нажимается кнопка «ОК». | Выводится сообщение о некорректном вводе с перечнем допущенных ошибок | **Тест пройден** |
| Массив товаров пуст. Нажимается кнопка «Добавить», нажимается кнопка «Отменить». | Выводится сообщение с подтверждением закрытия формы | **Тест пройден** |
| Массив товаров не пуст. Нажимается кнопка «Добавить», выполняется ввод данных, код товара повторяется с уже имеющимся в массиве товаров, нажимается кнопка «ОК». | Выводится сообщение, предупреждающее о повторяющемся коде | **Тест пройден** |
| Добавление периода | Массив периодов пуст. Нажимается кнопка «Добавить», выполняется выбор начального и конечного периодов, чек бокс не отмечен, нажимается кнопка «ОК». | В массив периодов добавляются все даты по дням от начального периода до конечного включительно, обновляется отображение. | **Тест пройден** |

Продолжение таблицы 5.1

|  |  |  |  |
| --- | --- | --- | --- |
| **Предмет теста** | **Условия и ход тестирования** | **Ожидаемый результат** | **Полученный результат** |
| Добавление периода | Массив периодов пуст. Нажимается кнопка «Добавить», выполняется выбор начального и конечного периодов, чек бокс отмечен, нажимается кнопка «ОК». | В массив периодов добавляются все даты по месяцам от начального периода до конечного, обновляется отображение. | **Тест пройден** |
| Массив периодов пуст. Нажимается кнопка «Добавить», выполняется выбор начального и конечного периодов, начальный период позже конечного, нажимается кнопка «ОК». | Выводится сообщение об ошибке | **Тест пройден** |
| Массив периодов не пуст, периоды по дням. Нажимается кнопка «Добавить». | В массив периодов добавляется дата дня, следующего за последним до добавления, отображение обновляется. | **Тест пройден** |
| Массив периодов не пуст, периоды по месяцам. Нажимается кнопка «Добавить». | В массив периодов добавляется дата месяца, следующего за последним до добавления, отображение обновляется. | **Тест пройден** |
| Изменение | Массив товаров не пуст. В отображении выбирается товар для изменения, нажимается кнопка «Изменить», выполняется корректное изменение, нажимается кнопка «ОК» | Изменения вносятся в массив товаров, отображение обновляется | **Тест пройден** |

Продолжение таблицы 5.1

|  |  |  |  |
| --- | --- | --- | --- |
| **Предмет теста** | **Условия и ход тестирования** | **Ожидаемый результат** | **Полученный результат** |
| Изменение | Массив товаров не пуст. В отображении выбирается товар для изменения, нажимается кнопка «Изменить», выполняется изменение, код для изменения повторяется с имеющимся в массиве товаров, нажимается кнопка «ОК» | Выводится сообщение, что товар с данным кодом уже существует | **Тест пройден** |
| Удаление | Массив товаров не пуст. В отображении выбирается товар для удаления, нажимается кнопка «Удалить» | Товар удаляется из массива товаров, отображение обновляется | **Тест пройден** |
| Очистка | Массив товаров не пуст. Нажимается кнопка «Очистить список» | Массив товаров полностью очищается, отображение обновляется | **Тест пройден** |
| Сортировка | Массив товаров не пуст и не отсортирован. Нажимается на заголовок listview с названием «Код» | Массив товаров сортируется по информации о кодах товаров, отображение обновляется | **Тест пройден** |
| Поиск | Массив периодов не пуст. В комбобоксе, указывающем на колонку для поиска, выбрано «Прибыль». В поле ввода информации для поиска вводится информация, нажимается Enter. Информация для поиска существует. | Отображение обновлено в соответствии с найденными периодами | **Тест пройден** |
| Открытие файла | Массив периодов пуст. В разделе «Файл» меню программы нажимается кнопка «Открыть», выбирается корректный файл расширения .table | Информация из файла вносится в массив периодов, отображение обновляется | **Тест пройден** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Предмет теста** | **Условия и ход тестирования** | **Ожидаемый результат** | **Полученный результат** |
| Сохранение | Массив периодов не пуст. В разделе «Файл» меню программы нажимается кнопка «Сохранить», выбирается расширение .table, введено корректное имя | Данные из массива периодов сохраняются в файл с указанным именем и расширением .table | **Тест пройден** |
| Анализ | Массив товаров не пуст. В разделе «Анализ» меню раскрывается подраздел «Таблица товаров», выбирается пункт «Наиболее и наименее популярный производитель» | Вывод сообщения, указывающего на наиболее и наименее популярных производителей, указывая количество товаров у них | **Тест пройден** |
| Массив товаров не пуст. В разделе «Анализ» меню раскрывается подраздел «Таблица товаров», раскрывается подраздел «Лучшее и худшее по», выбирается пункт «продажам (количество)» | Выводится сообщение, указывающее Лучший и худший товар по количеству продаж | **Тест пройден** |
| Массив товаров не пуст. В разделе «Анализ» меню раскрывается подраздел «Таблица товаров», раскрывается подраздел «Лучшие товары по произ-водителям», выбирается пункт «По количеству продаж» | Вывод сообщения с лучшими товарами по количеству продаж у каждого из производителей | **Тест пройден** |
| Массив периодов не пуст. В разделе «Анализ» меню раскрывается подраздел «Таблица итогов», выбирается пункт «Общий итог, прибыль и траты» | Вывод сообщения с общими прибылью, тратами и итогами за все периоды | **Тест пройден** |

Продолжение таблицы 5.1

Продолжение таблицы 5.1

|  |  |  |  |
| --- | --- | --- | --- |
| **Предмет теста** | **Условия и ход тестирования** | **Ожидаемый результат** | **Полученный результат** |
| Анализ | Массив периодов не пуст. В разделе «Анализ» меню раскрывается подраздел «Таблица итогов», раскрывается подраздел «Лучшая и худшая дата по», выбирается пункт «Прибыли» | Выводится сообщение, указывающее худшую и лучшую даты по прибыли | **Тест пройден** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Предмет теста** | **Условия и ход тестирования** | **Ожидаемый результат** | **Полученный результат** |
| Прогнозирование и построение графиков | Массив периодов содержит 6 периодов. Нажимается пункт «Графики» меню. Выбирается метод прогнозирования «Скользящего среднего», данные для анализа – прибыль, нажимается кнопка «Построить график» | На изображении нарисован график по исходным данным о прибыли с прогнозным отрезком пунктирной линией | **Тест пройден** |
| Массив периодов содержит 6 периодов. Нажимается пункт «Графики» меню. Выбирается метод прогнозирования «Скользящего среднего», данные для анализа – итоги, нажимается кнопка «Построить график» | На изображении нарисован график по исходным данным об итогах с прогнозным отрезком пунктирной линией | **Тест пройден** |

В таблице 5.3 представлены результаты тестирования возможностей прогнозирования и построения графиков програмы.

Таблица 5.3 – Тестирование функций прогнозирования и простроения графиков

Продолжение таблицы 5.2

|  |  |  |  |
| --- | --- | --- | --- |
| **Предмет теста** | **Условия и ход тестирования** | **Ожидаемый результат** | **Полученный результат** |
|  | Массив периодов содержит 6 периодов. Нажимается пункт «Графики» меню. Выбирается метод прогнозирования «Скользящего среднего», данные для анализа – траты, нажимается кнопка «Построить график» | На изображении нарисован график по исходным данным об тратах с прогнозным отрезком пунктирной линией | **Тест пройден** |
|  | Массив периодов содержит 6 периодов. Нажимается пункт «Графики» меню. Выбирается метод прогнозирования «Экпоненциального сглаживания», данные для анализа – прибыль, коэффициент сглаживания – 0.2, нажимается кнопка «Построить график» | На изображении нарисованы график по исходным данным о прибыли с прогнозным отрезком пунктирной линией и график экпоненциального сглаживания | **Тест пройден** |
| Массив периодов содержит 6 периодов. Нажимается пункт «Графики» меню. Выбирается метод прогнозирования «Скользящего среднего», данные для анализа – траты, коэффициент сглаживания – 0.7, нажимается кнопка «Построить график» | На изображении нарисованы график по исходным данным о тратах с прогнозным отрезком пунктирной линией и график экпоненциального сглаживания | **Тест пройден** |
| Массив периодов содержит 6 периодов. Нажимается пункт «Графики» меню. Выбирается метод прогнозирования «Скользящего среднего», данные для анализа – итоги, коэффициент сглаживания – 0.4, нажимается кнопка «Построить график» | На изображении нарисованы график по исходным данным об итогах с прогнозным отрезком пунктирной линией и график экпоненциального сглаживания | **Тест пройден** |

Продолжение таблицы 5.2

|  |  |  |  |
| --- | --- | --- | --- |
| **Предмет теста** | **Условия и ход тестирования** | **Ожидаемый результат** | **Полученный результат** |
|  | Массив периодов содержит 6 периодов. Нажимается пункт «Графики» меню. Выбирается метод прогнозирования «Наименьших квадратов», данные для анализа – прибыль, нажимается кнопка «Построить график» | На изображении нарисован график по исходным данным о прибыли с прогнозным отрезком пунктирной линией | **Тест пройден** |
| Массив периодов содержит 6 периодов. Нажимается пункт «Графики» меню. Выбирается метод прогнозирования «Наименьших квадратов», данные для анализа – траты, нажимается кнопка «Построить график» | На изображении нарисован график по исходным данным о тратах с прогнозным отрезком пунктирной линией | **Тест пройден** |
| Массив периодов содержит 6 периодов. Нажимается пункт «Графики» меню. Выбирается метод прогнозирования «Наименьших квадратов», данные для анализа – итоги, нажимается кнопка «Построить график» | На изображении нарисован график по исходным данным об итогах с прогнозным отрезком пунктирной линией | **Тест пройден** |
| Сохранение графика | График построен. Нажимается кнопка «Сохранить график», вводится имя файла, выбирается расширение, подтверждается. | Изображение текущего построенного графика сохранилось в файл с введённым именем и выбранным расширением | **Тест пройден** |

В результате тестирования программного средства все проведённые тесты, которые были направлены на проверку корректного выполнения программным средством поставленных задач, были успешно пройдены, что даёт основание полагаться на работоспособность разработанного программного средства.

# **РУКОВОДСТВО ПО УСТАНОВКЕ И ИСПОЛЬЗОВАНИЮ**

Установка дополнительных библиотек и модулей для корректной работы программного средства не требуется. Также само программное средство не требует установки – запуск можно осуществлять при наличии .exe файла программы. В данном разделе будет представлено словестно-графическое описание последовательностей действий, которые необходимо выполнить пользователю для использования той или иной возможности программного средства.

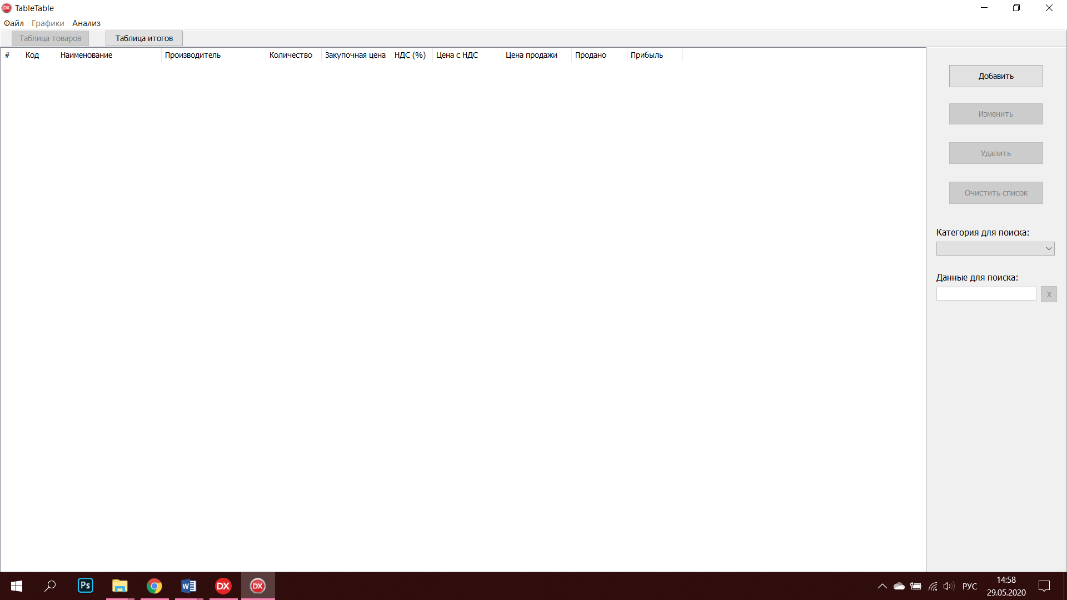
 Для запуска программы пользователю необходимо запустить файл TableTable.exe, после чего будет открыто главное окно программы, в котором будет представлена таблица товаров (см. рис. 6.1).

Рис. 6.1 – Главное окно программы

Для добавления товара необходимо нажать на кнопку «Добавить», расположенную в левой верхней части интерфейса программы. После этого будет открыто дополнительное окно, в котором предоставлены возможности внесения информации о товаре (см. рис. 6.2).

После заполнения всех полей корректной информацией, необходимо нажать на кнопку «Ок» или клавишу Enter клавиатуры, после чего товар будет добавлен (см. рис. 6.3). Для отмены процедуры добавления необходимо нажать на кнопку «Отменить» или закрыть окно и подтвердить действие.

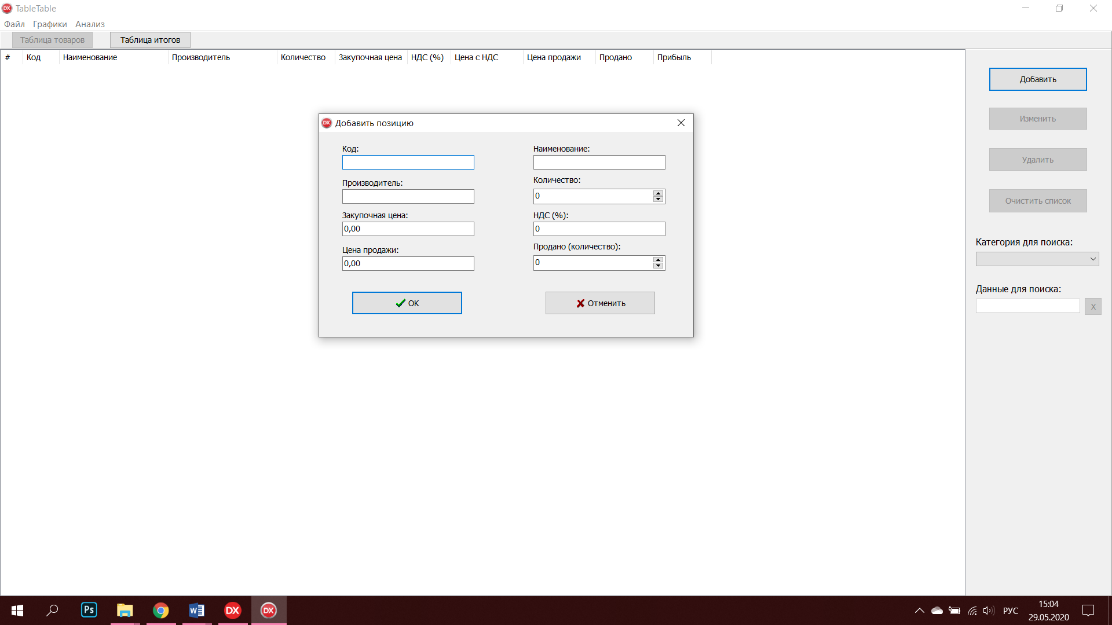
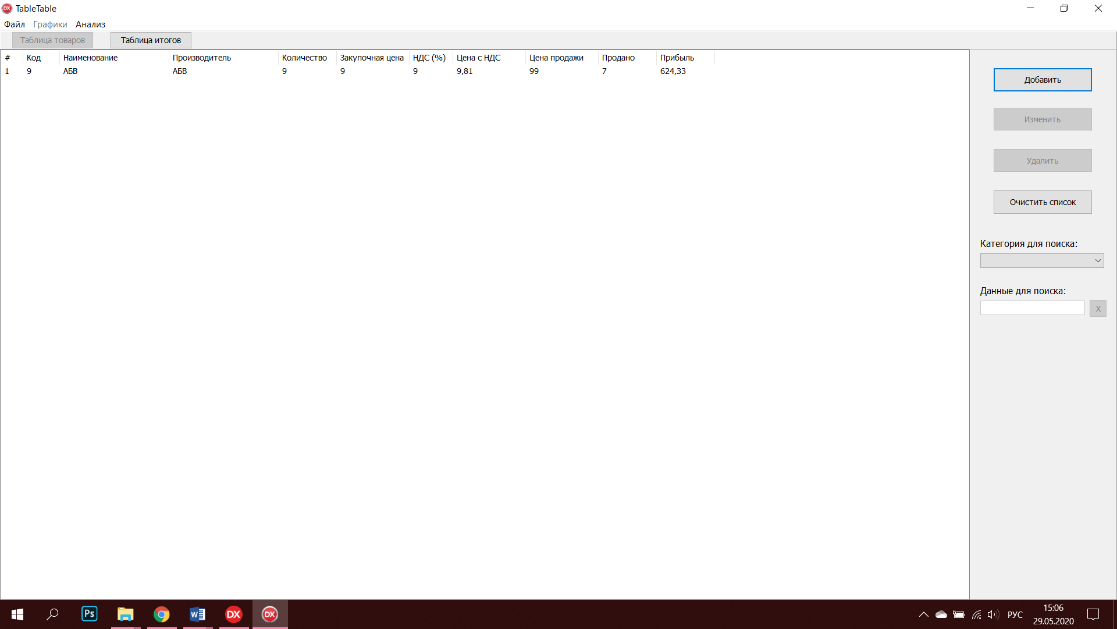
Рис. 6.2 – Окно для добавления товара

Рис. 6.3 – Товар добавлен

Для изменения позиции в таблице необходимо выбрать позицию для изменения, щёлкнув по ней мышью, после чего нажать кнопку «Изменить» в правой части окна. В результате будет окрыто окно изменения. Далее процедура изменения совпадает с процедурой добавления (см. рис. 6.4).

Для удаления товара из таблицы необходимо выбрать удаляемую позицию, щёлкнув по ней мышью, после чего нажать кнопку «Удалить». В результате вабранный товар будет удалён.

В случае, если требуется удалить все товары из таблицы, необходимо нажать на кнопку «Очистить список» (см. рис. 6.5-6.6).

Для переключения на таблицу товаров или таблицу итогов (периодов) необходимо использовать соответственно кнопки «Таблица товаров» и «Таблица итогов».

Последовательность действий изменения, добавления и удаления позиции для таблицы итогов идентична таковой в таблице товаров.

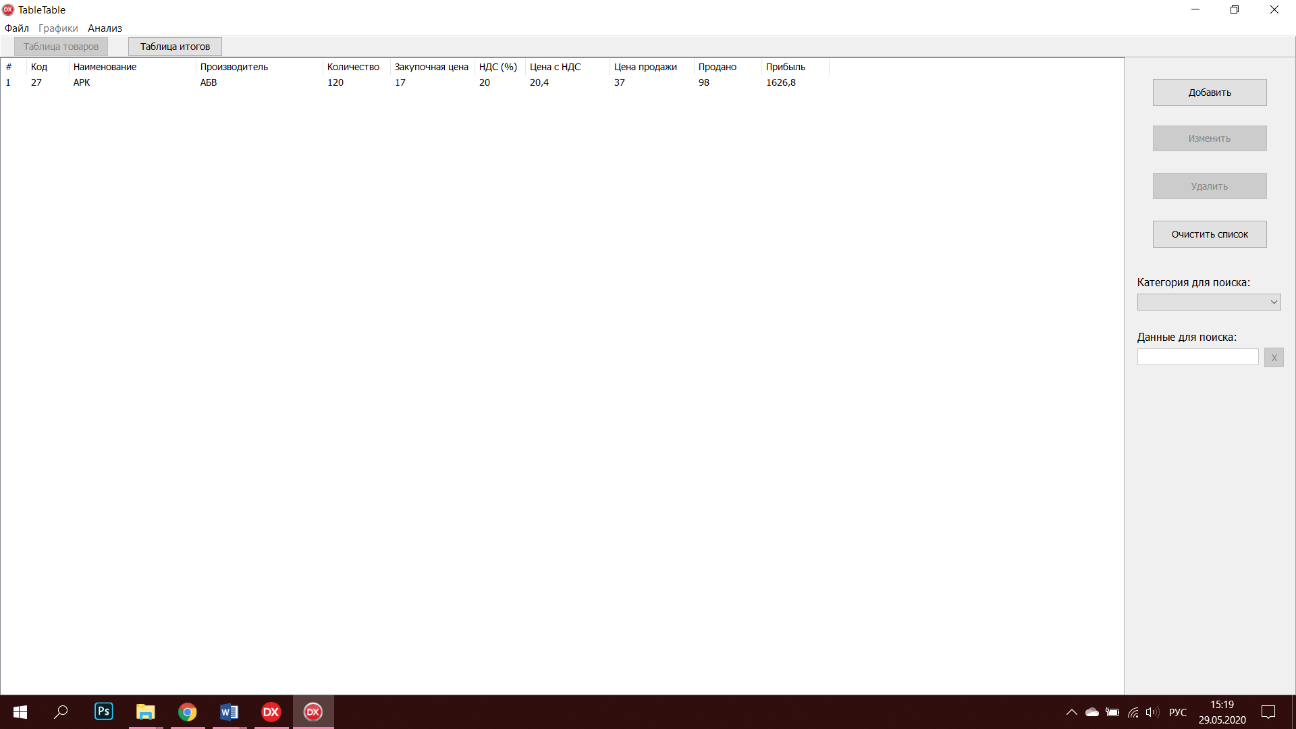
 При необходимости заполнить таблицу из файла, нужно открыть файл с таблицей. Для открытия файла необходимо навести мышь на раздел «Файл» меню и в раскрывшемся списке выбрать «Открыть» или использовать сочетание клавиш Ctrl+O. В появившемся диалоговом окне нужно выбрать файл таблицы, в результате данные из файла будут помещены в таблицу (см. рис. 6.7-6.8).

Рис. 6.4 – Товар изменён

Для сохранения одной из таблиц необходимо навести мышь на пункт «Файл» меню и в появившемся списке нажать на пункт «Сохранить» или использовать сочетание клавиш Ctrl+S. Для сохранения обеих таблиц, в появившемся списке необходимо выбрать пункт «Сохранить всё» или использовать сочетание клавиш Shift+Ctrl+S.

В данном окне (см. рис. 6.9) можно настроить параметры для построения графика с прогнозом, после чего необходимо нажать на кнопку «Построить», в результате чего будет нарисован график с прогнозом (см. рис. 6.10).

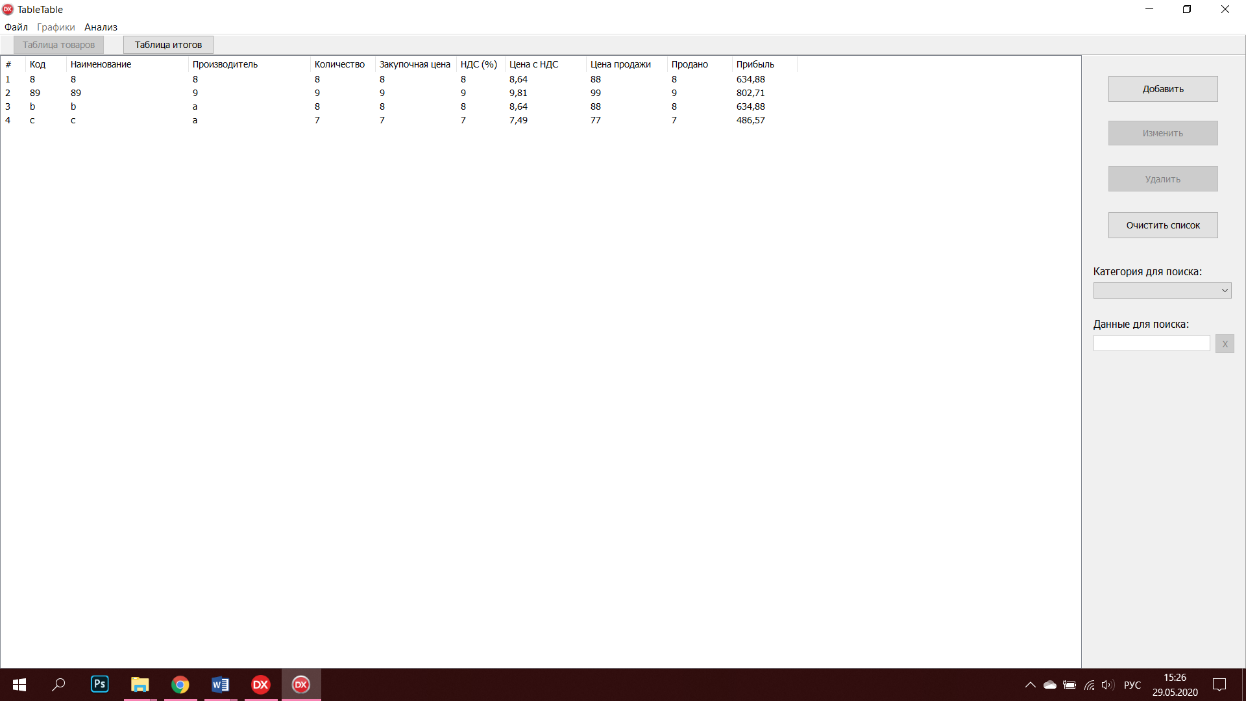
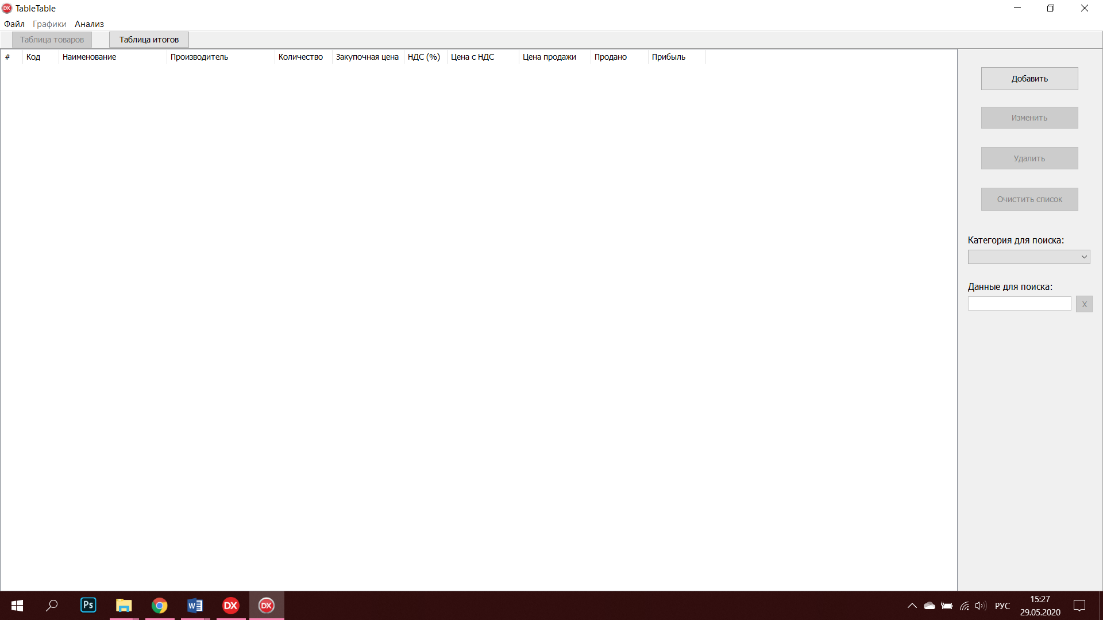
Рис 6.5 – Таблица до очистки

Рис. 6.6 – Таблица после очистки

Для доступа к анализу данных таблицы товаров необходимо навести мышь на пункт «Анализ» меню и в появившемся списке навести мышь на пункт «Таблица товаров». В результате в ещё одном появившемся списке будут представлены возможности анализа данных таблицы товаров (см. рис. 6.11)

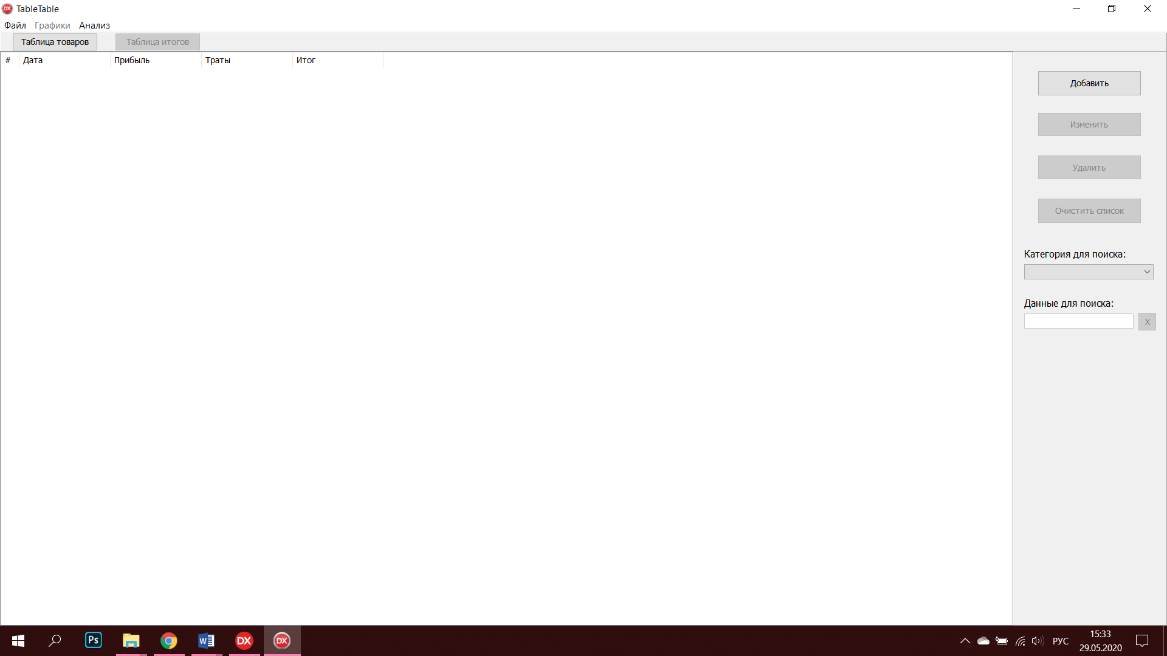
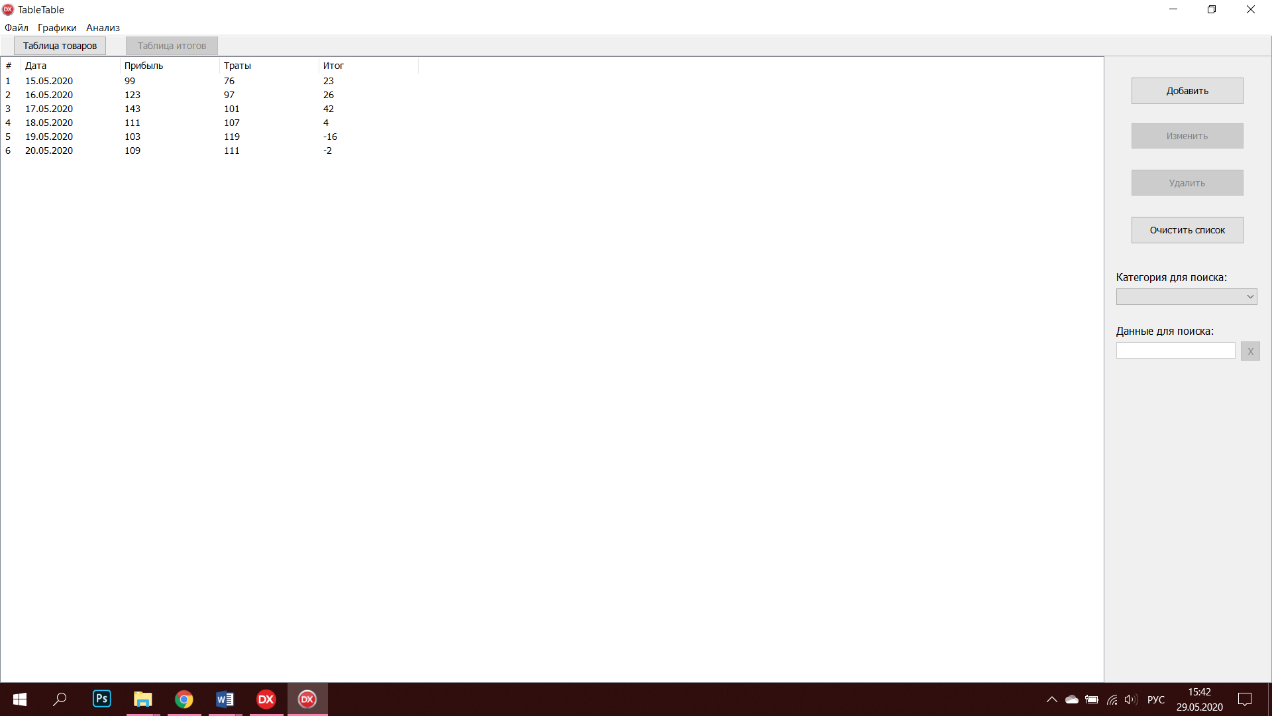
Рис. 6.7 – Таблица итогов до открытия файла

Рис. 6.8 – Открытие файла

При выборе пункта «Наиболее и наименее популярный производитель» будет предоставлена информация по данному пункту в виде информирующего сообщения, которое может быть закрыто по кнопке «Ок» или кнопки закрытия (см. рис. 6.12).

При наведении на пункты «Лучшее и худшее по…» или «Лучшие товары по производителям» появятся дополнительные списки для выбора( см рис. 6.13-6.14).

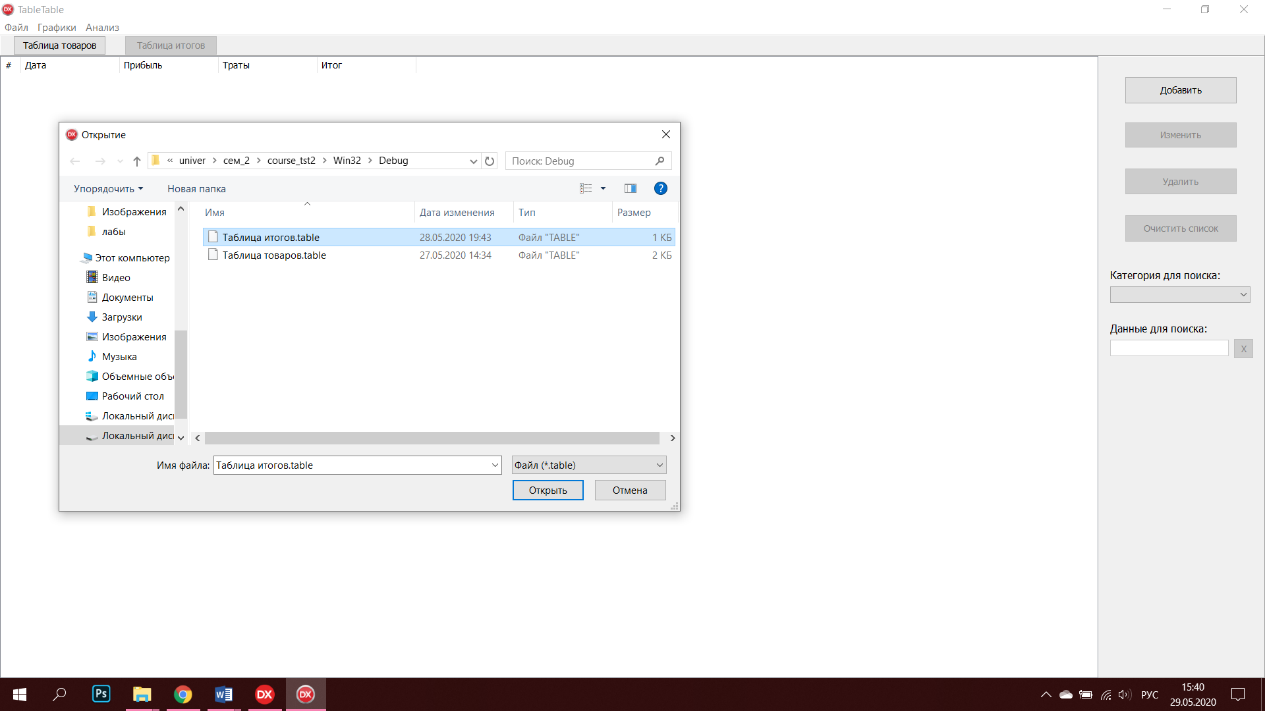
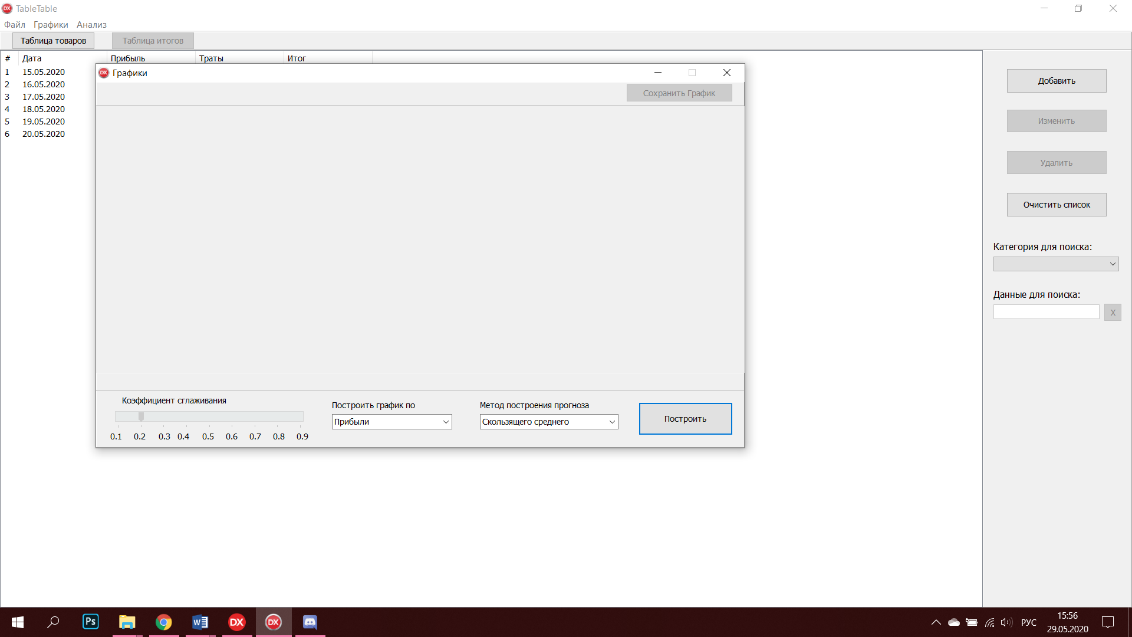
Рис. 6.9 – Результат открытия файла

Рис. 6.10 – Окно построения графиков с прогнозом

При выборе пунктов из данных списков (см. рис. 6.13-6.14) будет выведено информирующее сообщение с результатами выполнения.

Для доступа к анализу данных таблицы итогов (периодов) необходимо навести мышь на пункт «Анализ» меню и в появившемся списке навести мышь на пункт «Таблица итогов». В результате в ещё одном появившемся списке будут представлены возможности анализа данных таблицы товаров (см. рис. 6.15).

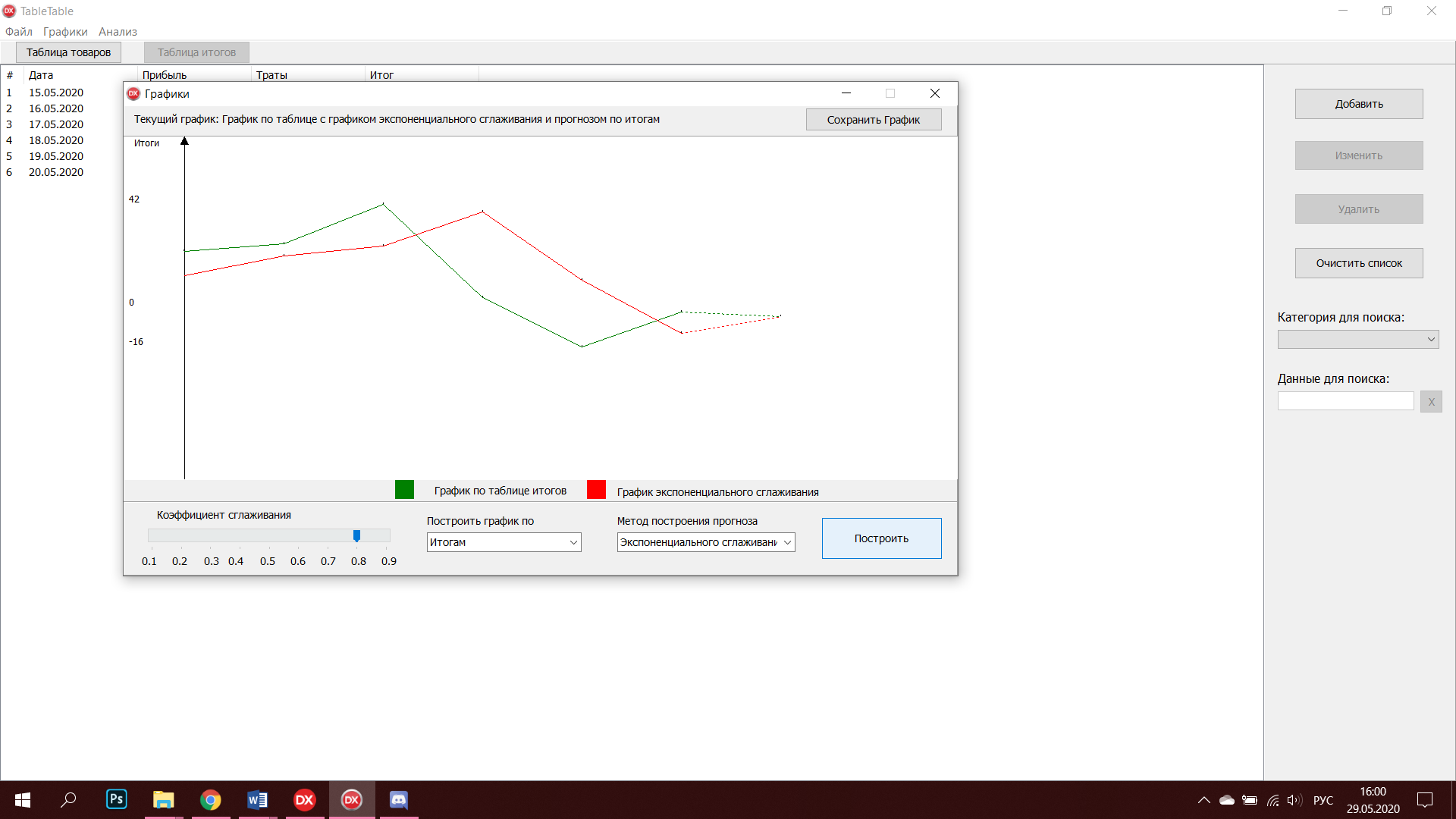
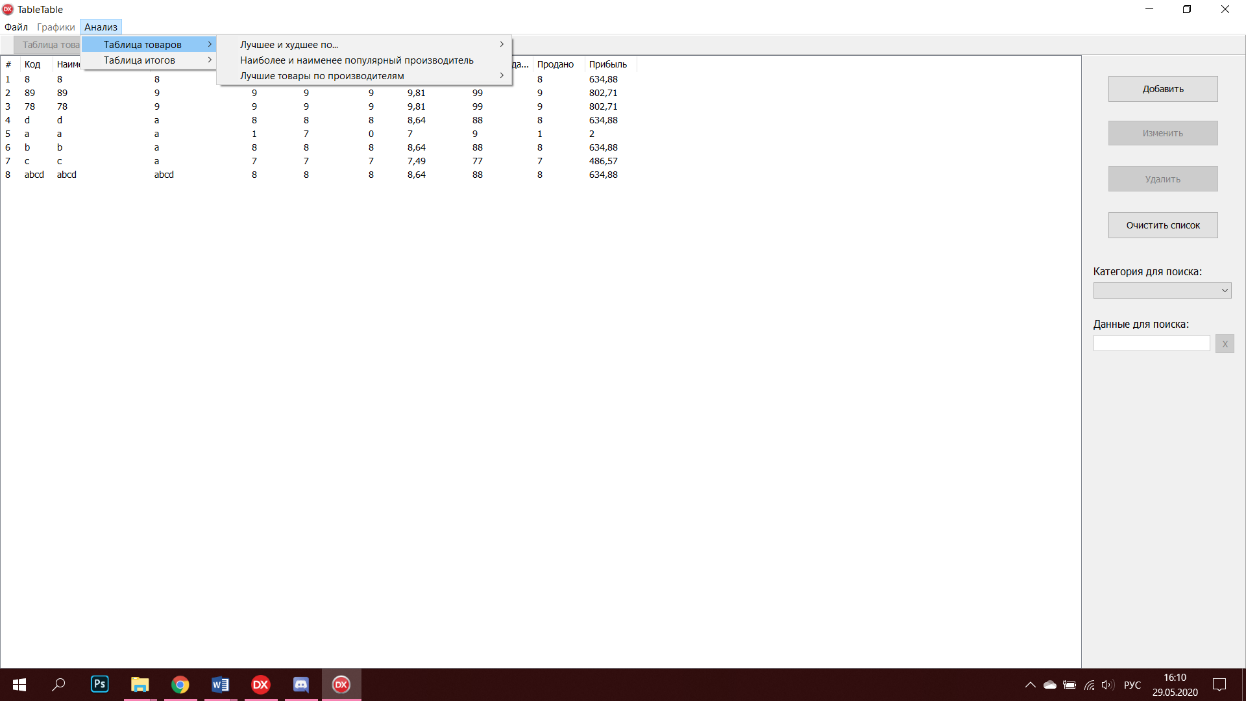
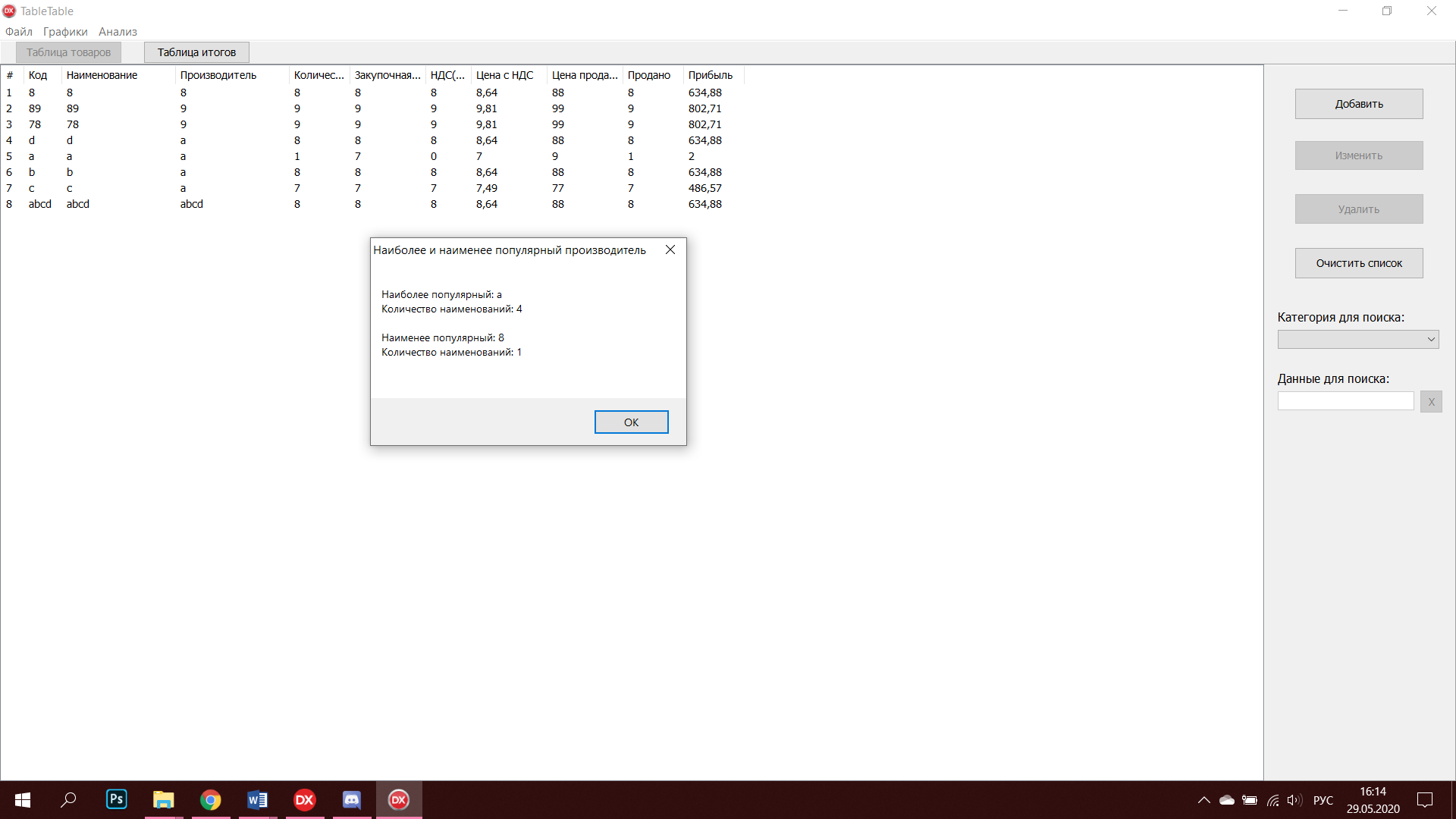
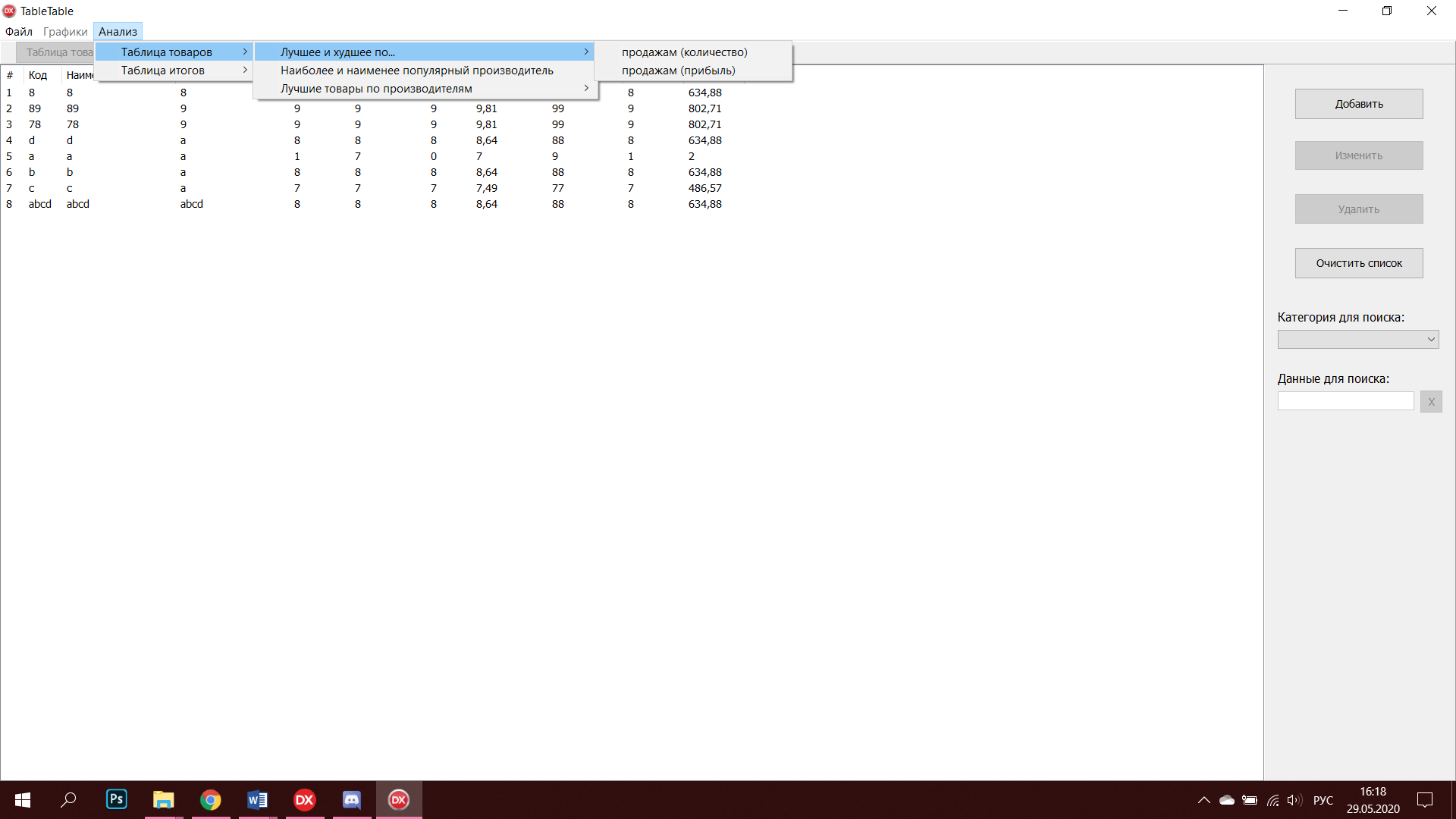
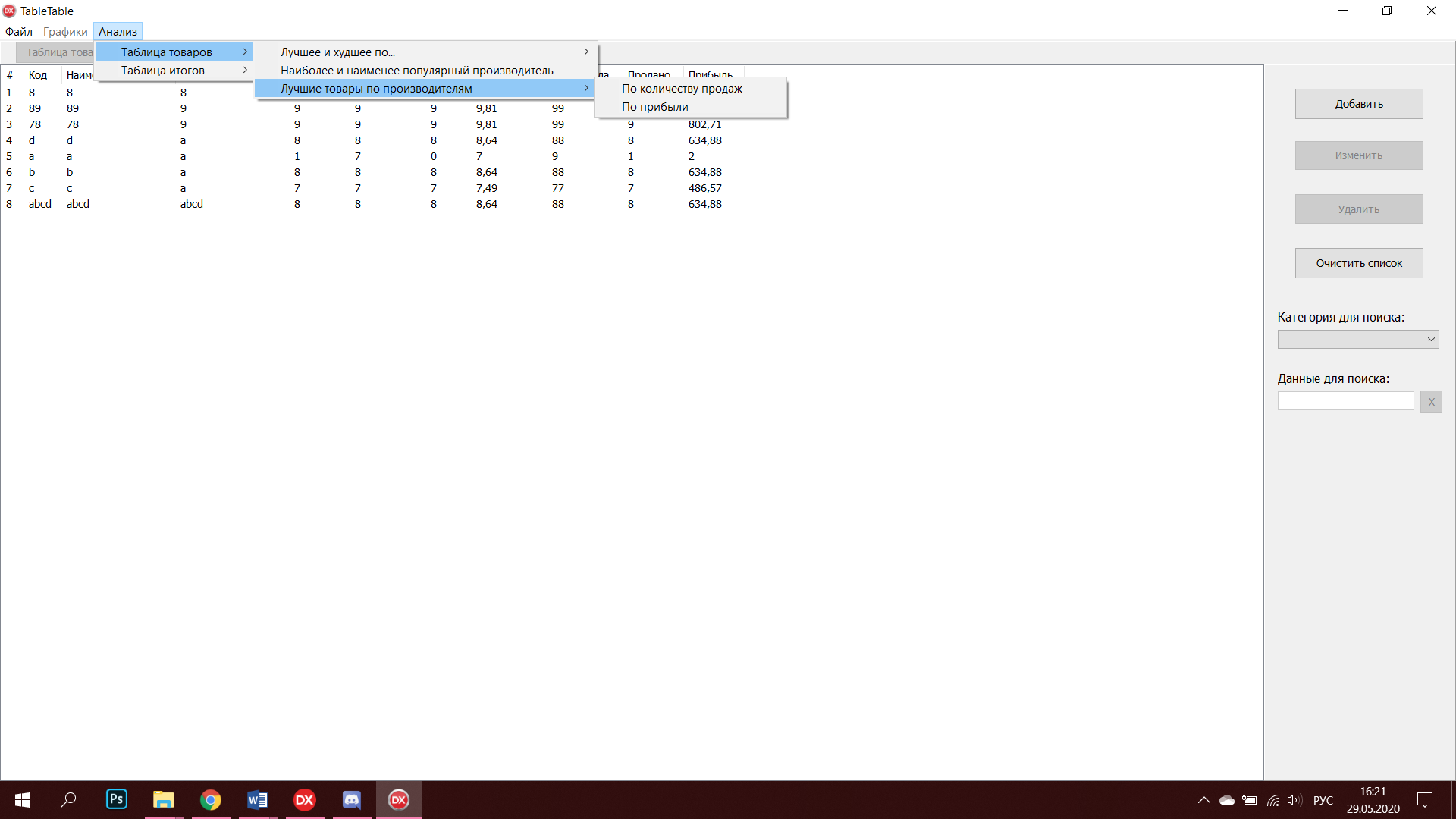
Рис. 6.11 – Результат построения графика с прогнозом

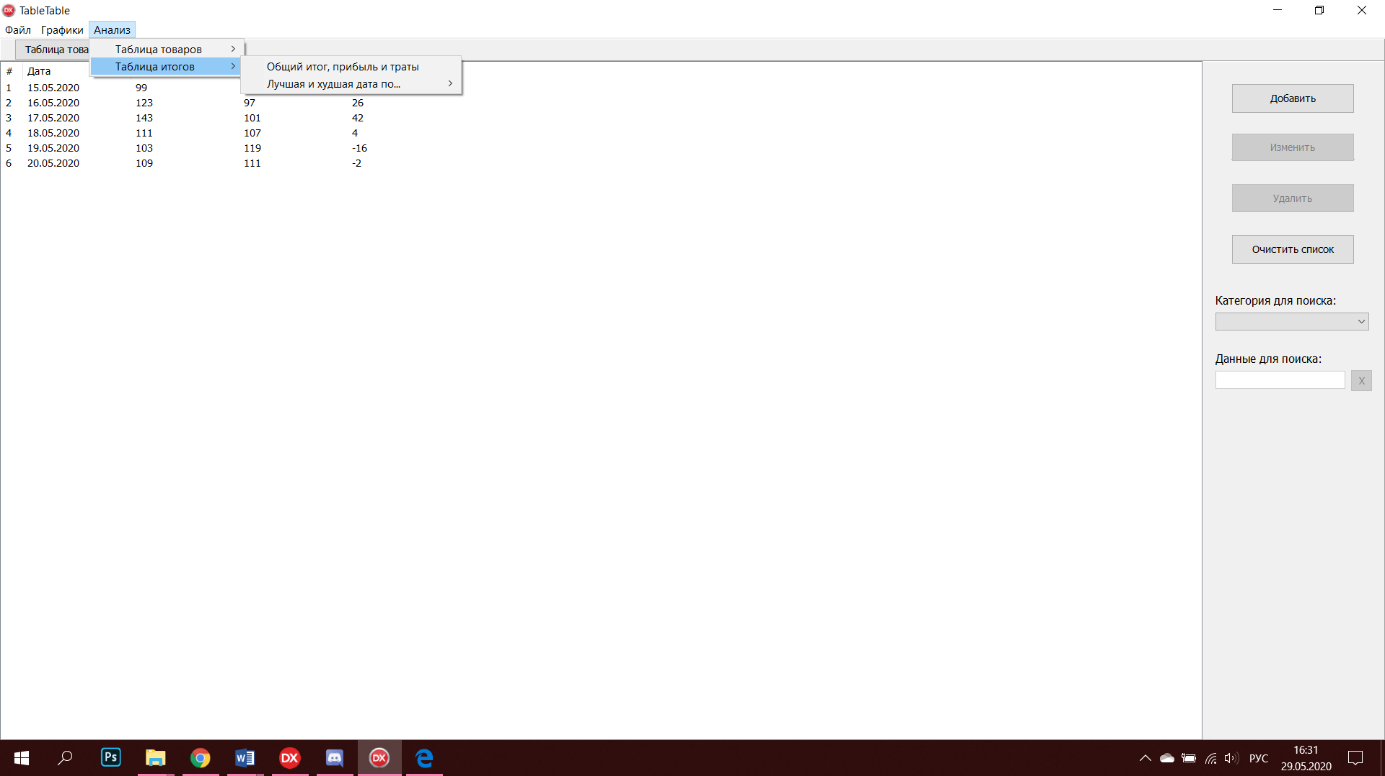
Рис. 6.12 – Доступ к возможностям анализа таблицы товаров

При выборе пункта «Общий итог, прибыль и траты» будет предоставлена информация по данному пункту в виде информирующего сообщения.

При наведении на пункт «Лучшая и худшая дата по…» появится дополнительный список для выбора (см рис. 6.16).

Рис. 6.13 – Информирующее сообщение

Рис. 6.14 – Дополнительный список «Лучшее и худшее по…»

Рис. 6.15 – Дополнительный список «Лучшие товары по производителям»

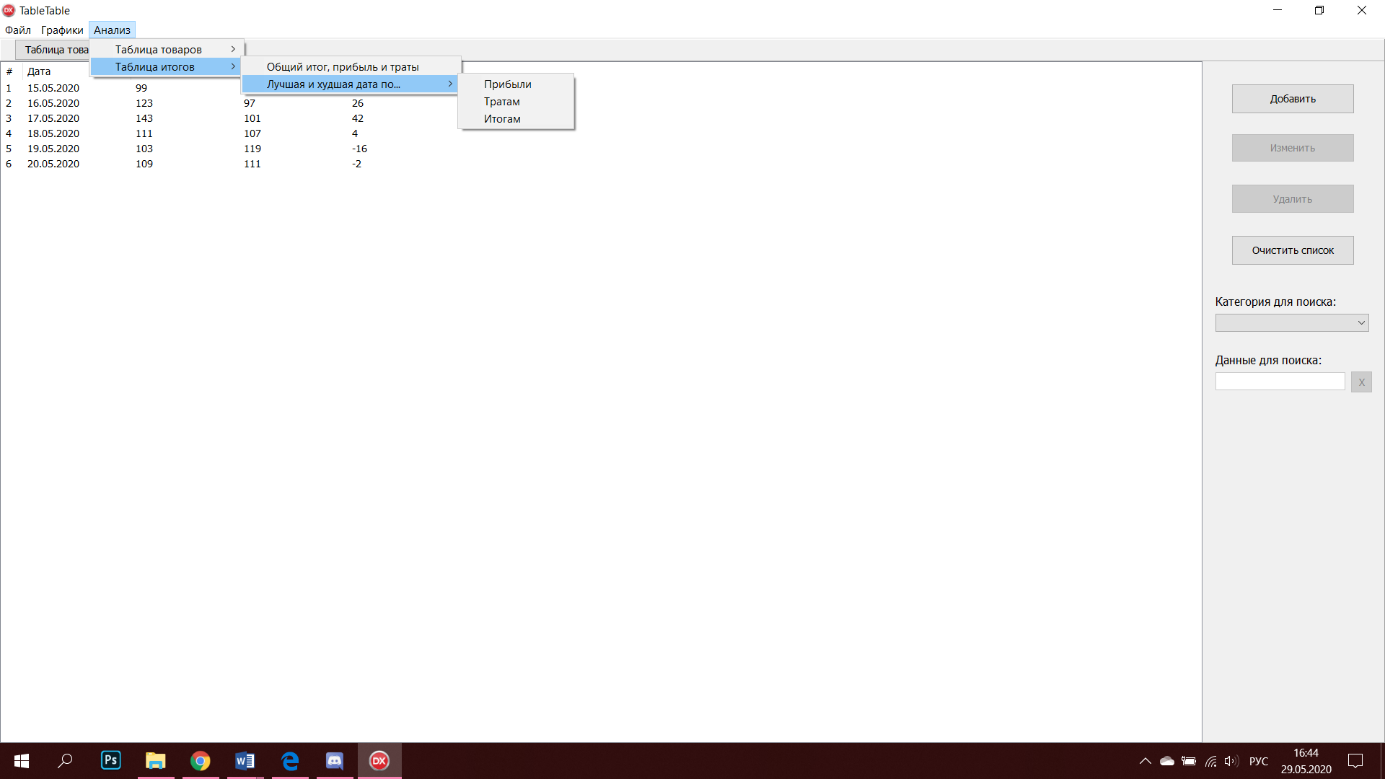
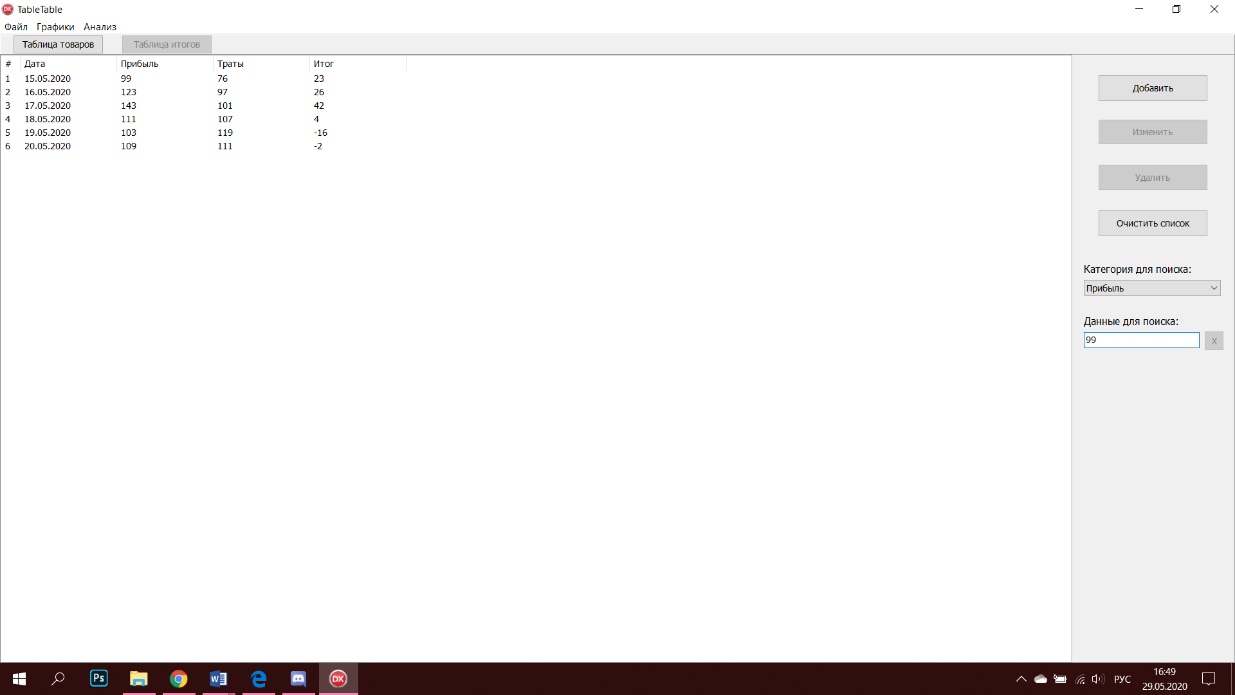
Рис. 6.16 - Доступ к возможностям анализа таблицы итогов

Рис. 6.17 - Дополнительный список «Лучшая и худшая дата по…»

Для выполнения поиска по таблице необходимо выбрать колонку таблицы для поиска среди предлагаемых, используя комбинированный список, а после ввести искомый текст (см. рис. 6.17).

Ввод искомого текста необходимо завершить нажатием Enter для начала поиска. Результат поиска будет выведен в таблицу (см. рис. 6.18).

Для отмены отображения результатов поиска необходимо нажать на кнопку «Х» справа от поля ввода искомого текста.

 Для выполнения сортировка необходимо нажать мышью на заголовок колонки таблицы. В зависимости от нажатого заголовка будет выполнена сортировка в таблице по той или иной информации.

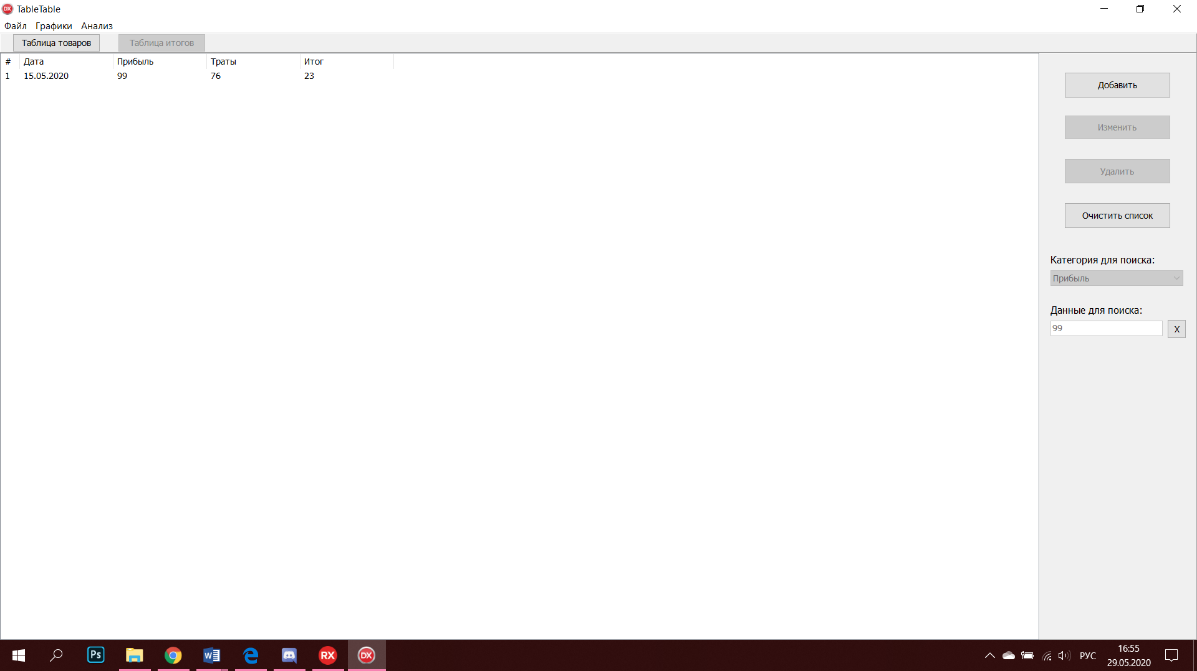
Рис. 6.18 – Подготовка к поиску

Рис. 6.19 – Результат выполнения поиска

# **ЗАКЛЮЧЕНИЕ**

В ходе разработки была изучена предметная область темы курсовой работы, подробно рассмотрены некоторые существующие методы прогнозирования и создана их реализация в рамках программного средства. Также были реализованы и иные возможности анализа данных, связанных с продажами и товарами.

После создания программное средство было тщательно протестированно с целью проверки его работоспособности. Так как результаты тестирования были удовлетворительными, можно считать, что в результате выполнения курсового проектирования было создано работоспособное программное средство, позволяющее выполнять все поставленные задачи.

С целью обеспечения пользователя информацией об использовании возможностей разработанной программы было создано руководство, описывающее необходимые взаимодействия пользователя с интерфейсом для доступа к той или иной возможности доступной в программе. Таким образом разработанное программное средство может быть беспрепятственно использовано в рамках его области и возможностей.

# **СПИСОК ЛИТЕРАТУРЫ**

[1] Былина, В. Ю. Выбор наилучшего программного обеспечения прогнозирования продаж / В. Ю. Былина // Вестник Белорусского государственного экономического университета. - 2012. - № 5. - С. 29-35

[2] Прогнозирование на основе метода экспоненциального сглаживания [Электронный ресурс]. – Режим доступа: <http://www.ekonomika-st.ru/drugie/metodi/metodi-prognoz-1-4.html>

[3] Разработка прогноза с помощью метода скользящей средней [Электронный ресурс]. – Режим доступа: <http://www.ekonomika-st.ru/drugie/metodi/metodi-prognoz-1-3.html>

[4] Разработка прогноза с помощью метода наименьших квадратов [Электронный ресурс]. – Режим доступа: <http://www.ekonomika-st.ru/drugie/metodi/metodi-prognoz-1-5.html>

# **ПРИЛОЖЕНИЕ 1**

# **ПРИЛОЖЕНИЕ 2**

unit MainUnit;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, System.Actions, Vcl.ActnList, Vcl.Menus,

Vcl.ToolWin, Vcl.ComCtrls, Vcl.StdCtrls, Vcl.ActnMan, Vcl.ActnCtrls,

System.ImageList, Vcl.ImgList, Vcl.ExtCtrls, Vcl.Mask, Vcl.Buttons, comObj,

System.Win.TaskbarCore, Vcl.Taskbar, Vcl.ButtonGroup;

type

TReportPosRec = Record

Code: string[30];

Name, Prod: string[50];

amount: integer;

price: real;

VAT: integer;

VatPrice, PriceSold: real;

AmountSold: integer;

income: real;

End;

TReportRec = Record

Date: TDate;

Income, Expense, Total: real;

end;

TReportPos = array of TReportPosRec;

TReport = array of TReportRec;

TFilePos = file of TReportPosRec;

TFileTotal = file of TReportRec;

TForm1 = class(TForm)

MenuM: TMainMenu;

btnFile: TMenuItem;

lvReport: TListView;

PnlButtons: TPanel;

actList: TActionList;

actSave: TAction;

actSaveAll: TAction;

btnOpen: TMenuItem;

btnSave: TMenuItem;

btnSaveAs: TMenuItem;

N5: TMenuItem;

actExit: TAction;

N9: TMenuItem;

btnExit: TMenuItem;

btnAdd: TButton;

btnRemove: TButton;

btnEdit: TButton;

btnGraphic: TMenuItem;

btnAnalyze: TMenuItem;

actAdd: TAction;

actEdit: TAction;

actRemove: TAction;

actSearch: TAction;

EditMaskSrch: TMaskEdit;

ComBoxSrch: TComboBox;

labelPos: TLabel;

LabelCat: TLabel;

BtnCancelSrch: TBitBtn;

dlgOpen: TOpenDialog;

dlgSave: TSaveDialog;

btnTablePos: TMenuItem;

btnTableSales: TMenuItem;

N1: TMenuItem;

btnBWSalesAmount: TMenuItem;

btnBWSalsesIncome: TMenuItem;

N4: TMenuItem;

N7: TMenuItem;

btnBWIncome: TMenuItem;

btnBWExpense: TMenuItem;

btnBWTotal: TMenuItem;

pnlTables: TPanel;

btnTable1: TButton;

btnTable2: TButton;

btnTotalAn: TMenuItem;

btnClear: TButton;

actClear: TAction;

actGraphic: TAction;

actBWSalesAmount: TAction;

actBWSalesIncome: TAction;

actMPProd: TAction;

actTotalAn: TAction;

actBWExpense: TAction;

actBWIncome: TAction;

actBWTotal: TAction;

actSaveGraphic: TAction;

actOpen: TAction;

actMPNameProdAmount: TAction;

N2: TMenuItem;

N3: TMenuItem;

actMPNameProdIncome: TAction;

N6: TMenuItem;

procedure FormCreate(Sender: TObject);

procedure actListUpdate(Action: TBasicAction; var Handled: Boolean);

procedure actAddExecute(Sender: TObject);

procedure actEditExecute(Sender: TObject);

procedure actRemoveExecute(Sender: TObject);

procedure lvReportColumnClick(Sender: TObject; Column: TListColumn);

procedure ComBoxSrchSelect(Sender: TObject);

procedure EditMaskSrchKeyPress(Sender: TObject; var Key: Char);

procedure BtnCancelSrchClick(Sender: TObject);

procedure actOpenExecute(Sender: TObject);

procedure btnTable1Click(Sender: TObject);

procedure btnTable2Click(Sender: TObject);

procedure actClearExecute(Sender: TObject);

procedure actTotalAnExecute(Sender: TObject);

procedure actBWIncomeExecute(Sender: TObject);

procedure actBWSalesAmountExecute(Sender: TObject);

procedure actBWSalesIncomeExecute(Sender: TObject);

procedure actBWExpenseExecute(Sender: TObject);

procedure actBWTotalExecute(Sender: TObject);

procedure actGraphicExecute(Sender: TObject);

procedure actSaveGraphicExecute(Sender: TObject);

procedure actSaveExecute(Sender: TObject);

procedure actSaveAllExecute(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

procedure actExitExecute(Sender: TObject);

procedure actMPProdExecute(Sender: TObject);

procedure actMPNameProdAmountExecute(Sender: TObject);

procedure actMPNameProdIncomeExecute(Sender: TObject);

private

procedure AnalyzeTable(CaseToAnalyze:byte);

procedure AddTotal(StartDate, EndDate: TDate);

procedure AddPosition(const Code, Name, Prod: string; Amount: integer;

price: Real; VAT: integer; PriceSold: real; AmountSold:integer);

procedure EditTotal(i: integer; Income, Expense: real);

procedure EditPosition(I: integer; const Code, Name, Prod: string; Amount: integer;

price: Real; VAT: integer; PriceSold:real; amountSold:integer);

procedure RemovePosition(I: integer);

procedure UpdateView(ReportUpdate: TReportPos; ReportUpdate2: TReport);

procedure Search(text: string);

function SearchPosSetItem(index: integer):string;

function SearchTotalSetItem(index: integer):string;

procedure SortModel(column: integer;var UpdateReport: TReportPos);

function CountXSort(left, right, ColumnNum: integer;UpdateReport: TReportPos): variant;

function Check1Sort(i: integer; x: variant; ColumnNum:integer;UpdateReport: TReportPos): boolean;

function Check2Sort(j: integer; x: variant; ColumnNum:integer;UpdateReport: TReportPos): boolean;

function Check3Sort(i, j, ColumnNum:integer;UpdateReport: TReportPos): boolean;

procedure MostPopularProd;

procedure MostPopularProdSearch(BuferReport: TReportPos);

procedure MostPopularNameProdAmount(BuferReport: TReportPos);

procedure MostPopularNameProdIncome(BuferReport: TReportPos);

public

FileInTxtPos, FileInTxtTotal: TextFile;

FileInPos : TFilePos;

FileInTotal : TFileTotal;

IsMonth: boolean;

ReportPos: TReportPos;

ReportTotal: TReport;

{ Public declarations }

end;

var

Form1: TForm1;

implementation

uses AddForm, Graphic;

{$R \*.dfm}

procedure TForm1.actListUpdate(Action: TBasicAction; var Handled: Boolean);

begin // обновление доступности некоторых действий

actEdit.Enabled:=(lvReport.ItemIndex<>-1)and not (BtnCancelSrch.Enabled);

actRemove.Enabled:=(lvReport.ItemIndex<>-1)and not (BtnCancelSrch.Enabled);

if lvReport.Tag=0 then

begin

actAdd.Enabled:=True;

actClear.Enabled:=(length(ReportPos)>0);

end

else

actClear.Enabled:=(length(ReportTotal)>0);

actBWSalesAmount.Enabled:=(High(ReportPos)>0);

actBWSalesIncome.Enabled:=(High(ReportPos)>0);

actBWIncome.Enabled:=(High(ReportTotal)>0);

actBWExpense.Enabled:=(High(ReportTotal)>0);

actBWTotal.Enabled:=(High(ReportTotal)>0);

actMPProd.Enabled:=(High(ReportPos)>=0);

actMPNameProdAmount.Enabled:=(High(ReportPos)>=0);

actMPNameProdIncome.Enabled:=(High(ReportPos)>=0);

actTotalAn.Enabled:=(High(ReportTotal)>=0);

actGraphic.Enabled:=(High(ReportTotal)>1);

end;

procedure TForm1.actMPNameProdAmountExecute(Sender: TObject);

var BuferReport: TReportPos;

begin //определение самого популярного наименования по производителям

//по количеству (с подготовкой)

lvReport.Selected:=nil;

BuferReport:=Copy(ReportPos);

SortModel(3, BuferReport);

MostPopularNameProdAmount(BuferReport);

BuferReport:=nil;

end;

procedure TForm1.actMPNameProdIncomeExecute(Sender: TObject);

var BuferReport: TReportPos;

begin //определение самого популярного наименования по производителям

//по прибыли (с подготовкой)

lvReport.Selected:=nil;

BuferReport:=Copy(ReportPos);

SortModel(3, BuferReport);

MostPopularNameProdIncome(BuferReport);

BuferReport:=nil;

end;

procedure TForm1.actMPProdExecute(Sender: TObject);

begin

lvReport.Selected:=nil;

MostPopularProd;

end;

procedure TForm1.actOpenExecute(Sender: TObject);

var i: integer;

begin //открытие файла

with DlgOpen do

begin

FileName:='';

if lvReport.Tag=0 then

begin

if Execute then

begin

if FilterIndex=1 then

begin

AssignFile(FileInPos, FileName);

Reset(FileInPos);

i:=0;

while not Eof(FileInPos) do

begin

SetLength(ReportPos, i+1);

seek(FileInPos, i);

read(FileInPos, ReportPos[i]);

inc(i);

end;

closeFile(FileInPos);

end;

UpdateView(ReportPos, nil);

end;

end

else

begin

if Execute then

begin

if FilterIndex=1 then

begin

AssignFile(FileInTotal, FileName);

Reset(FileInTotal);

i:=0;

while not Eof(FileInTotal) do

begin

SetLength(ReportTotal, i+1);

seek(FileInTotal, i);

read(FileInTotal, ReportTotal[i]);

inc(i);

end;

closeFile(FileInTotal);

end;

UpdateView(nil, ReportTotal);

end;

end;

end;

end;

procedure TForm1.AddPosition(const Code: string; const Name: string; const Prod: string;

Amount: Integer; price: Real; VAT: Integer; PriceSold: real; AmountSold: Integer);

var i: integer;

begin // внесение добавления в массив данных о товарах

i:= length(ReportPos);

SetLength(ReportPos, i+1);

ReportPos[i].Code := code;

ReportPos[i].Name := name;

ReportPos[i].Prod := Prod;

ReportPos[i].amount := Amount;

ReportPos[i].price := price;

ReportPos[i].VAT := VAT;

ReportPos[i].VatPrice := Round(price\*(100+Vat)) / 100;

ReportPos[i].PriceSold := ROund(PriceSold\*100)/100;

ReportPos[i].AmountSold:= AmountSold;

ReportPos[i].income := (ReportPos[i].PriceSold-ReportPos[i].VatPrice) \* AmountSold;

end;

procedure TForm1.AddTotal(StartDate, EndDate: TDate);

var i: integer;

CurDate: TDate;

begin // внесение добавления в массив данных о периодах

i:=0;

CurDate:=StartDate;

if NOT IsMonth then

begin

SetLength(ReportTotal, (Trunc(EndDate)-Trunc(StartDate)+1));

while CurDate<=EndDate do

begin

ReportTotal[i].Date := CurDate;

ReportTotal[i].Income := 0.00;

ReportTotal[i].Expense := 0.00;

ReportTotal[i].Total := 0.00;

CurDate:=CurDate+1;

inc(i);

end;

end

else

begin

while CurDate<=EndDate do

begin

SetLength(ReportTotal, i+1);

ReportTotal[i].Date := CurDate;

ReportTotal[i].Income := 0.00;

ReportTotal[i].Expense := 0.00;

ReportTotal[i].Total := 0.00;

CurDate:=incMonth(CurDate);

inc(i);

end;

end;

end;

procedure TForm1.AnalyzeTable(CaseToAnalyze: byte);

var i, BestIndex, WorstIndex:integer;

Best, Worst: variant;

begin

case CaseToAnalyze of

1:

begin //определение худшего и лучшего товара по количеству продаж

Best:=ReportPos[0].AmountSold;

Worst:=ReportPos[0].AmountSold;

BestIndex:=0; WorstIndex:=0;

for i := low(ReportPos)+1 to High(ReportPos) do

begin

if ReportPos[i].AmountSold>Best then

begin

Best:=ReportPos[i].AmountSold;

BestIndex:=i;

end;

if ReportPos[i].AmountSold<Worst then

begin

Worst:=ReportPos[i].AmountSold;

WorstIndex:=i;

end;

end;

MessageBox(Handle, PWideChar('Худший по количеству продаж: '+ '#' + IntToStr(WorstIndex+1) +

': ' +ReportPos[Worstindex].Name+#13+'Количество: '+IntToStr(ReportPos[Worstindex].AmountSold)

+#13 + #13 + 'Лучший по количеству продаж: '+ '#'+ IntToStr(BestIndex+1)+ ': '

+ReportPos[bestindex].Name+#13+ 'Количество: ' + IntToStr(ReportPos[bestindex].AmountSold)

+#13+#13), 'Результат', MB\_OK);

end;

2:

begin //определение худшего и лучшего товара по прибыли

Best:=ReportPos[0].Income;

Worst:=ReportPos[0].Income;

BestIndex:=0; WorstIndex:=0;

for i := low(ReportPos)+1 to High(ReportPos) do

begin

if ReportPos[i].Income>Best then

begin

Best:=ReportPos[i].Income;

BestIndex:=i;

end;

if ReportPos[i].Income<Worst then

begin

Worst:=ReportPos[i].Income;

WorstIndex:=i;

end;

end;

MessageBox(Handle, PWideChar('Худший по прибыли: '+ '#' + IntToStr(WorstIndex+1) +

': ' +ReportPos[Worstindex].Name+#13+'Количество: '+FLoatToStr(ReportPos[Worstindex].income)

+#13 + #13 + 'Лучший по прибыли: '+ '#'+ IntToStr(BestIndex+1)+ ': '

+ReportPos[bestindex].Name+#13+ 'Количество: ' + FLoatToStr(ReportPos[bestindex].income)

+#13+#13), 'Результат', MB\_OK);

end;

3:

begin //определение худшего и лучшего периода по прибыли

Best:=ReportTotal[0].Income;

Worst:=ReportTotal[0].Income;

BestIndex:=0; WorstIndex:=0;

for i := Low(ReportTotal)+1 to High(ReportTotal) do

begin

if ReportTotal[i].Income>Best then

begin

Best:=ReportTotal[i].Income;

BestIndex:=i;

end;

if (ReportTotal[i].Income<Worst) and (ReportTotal[i].Income<>Best) then

begin

Worst:=ReportTotal[i].Income;

WorstIndex:=i;

end;

end;

MessageBox(Handle, PWideChar('Худшая дата по прибыли: '+ DateToStr(ReportTotal[Worstindex].dATe)

+#13+'Прибыль: ' + FloatToStr(ReportTotal[Worstindex].Income) + #13 +#13 +

'Лучшая дата по прибыли: ' +DateTOStr(ReportTotal[bestindex].Date)+ #13

+ 'Прибыль: ' + FloatToStr(ReportTotal[bestindex].Income)+#13+#13 ), 'Результат', MB\_OK);

end;

4:

begin //определение худшего и лучшего периода по тратам

Best:=ReportTotal[0].expense;

Worst:=ReportTotal[0].expense;

BestIndex:=0; WorstIndex:=0;

for i := Low(ReportTotal)+1 to High(ReportTotal) do

begin

if ReportTotal[i].expense>Worst then

begin

Worst:=ReportTotal[i].expense;

WorstIndex:=i;

end;

if (ReportTotal[i].expense<Best) and (ReportTotal[i].expense<>Worst) then

begin

Best:=ReportTotal[i].expense;

BestIndex:=i;

end;

end;

MessageBox(Handle, PWideChar('Худшая дата по тратам: '+ DateToStr(ReportTotal[Worstindex].dATe)

+#13+'Траты: ' + FloatToStr(ReportTotal[Worstindex].Expense) + #13 +#13 +

'Лучшая дата по тратам: ' +DateTOStr(ReportTotal[bestindex].Date) + #13

+ 'Траты: ' + FloatToStr(ReportTotal[bestindex].Expense)+#13+#13 ), 'Результат', MB\_OK);

end;

5: //определение худшего и лучшего периода по итогам

begin

Best:=ReportTotal[0].total;

Worst:=ReportTotal[0].total;

BestIndex:=0; WorstIndex:=0;

for i := Low(ReportTotal)+1 to High(ReportTotal) do

begin

if ReportTotal[i].total>Worst then

begin

Best:=ReportTotal[i].total;

BestIndex:=i;

end;

if (ReportTotal[i].total<Worst) and (ReportTotal[i].total<>Best) then

begin

Worst:=ReportTotal[i].total;

WorstIndex:=i;

end;

end;

MessageBox(Handle, PWideChar('Худшая дата по итогу: '+ DateToStr(ReportTotal[Worstindex].dATe)

+#13+'Итог: ' + FloatToStr(ReportTotal[Worstindex].Total) + #13 +#13 +

'Лучшая дата по итогам: ' +DateTOStr(ReportTotal[bestindex].Date) + #13

+ 'Итог: ' + FloatToStr(ReportTotal[bestindex].Total)+#13+#13 ), 'Результат', MB\_OK);

end;

end;

end;

procedure TForm1.BtnCancelSrchClick(Sender: TObject);

begin

if lvReport.Tag=0 then

UpdateView(ReportPos, nil)

else

UpdateView(nil, ReportTotal);

EditMAskSrch.text:='';

EditMaskSrch.Enabled:=True;

ComBoxSrch.Enabled:=True;

BtnCancelSrch.Enabled:=False;

end;

procedure TForm1.btnTable1Click(Sender: TObject);

var i: integer;

begin //переключение на таблицу товаров

lvReport.Tag:=0;

with lvReport do

begin

Clear;

Columns.Clear;

Columns.Add; Columns[0].Caption:='#'; Column[0].Width:=30;

Columns.Add; Columns[1].Caption:='Код'; Column[1].Width:=50;

Columns.Add; Columns[2].Caption:='Наименование'; Column[2].Width:=150;

Columns.Add; Columns[3].Caption:='Производитель'; Column[3].Width:=150;

Columns.Add; Columns[4].Caption:='Количество'; Column[4].Width:=80;

Columns.Add; Columns[5].Caption:='Закупочная цена'; Column[5].Width:=100;

Columns.Add; Columns[6].Caption:='НДС(%)'; Column[6].Width:=60;

Columns.Add; Columns[7].Caption:='Цена с НДС'; Column[7].Width:=100;

Columns.Add; Columns[8].Caption:='Цена продажи'; Column[8].Width:=100;

Columns.Add; Columns[9].Caption:='Продано'; Column[9].Width:=80;

Columns.Add; Columns[10].Caption:='Прибыль'; Column[10].Width:=80;

end;

ComBoxSrch.Clear;

comboxsrch.Items.Add('');

for i := 1 to lvReport.Columns.Count-1 do

comboxsrch.Items.Add(lvReport.Column[i].caption);

EditMaskSrch.Enabled:=False;

UpdateView(ReportPos, nil);

btnTable1.Enabled:=False;

btnTable2.Enabled:=True;

end;

procedure TForm1.btnTable2Click(Sender: TObject);

var i: integer;

begin //переключение на таблицу итогов

lvReport.Tag:=1;

with lvReport do

begin

Clear;

Columns.Clear;

Columns.Add; Columns[0].Caption:='#'; Column[0].Width:=30;

Columns.Add; Columns[1].Caption:='Дата'; Column[1].Width:=150;

Columns.Add; Columns[2].Caption:='Прибыль'; Column[2].Width:=150;

Columns.Add; Columns[3].Caption:='Траты'; Column[3].Width:=150;

Columns.Add; Columns[4].Caption:='Итог'; Column[4].Width:=150;

end;

ComBoxSrch.Clear;

comboxsrch.Items.Add('');

for i := 1 to lvReport.Columns.Count-1 do

comboxsrch.Items.Add(lvReport.Column[i].caption);

EditMaskSrch.Enabled:=False;

UpdateView(nil, ReportTotal);

btnTable2.Enabled:=False;

btnTable1.Enabled:=True;

end;

procedure TForm1.actAddExecute(Sender: TObject);

var Res: TModalResult;

index: integer;

begin // добавление (включая вызов формы для добавления)

if (not Assigned(Form2)) then

Form2:=TForm2.Create(Self);

if lvReport.Tag=0 then

begin

Res:= Form2.AddNewPosition;

if Res<>mrOk then Exit;

with Form2 do

AddPosition(Code, Name, Prod, Amount, price, VAT, PriceSold, AmountSold);

UpdateView(ReportPos, nil);

end

else

begin

if length(ReportTotal)>0 then

begin

if BtnCancelSrch.Enabled then

BtnCancelSrchClick(nil);

SetLength(ReportTotal, length(ReportTotal)+1);

index:=High(ReportTotal);

if IsMonth then

ReportTotal[index].Date:=incMonth(ReportTotal[index-1].Date)

else

ReportTotal[index].Date:=ReportTotal[index-1].Date+1;

ReportTotal[index].Income:=0;

ReportTotal[index].Expense:=0;

ReportTotal[index].Total:=0;

end

else

begin

Res:= Form2.AddNewTotal;

if Res<>mrOk then Exit;

IsMonth:=Form2.IsMonth;

with Form2 do

AddTotal(StartDate, EndDate);

end;

UpdateView(nil, ReportTotal);

end;

end;

function TForm1.SearchPosSetItem(index: integer):string;

begin

if ComBoxSrch.Text=lvReport.Column[1].Caption then

result:=ReportPos[index].Code

else

begin

if ComBoxSrch.Text=lvReport.Column[2].Caption then

result:=ReportPos[index].Name

else

begin

if ComBoxSrch.Text=lvReport.Column[3].Caption then

result:=ReportPos[index].Prod

else

begin

if ComBoxSrch.Text=lvReport.Column[4].Caption then

result:=IntToStr(ReportPos[index].amount)

else

begin

if ComBoxSrch.Text=lvReport.Column[5].Caption then

result:=FloatToStr(ReportPos[index].price)

else

begin

if ComBoxSrch.Text=lvReport.Column[6].Caption then

result:=IntToStr(ReportPos[index].VAT)

else

begin

if ComBoxSrch.Text=lvReport.Column[7].Caption then

result:=FloatToStr(ReportPos[index].VatPrice)

else

begin

if ComBoxSrch.Text=lvReport.Column[8].Caption then

result:=FloatToStr(ReportPos[index].PriceSold)

else

begin

if ComBoxSrch.Text=lvReport.Column[9].Caption then

result:=IntToStr(ReportPos[index].AmountSold)

else

if ComBoxSrch.Text=lvReport.Column[10].Caption then

result:=FloatToStr(ReportPos[index].income);

end;

end;

end;

end;

end;

end;

end;

end;

end;

function TForm1.SearchTotalSetItem(index: integer): string;

begin

if ComBoxSrch.Text=lvReport.Column[1].Caption then

result:=DateToStr(ReportTotal[index].Date)

else

begin

if ComBoxSrch.Text=lvReport.Column[2].Caption then

result:=FloatToStr(ReportTotal[index].Income)

else

begin

if ComBoxSrch.Text=lvReport.Column[3].Caption then

result:=FloatToStr(ReportTotal[index].Expense)

else

begin

if ComBoxSrch.Text=lvReport.Column[4].Caption then

result:=FloatToStr(ReportTotal[index].Total)

end;

end;

end;

end;

procedure TForm1.SortModel(column: integer;var UpdateReport: TReportPos);

var i, j: integer;

x: variant;

y: TReportPosRec;

Procedure Sort(left, Right: integer); //быстрая сортировка

begin

i:=left; j:=right;

x:=CountXSort(left, right,column,UpdateReport);

repeat

while Check1Sort(i, x, column,UpdateReport) do

inc(i);

while Check2Sort(j, x, column,UpdateReport) do

dec(j);

if i<=j then

begin

if Check3Sort(i, j, column,UpdateReport) then

begin

y:=UpdateReport[i];

UpdateReport[i]:=UpdateReport[j];

UpdateReport[j]:=y;

end;

inc(i);

dec(j);

end;

until i>=j;

if left<j then

Sort(left, j);

if i<right then

Sort(i, right);

end;

begin

Sort(Low(UpdateReport), High(UpdateReport));

end;

procedure TForm1.Search(text: string);

var BuferReport:TReportPos;

BuferReport2:TReport;

i: integer;

begin // поиск

if lvReport.Tag=0 then

begin

SetLength(BuferReport, 0);

for i := 0 to High(ReportPos) do

if Pos(text, SearchPosSetItem(i))>0 then

begin

SetLength(BuferReport, length(buferReport)+1);

BuferReport[High(BuferReport)]:=ReportPos[i]

end;

if length(BuferReport)>0 then

begin

UpdateView(BuferReport, nil);

EditMaskSrch.enabled:=False;

ComBoxSrch.Enabled:=False;

BtnCancelSrch.Enabled:=True;

end

else

MessageBox(Handle, 'Позиция не найдена', 'Уведомление', MB\_OK or MB\_ICONQUESTION);

end

else

begin

SetLength(BuferReport2,0);

for i := 0 to High(ReportTotal) do

if Pos(text, SearchTotalSetItem(i))>0 then

begin

SetLength(BuferReport2, length(buferReport2)+1);

BuferReport2[High(BuferReport2)]:=ReportTotal[i];

end;

if length(BuferReport2)>0 then

begin

UpdateView(nil, BuferReport2);

EditMaskSrch.enabled:=False;

ComBoxSrch.Enabled:=False;

BtnCancelSrch.Enabled:=True;

end

else

MessageBox(Handle, 'Позиция не найдена', 'Уведомление', MB\_OK or MB\_ICONQUESTION);

end;

end;

procedure TForm1.EditMaskSrchKeyPress(Sender: TObject; var Key: Char);

begin // обработка нажатия Enter в поле ввода для поиска

if key = ^M then

begin

Search(editmaskSrch.Text);

key:=#0;

end;

end;

procedure TForm1.EditPosition(I: integer; const Code, Name, Prod: string;

Amount: integer; price: Real; VAT:integer; PriceSold:real; AmountSold: integer);

begin // Внесение изменений в массив товаров

ReportPos[i].Code := code;

ReportPos[i].Name := name;

ReportPos[i].Prod := Prod;

ReportPos[i].amount := Amount;

ReportPos[i].price := price;

ReportPos[i].VAT := VAT;

ReportPos[i].VatPrice := price\*(100+Vat) / 100;

ReportPos[i].PriceSold := PriceSold;

ReportPos[i].AmountSold:= AmountSold;

ReportPos[i].income := (ReportPos[i].PriceSold-ReportPos[i].VatPrice) \* AmountSold;

end;

procedure TForm1.EditTotal(i: integer;Income, Expense: real);

begin // Внесение изменений в массив периодов

ReportTotal[i].Income := Income;

ReportTotal[i].Expense := Expense;

ReportTotal[i].Total := Round((income-Expense)\*1000)/1000;

end;

procedure TForm1.actRemoveExecute(Sender: TObject);

var i: integer;

begin //удаление

i:=lvReport.ItemIndex;

RemovePosition(i);

if lvReport.Tag=0 then

UpdateView(ReportPos, nil)

else

UpdateView(nil, ReportTotal);

if i<lvReport.Items.Count then

lvReport.ItemIndex:=i

else

lvReport.ItemIndex:=i-1;

end;

procedure TForm1.actSaveAllExecute(Sender: TObject);

var i: integer;

begin // сохранение обоих массивов данных

with dlgSave do

begin

if Length(ReportPos)>0 then

begin

FileName:='Таблица товаров';

if Execute then

begin

if FilterIndex=1 then

begin

AssignFile(FileInPos, FileName);

Rewrite(FileInPos);

for i := Low(ReportPos) to High(ReportPos) do

write(FileInPos, ReportPos[i]);

CloseFile(FileInPos);

end;

if FilterIndex=2 then

begin

AssignFile(FileInTxtPos, FileName);

Rewrite(FileInTxtPos);

writeln(FileInTxtPos,'|','Код':30,'|','Наименование':50,'|','Производитель':50,'|','Количество','|'

,'Цена':10,'|','НДС','|','Цена С НДС':11,'|','Цена продажи':12,'|','Количество проданых':20,'|',

'Прибыль':14,'|');

for i := Low(ReportPos) to High(ReportPos) do

begin

with ReportPos[i] do

writeln(FileInTxtPos, '|',Code:30,'|',Name:50,'|',Prod:50,'|',amount:10,'|',price:10:3

,'|',VAT:3,'|',VatPrice:11,'|',PriceSold:12:3,'|',AmountSold:20,'|',income:14:3,'|');

end;

CloseFile(FileInTxtPos);

end;

end;

end

else

MessageBox(Handle, 'Таблица товаров пуста, сохранение невозможно', 'Сообщение', MB\_OK or MB\_ICONWARNING);

if length(ReportTotal)>0 then

begin

FileName:='Таблица итогов';

if Execute then

begin

if FilterIndex=1 then

begin

AssignFile(FileInTotal, FileName);

Rewrite(FileInTotal);

for i := Low(ReportTotal) to High(ReportTotal) do

write(FileInTotal, ReportTotal[i]);

CloseFile(FileInTotal);

end;

if FilterIndex=2 then

begin

AssignFile(FileInTxtTotal, FileName);

Rewrite(FileInTxtTotal);

writeln(FileInTxtTotal,'|','Дата':12,'|','Прибыль':15,'|','Траты':15,'|','Итог':15,'|');

for i := Low(ReportTotal) to High(ReportTotal) do

begin

with ReportTotal[i] do

writeln(FileInTxtTotal, '|',Date:12,'|',Income:15,'|',Expense:15,'|',Total:15,'|');

end;

CloseFile(FileInTxtPos);

end;

end;

end

else

MessageBox(Handle, 'Таблица итогов пуста, сохранение невозможно', 'Сообщение', MB\_OK or MB\_ICONWARNING);

end;

end;

procedure TForm1.actSaveExecute(Sender: TObject);

var i: integer;

begin // сохранение одного из массивов данных

with dlgSave do

begin

if lvReport.Items.Count>0 then

begin

if lvReport.Tag=0 then

begin

FileName:='Таблица товаров';

if Execute then

begin

if FilterIndex=1 then

begin

AssignFile(FileInPos, FileName);

Rewrite(FileInPos);

for i := Low(ReportPos) to High(ReportPos) do

write(FileInPos, ReportPos[i]);

CloseFile(FileInPos);

end;

if FilterIndex=2 then

begin

AssignFile(FileInTxtPos, FileName);

Rewrite(FileInTxtPos);

writeln(FileInTxtPos,'|','Код':30,'|','Наименование':50,'|','Производитель':50,'|','Количество','|'

,'Цена':10,'|','НДС','|','Цена С НДС':11,'|','Цена продажи':12,'|','Количество проданых':20,'|',

'Прибыль':14,'|');

for i := Low(ReportPos) to High(ReportPos) do

begin

with ReportPos[i] do

writeln(FileInTxtPos, '|',Code:30,'|',Name:50,'|',Prod:50,'|',amount:10,'|',price:10:3

,'|',VAT:3,'|',VatPrice:11:3,'|',PriceSold:12:3,'|',AmountSold:20,'|',income:14:3,'|');

end;

CloseFile(FileInTxtPos);

end;

end;

end

else

begin

FileName:='Таблица итогов';

if Execute then

begin

if FilterIndex=1 then

begin

AssignFile(FileInTotal, FileName);

Rewrite(FileInTotal);

for i := Low(ReportTotal) to High(ReportTotal) do

write(FileInTotal, ReportTotal[i]);

CloseFile(FileInTotal);

end;

if FilterIndex=2 then

begin

AssignFile(FileInTxtTotal, FileName);

Rewrite(FileInTxtTotal);

writeln(FileInTxtTotal,'|','Дата':12,'|','Прибыль':15,'|','Траты':15,'|','Итог':15,'|');

for i := Low(ReportTotal) to High(ReportTotal) do

begin

with ReportTotal[i] do

writeln(FileInTxtTotal, '|',Date:12,'|',Income:15:3,'|',Expense:15:3,'|',Total:15:3,'|');

end;

CloseFile(FileInTxtPos);

end;

end;

end;

end

else

MessageBox(Handle, 'Таблица пуста, сохранение невозможно', 'Сообщение', MB\_OK or MB\_ICONWARNING);

end;

end;

procedure TForm1.actSaveGraphicExecute(Sender: TObject);

begin // сохранение графика

with Form3 do

begin

with DlgSaveGraphic do

begin

FileName:='chart';

if Execute then

Image1.Picture.SaveTofile(FileName);

end;

end;

end;

procedure TForm1.actTotalAnExecute(Sender: TObject);

var Total, Income, Expense: real;

I: integer;

begin // Определение общей прибыли, трат и итогов за периоды

Income:=0; Expense:=0;

for i := Low(ReportTotal) to High(ReportTotal) do

begin

income:=income+ReportTOtal[i].Income;

expense:=expense+ReportTOtal[i].expense;

Total:=Total+ReportTOtal[i].Total;

end;

MessageBox(Handle, PWideChar('Общая прибыль за период: ' + FloatToStr(Income) + #13#13 +

'Общие траты за период: ' + FloatToStr(Expense) + #13#13 +

'Общий итог за период: ' + FloatToStr(Income-Expense) +#13#13), 'Результат', MB\_Ok);

end;

procedure TForm1.actBWSalesAmountExecute(Sender: TObject);

begin

AnalyzeTable(1);

end;

procedure TForm1.actBWSalesIncomeExecute(Sender: TObject);

begin

AnalyzeTable(2);

end;

procedure TForm1.actBWIncomeExecute(Sender: TObject);

begin

AnalyzeTable(3);

end;

procedure TForm1.actBWExpenseExecute(Sender: TObject);

begin

AnalyzeTable(4);

end;

procedure TForm1.actBWTotalExecute(Sender: TObject);

begin

AnalyzeTable(5);

end;

procedure TForm1.actClearExecute(Sender: TObject);

begin // очистка всех данных массива, который сейчас отображается

if BtnCancelSrch.Enabled then

BtnCancelSrchClick(nil);

LvReport.Clear;

if lvReport.Tag=0 then

ReportPos:=nil

else

ReportTotal:=nil;

btnAdd.Enabled:=True;

btnClear.Enabled:=False;

end;

procedure TForm1.actEditExecute(Sender: TObject);

var i: integer;

res: TModalResult;

Item: TListItem;

begin // изменение (с вызовом окна изменения)

Item:=lvReport.Selected;

i:= lvReport.ItemIndex;

if (not Assigned(Form2)) then

Form2:=TForm2.Create(Self);

if lvReport.Tag=0 then

begin

Res:= Form2.EditCurPosition(I, ReportPos[i].Code, ReportPos[i].Name, ReportPos[i].Prod,

ReportPos[i].Amount, ReportPos[i].Price, ReportPos[i].Vat,

ReportPos[i].PriceSold, ReportPos[i].AmountSold);

if Res<>mrOk then Exit;

with Form2 do

EditPosition(I, Code, Name, Prod, Amount, price, VAT, PriceSold, AmountSold);

UpdateView(ReportPos, nil);

end

else

begin

Res:= Form2.EditCurTotal(ReportTotal[i].Income, ReportTotal[i].Expense);

if Res<>mrOk then Exit;

with Form2 do

EditTotal(i, income, Expense);

UpdateView(nil, ReportTotal);

if i<lvReport.Items.Count-1 then

LvReport.ItemIndex:=i+1;

end;

end;

procedure TForm1.actExitExecute(Sender: TObject);

begin

Form1.Close;

end;

procedure TForm1.actGraphicExecute(Sender: TObject);

begin // открытие окна для анализа и прогнозирования с построением графиков

if length(ReportTotal)<=366 then

begin

if (not Assigned(Form3)) then

Form3:=TForm3.Create(Self);

form3.Show;

end

else

begin

MessageBox(Handle, 'Превышено количество позиций (366), построение графика невозможно.',

'Предупреждение', MB\_Ok or MB\_ICONWARNING);

end;

end;

procedure TForm1.FormCloseQuery(Sender: TObject; var CanClose: Boolean);

begin // обработка закрытия основного окна

canClose:=MessageBox(Handle, 'Вы точно хотите закрыть приложение (несохранённые данные могут быть утеряны)?',

'Предупреждение', MB\_OKCancel or MB\_ICONWARNING)=mrOK;

if CanClose then

begin

ReportPos:=nil;

ReportTotal:=nil;

end;

end;

procedure TForm1.FormCreate(Sender: TObject);

var i: integer;

begin

dlgOpen.InitialDir:=ExtractFilePath(Application.ExeName);

dlgSave.InitialDir:=ExtractFilePath(Application.ExeName);

form1.WindowState:=wsMaximized;

end;

procedure TForm1.RemovePosition(I: integer);

var j: integer;

begin // удаление

if lvReport.Tag=0 then

begin

for j := i to High(ReportPos)-1 do

ReportPos[j]:=ReportPos[j+1];

Setlength(ReportPos, length(ReportPos)-1);

end

else

begin

for j := i to High(ReportTotal)-1 do

ReportTotal[j]:=ReportTotal[j+1];

Setlength(ReportTotal, length(ReportTotal)-1);

end;

end;

function TForm1.Check1Sort(i: integer; x: variant; columnNum: integer; UpdateReport: TReportPos): boolean;

begin

result:=False;

case columnNum of

1:

if UpdateReport[i].Code<x then

result:=True;

2:

if UpdateReport[i].Name<x then

result:=True;

3:

if UpdateReport[i].Prod<x then

result:=True;

4:

if UpdateReport[i].amount<x then

result:=True;

5:

if UpdateReport[i].price<x then

result:=True;

6:

if UpdateReport[i].VAT<x then

result:=True;

7:

if UpdateReport[i].VatPrice<x then

result:=True;

8:

if UpdateReport[i].PriceSold<x then

result:=True;

9:

if UpdateReport[i].AmountSold<x then

result:=True;

10:

if UpdateReport[i].income<x then

result:=True;

end;

end;

function TForm1.Check2Sort(j: integer; x: variant; columnNum: integer;UpdateReport:TReportPos): boolean;

begin

result:=False;

case columnNum of

1:

if UpdateReport[j].Code>x then

result:=True;

2:

if UpdateReport[j].Name>x then

result:=True;

3:

if UpdateReport[j].Prod>x then

result:=True;

4:

if UpdateReport[j].amount>x then

result:=True;

5:

if UpdateReport[j].price>x then

result:=True;

6:

if UpdateReport[j].VAT>x then

result:=True;

7:

if UpdateReport[j].VatPrice>x then

result:=True;

8:

if UpdateReport[j].PriceSold>x then

result:=True;

9:

if UpdateReport[j].AmountSold>x then

result:=True;

10:

if UpdateReport[j].income>x then

result:=True;

end;

end;

function TForm1.Check3Sort(i, j: integer; columnNum: integer;UpdateReport:TReportPos): boolean;

begin

result:=False;

case columnNum of

1:

if UpdateReport[i].Code>UpdateReport[j].Code then

result:=True;

2:

if UpdateReport[i].Name>UpdateReport[j].Name then

result:=True;

3:

if UpdateReport[i].Prod>UpdateReport[j].Prod then

result:=True;

4:

if UpdateReport[i].amount>UpdateReport[j].amount then

result:=True;

5:

if UpdateReport[i].price>UpdateReport[j].price then

result:=True;

6:

if UpdateReport[i].VAT>UpdateReport[j].VAT then

result:=True;

7:

if UpdateReport[i].VatPrice>UpdateReport[j].VatPrice then

result:=True;

8:

if UpdateReport[i].PriceSold>UpdateReport[j].PriceSold then

result:=True;

9:

if UpdateReport[i].AmountSold>UpdateReport[j].AmountSold then

result:=True;

10:

if UpdateReport[i].income>UpdateReport[j].income then

result:=True;

end;

end;

function Tform1.CountXSort(left: Integer; right: Integer; ColumnNum: Integer;UpdateReport:TReportPos) : variant;

begin

case columnNum of

1: result:=UpdateReport[left+random(right-left+1)].Code;

2: result:=UpdateReport[left+random(right-left+1)].Name;

3: result:=UpdateReport[left+random(right-left+1)].Prod;

4: result:=UpdateReport[left+random(right-left+1)].amount;

5: result:=UpdateReport[left+random(right-left+1)].price;

6: result:=UpdateReport[left+random(right-left+1)].VAT;

7: result:=UpdateReport[left+random(right-left+1)].VatPrice;

8: result:=UpdateReport[left+random(right-left+1)].PriceSold;

9: result:=UpdateReport[left+random(right-left+1)].AmountSold;

10: result:=UpdateReport[left+random(right-left+1)].income;

end;

end;

procedure TForm1.ComBoxSrchSelect(Sender: TObject);

begin // обработка активации возможности ввода в поле ввода для поиска

// по выбору колонки для поиска

if comboxsrch.Text='' then

begin

EditMaskSrch.Enabled:=False;

EditMaskSrch.Text:='';

end

else

EditMaskSrch.Enabled:=True;

end;

procedure TForm1.lvReportColumnClick(Sender: TObject; Column: TListColumn);

begin // обработка сортировки по нажатию на колонку списка

if lvReport.tag=0 then

begin

if lvReport.Items.Count>1 then

begin

SortModel(Column.index, ReportPos);

UpdateView(ReportPos, nil);

end;

end

else

MessageBox(Handle, 'Сортировка недоступна для данной таблицы', 'Сообщение', Mb\_Ok or MB\_ICONINFORMATION);

end;

procedure TForm1.MostPopularNameProdAmount(BuferReport: TReportPos);

var BuferProd, NameBest, FinishLine: string;

AmountBest, i: integer;

begin //определение наиболее популярного наименования по производителям по кол-ву

i:=0;

FinishLine:='';

while i<=High(BuferReport) do

begin

BuferProd:=BuferReport[i].Prod;

AmountBest:=0;

while BuferProd=BuferReport[i].Prod do

begin

if BuferReport[i].amountSold>AmountBest then

begin

NameBest:=BuferReport[i].Name;

AmountBest:=BuferReport[i].amountSold;

end;

inc(i);

end;

FinishLine:=FinishLine+'Лучший товар у производителя ' + BuferProd + ': '+ NameBest + #13 +

'Количество: ' + IntToStr(AmountBest)+ #13+#13;

end;

MessageBox(Handle, PWideChar('Лучшие товары по производителям (по количеству продаж)' + #13+#13

+ FInishLine), 'Результат', MB\_Ok);

end;

procedure TForm1.MostPopularNameProdIncome(BuferReport: TReportPos);

var BuferProd, NameBest, FinishLine: string;

i: integer;

IncomeBest: real;

begin //определение наиболее популярного наименования по производителям по прибыли

i:=0;

FinishLine:='';

while i<=High(BuferReport) do

begin

BuferProd:=BuferReport[i].Prod;

IncomeBest:=0;

while BuferProd=BuferReport[i].Prod do

begin

if BuferReport[i].income>IncomeBest then

begin

NameBest:=BuferReport[i].Name;

IncomeBest:=BuferReport[i].income;

end;

inc(i);

end;

FinishLine:=FinishLine+'Лучший товар у производителя ' + BuferProd + ': '+ NameBest + #13 +

'Прибыль: ' + FloatToStr(IncomeBest)+ #13+#13;

end;

MessageBox(Handle, PWideChar('Лучшие товары по производителям (по прибыли)' + #13+#13

+ FInishLine), 'Результат', MB\_Ok);

end;

procedure TForm1.MostPopularProd;

var BuferReport: TReportPos;

begin //определение самого популярного производителя (с подготовкой)

BuferReport:=Copy(ReportPos);

SortModel(3, BuferReport);

MostPopularProdSearch(BuferReport);

BuferReport:=nil;

end;

procedure TForm1.MostPopularProdSearch(BuferReport: TReportPos);

var ProdBest, ProdCur, ProdWorst: string;

AmountBest, AmountCur, AmountWorst, i: integer;

begin //определение самого популярного производителя

i:=0;

AmountBest:=0;

ProdBest:=BuferReport[0].Prod;

while ProdBest=BuferReport[i].Prod do

begin

inc(AmountBest);

inc(i);

end;

ProdWorst:=ProdBest;

AmountWorst:=AmountBest;

while i <= High(BuferReport) do

begin

AmountCur:=0;

ProdCur:=BuferReport[i].Prod;

while (i<=High(BuferReport)) and (ProdCur=BuferReport[i].Prod) do

begin

inc(AmountCur);

inc(i);

end;

if AmountCur>AmountBest then

begin

ProdBest:=ProdCur;

AmountBest:=AmountCur;

end;

if AmountCur<AmountWorst then

begin

ProdWorst:=ProdCur;

AmountWorst:=AmountCur;

end;

end;

if AmountBest<>AmountWorst then

MessageBox(Handle, PChar('Наиболее популярный: ' + ProdBest+#13'Количество наименований: '

+IntToStr(AmountBest)+#13+#13+

'Наименее популярный: '+ProdWorst+#13+'Количество наименований: '+IntToStr(AmountWOrst)+#13+#13),

'Наиболее и наименее популярный производитель', MB\_Ok)

else

MessageBox(Handle, PChar('Наиболее популярный: ' + ProdBest +#13+'Количество наименований: '

+IntToStr(AmountBest)+#13+ #13+

'Наименее популярный не определён.'+#13+#13),

'Наиболее и наименее популярный производитель', MB\_Ok);

end;

procedure TForm1.UpdateView(ReportUpdate:TReportPos; ReportUpdate2:TReport);

var Item: TListItem;

i: integer;

begin // обновление отображения

lvReport.Items.Clear;

if ReportUpdate2=nil then

for i:=Low(ReportUpdate) to High(ReportUpdate) do

begin

Item := lvReport.Items.Add;

Item.Caption := IntToStr(i+1);

Item.SubItems.Add(ReportUpdate[i].Code);

Item.SubItems.Add(ReportUpdate[i].Name);

Item.SubItems.Add(ReportUpdate[i].Prod);

Item.SubItems.Add(IntTostr(ReportUpdate[i].amount));

Item.SubItems.Add(FloatToStr(ReportUpdate[i].price));

Item.SubItems.Add(IntTostr(ReportUpdate[i].VAT));

Item.SubItems.Add(FloatToStr(ReportUpdate[i].VatPrice));

Item.SubItems.Add(FloatToStr(ReportUpdate[i].PriceSold));

Item.SubItems.Add(IntTostr(ReportUpdate[i].AmountSold));

Item.SubItems.Add(FloatToStr(ReportUpdate[i].income));

end

else

for i := Low(ReportUpdate2) to High(ReportUpdate2) do

begin

Item := lvReport.Items.Add;

Item.Caption := IntToStr(i+1);

Item.SubItems.Add(DateToStr(ReportUpdate2[i].Date));

Item.SubItems.Add(FloatToStr(ReportUpdate2[i].Income));

Item.SubItems.Add(FloatToStr(ReportUpdate2[i].Expense));

Item.SubItems.Add(FloatTostr(ReportUpdate2[i].Total));

end;

end;

end.

unit AddForm;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Buttons,

Vcl.Samples.Spin, Vcl.ExtCtrls, Vcl.ComCtrls;

type

TForm2 = class(TForm)

EditlbCode: TLabeledEdit;

EditLbName: TLabeledEdit;

EditLbProd: TLabeledEdit;

EditSpinAmount: TSpinEdit;

EditLbPrice: TLabeledEdit;

EditLbVAT: TLabeledEdit;

EditSpinAmountSold: TSpinEdit;

LbAmount: TLabel;

LbAmountSold: TLabel;

BtnBitCancel: TBitBtn;

BtnBitConfirm: TBitBtn;

EditLbPriceSold: TLabeledEdit;

editDateEnd: TDateTimePicker;

lbStartDate: TLabel;

editDateStart: TDateTimePicker;

lbEndDate: TLabel;

CheckBoxDate: TCheckBox;

procedure FormShow(Sender: TObject);

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

procedure EditLbPriceKeyPress(Sender: TObject; var Key: Char);

procedure EditLbPriceSoldKeyPress(Sender: TObject; var Key: Char);

procedure EditLbVATKeyPress(Sender: TObject; var Key: Char);

private

function GetCode: string;

function GetName: string;

function GetProd: string;

function GetAmount: integer;

function GetPrice: real;

function GetVAT: integer;

function GetAmountSold: integer;

function GetPriceSold: real;

function GetStartDate: TDate;

function GetEndDate: TDate;

function GetIncome: real;

function GetExpense: real;

function GetIsMonth: boolean;

{ Private declarations }

public

property Code: string read GetCode;

property Name: string read GetName;

property Prod: string read GetProd;

property Amount: integer read GetAmount;

property price: real read GetPrice;

property VAT: integer read GetVAT;

property PriceSold: real read GetPriceSold;

property AmountSold: integer read GetAmountSold;

property StartDate: TDate read GetStartDate;

property EndDate: TDate read GetEndDate;

property Income: real read GetIncome;

property Expense: real read GetExpense;

property IsMonth: boolean read GetIsMonth;

function AddNewPosition :integer;

function AddNewTotal: integer;

function EditCurTotal (Income, Expense: real): integer;

function EditCurPosition(Index: integer; const Code, Name, Prod: string; Amount: integer;

price: Real; VAT: integer;PriceSold: real;AmountSold:integer):integer;

end;

var

Form2: TForm2;

PosIndex: integer;

implementation

uses MainUnit;

{$R \*.dfm}

function TForm2.AddNewPosition: integer;

begin

caption:='Добавить позицию';

tag:=0;

CheckBoxDate.Visible:=False;

editDateStart.Visible:=False;

editDateEnd.Visible:=False;

lbEndDate.Visible:=False;

lbStartDate.Visible:=False;

LbAmount.Visible:=True;

LbAmountSold.Visible:=True;

EditlbCode.Visible:=True;

EditlbName.Visible:=True;

EditLbProd.Visible:=True;

EditSpinAmount.Visible:=True;

EditlbPrice.Visible:=True;

EditlbVAT.Visible:=True;

EditlbPriceSold.Visible:=True;

EditSpinAmountSold.Visible:=True;

EditLbPrice.EditLabel.Caption:='Закупочная цена:';

EditLbPriceSold.EditLabel.Caption:='Цена продажи:';

editlbPrice.Top:=155;

EditlbPrice.Width:=182+46;

EditlbPriceSold.Width:=182+46;

EditlbPriceSold.top:=155+60;

editLbPrice.Left:=40;

EditLbPriceSold.Left:=40;

CheckBoxDate.Checked:=False;

EditLbCode.text:='';

EditLbName.text:='';

EditLbProd.text:='';

EditSpinAmount.text:='0';

EditlbPrice.text:='0,00';

EditlbVAT.text:='0';

EditlbPriceSold.Text:='0,00';

EditSpinAmountSold.text:='0';

result:=ShowModal;

end;

function TForm2.AddNewTotal: integer;

begin

caption:='Добавить период';

tag:=2;

CheckBoxDate.Visible:=True;

editDateStart.Visible:=True;

editDateEnd.Visible:=True;

lbEndDate.Visible:=True;

lbStartDate.Visible:=True;

LbAmount.Visible:=False;

LbAmountSold.Visible:=False;

EditlbCode.Visible:=False;

EditlbName.Visible:=False;

EditLbProd.Visible:=False;

EditSpinAmount.Visible:=False;

EditlbPrice.Visible:=False;

EditlbVAT.Visible:=False;

EditlbPriceSold.Visible:=False;

EditSpinAmountSold.Visible:=False;

result:=ShowModal;

end;

function TForm2.EditCurPosition(index: integer;const Code, Name, Prod: string; Amount: integer;

price: Real; VAT:integer; PriceSold:real; AmountSold: integer): integer;

begin

Caption:='Изменить позицию';

tag:=1;

EditLbPrice.EditLabel.Caption:='Закупочная цена:';

EditLbPriceSold.EditLabel.Caption:='Цена продажи:';

CheckBoxDate.Visible:=False;

editDateStart.Visible:=False;

editDateEnd.Visible:=False;

lbEndDate.Visible:=False;

lbStartDate.Visible:=False;

LbAmount.Visible:=True;

LbAmountSold.Visible:=True;

EditLbProd.Visible:=True;

EditSpinAmount.Visible:=True;

EditlbPrice.Visible:=True;

EditlbVAT.Visible:=True;

EditlbPriceSold.Visible:=True;

EditSpinAmountSold.Visible:=True;

EditLbCode.text:=code;

EditLbName.text:=name;

EditLbProd.text:=Prod;

EditSpinAmount.text:=IntTOStr(Amount);

EditlbPrice.text:=FloatToStr(Price);

EditlbVAT.text:=IntTOStr(VAT);

EditlbPriceSold.Text:=FloatToStr(PriceSold);

EditSpinAmountSold.text:=IntTOStr(AmountSold);

PosIndex:=Index;

result:=ShowModal;

end;

function TForm2.EditCurTotal(Income, Expense: real): integer;

begin

Caption:='Изменить позицию';

tag:=3;

EditLbPrice.EditLabel.Caption:='Прибыль:';

EditLbPriceSold.EditLabel.Caption:='Траты:';

EditlbPrice.Visible:=True;

EditlbPriceSold.Visible:=True;

EditlbPrice.Width:=300;

EditlbPrice.top:=70;

EditlbPriceSold.Width:=300;

EditlbPriceSold.top:=180;

editLbPrice.Left:=152;

EditLbPriceSold.Left:=152;

EditlbPrice.Text:=FloatToStr(Income);

EditLbPriceSold.Text:=FloatToStr(Expense);

CheckBoxDate.Visible:=False;

EditlbCode.Visible:=False;

EditLbName.Visible:=False;

editDateStart.Visible:=False;

editDateEnd.Visible:=False;

lbEndDate.Visible:=False;

lbStartDate.Visible:=False;

LbAmount.Visible:=False;

LbAmountSold.Visible:=False;

EditLbProd.Visible:=False;

EditSpinAmount.Visible:=False;

EditlbVAT.Visible:=False;

EditSpinAmountSold.Visible:=False;

EditlbPrice.Text:=FloatToStr(Income);

EditLbPriceSold.Text:=FloatToStr(Expense);

result:=ShowModal;

end;

procedure TForm2.EditLbPriceKeyPress(Sender: TObject; var Key: Char);

begin

case Key of

'0'..'9',#8:

;

'.',',':

begin

if key<>',' then

Key := ',';

if Pos(',', EditLbPrice.Text) <> 0

then Key := Chr(0);

end

else

key:=CHR(0);

end;

end;

procedure TForm2.EditLbPriceSoldKeyPress(Sender: TObject; var Key: Char);

begin

case Key of

'0'..'9',#8:

;

'.',',':

begin

if key<>',' then

Key := ','; // заменим разделитель на допустимый

if Pos(',', EditLbPriceSold.Text) <> 0

then Key := Chr(0); // запрет ввода второго разделителя

end

else

key:=CHR(0);

end;

end;

procedure TForm2.EditLbVATKeyPress(Sender: TObject; var Key: Char);

begin

case key of

'0'..'9',#8: ;

else

key:=CHR(0);

end;

end;

procedure TForm2.FormCloseQuery(Sender: TObject; var CanClose: Boolean);

var res: integer;

InputErrors: array[1..8] of string;

i: Integer;

begin

if ModalResult=MrOk then

begin

CanClose:=True;

if (tag=0) or (tag=1) then

begin

i:=Low(Form1.ReportPos);

while CanClose and (i <= High(Form1.ReportPos)) do

begin

if (Form1.ReportPos[i].Code=EditlbCode.Text) and (i<>PosIndex) then

CanClose:=False;

inc(i);

end;

if not CanClose then

MessageBox(handle, PWideChar('Вы уже имеете позицию с данным кодом.'+#13),

'Ошибка', MB\_OK or MB\_ICONWArning)

else

begin

if EditSpinAmount.text='' then

EditSpinAmount.text:='0';

if EditSpinAmountSold.text='' then

EditSpinAmountSold.text:='0';

if EditLbPriceSold.text='' then

EditLbPriceSold.text:='0';

if EditLbPrice.text='' then

EditLbPrice.text:='0';

if EditLbVAT.text='' then

EditLbVAT.text:='0';

if EditlbCode.Text='' then

InputErrors[1]:='-Код не введён'+#13;

if EditlbName.Text='' then

InputErrors[2]:='-Наименование не введено'+#13;

if EditlbProd.Text='' then

InputErrors[3]:='-Производитель не введён'+#13;

if StrToInt(EditSpinAmount.Text)<=0 then

InputErrors[4]:='-Количество отрицательное или нулевое'+#13;

if StrToFloat(EditLbPrice.Text)<=0 then

InputErrors[5]:='-Закупочная цена отрицательная или нулевая'+#13;

if (StrToInt(EditLbVAT.Text)<0) or (StrToInt(EditLbVAT.Text)>30) then

InputErrors[6]:='-НДС не попадает в допустимый диапазон от 0 до 30%'+#13;

if StrToFloat(EditLbPriceSold.Text)<=StrToFloat(EditlbPrice.Text)\*(StrToInt(EditlbVAT.Text)+100)/100 then

InputErrors[7]:='-Цена продажи не превышает расходы на закупку с учётом НДС'+#13;

if StrToInt(EditSpinAmountSold.Text)>StrToInt(EditSpinAmount.Text) then

InputErrors[8]:='-Количество проданых товаров превышает количество имеющихся'+#13;

for i := 1 to 8 do

if InputErrors[i]<>'' then

CanClose:=False;

if not CanClose then

MessageBox(Handle, PWideChar('Выполнение невозможно, допущены следующие ошибки:'+

#13 + InputErrors[1]+InputErrors[2] + InputErrors[3] + InputErrors[4] + InputErrors[5]

+ InputErrors[6] + InputErrors[7] + InputErrors[8]),

'Предупреждение', MB\_Ok or MB\_IconWArning);

end;

end;

if tag=2 then

begin

if editDateStart.Date>editDateEnd.Date then

begin

CanClose:=False;

MessageBox(Handle, PWideChar('Выполнение невозможно: '+#13+'Начальная дата позже конечной.')

,'Предупреждение', Mb\_Ok or MB\_ICONWARNING);

end;

end;

if tag=3 then

begin

if EditLbPriceSold.text='' then

EditLbPriceSold.text:='0';

if EditLbPrice.text='' then

EditLbPrice.text:='0';

if StrToFloat(EditLbPrice.Text)<0 then

InputErrors[1]:='Прибыль отрицательная'+#13;

if StrToFloat(EditLbPriceSold.Text)<0 then

InputErrors[2]:='Траты отрицательные'+#13;

for i := 1 to 2 do

if InputErrors[i]<>'' then

CanClose:=False;

if not CanClose then

MessageBox(Handle, PWideChar('Выполнение невозможно, допущены следующие ошибки:'+

#13 + InputErrors[1]+InputErrors[2]),

'Предупреждение', MB\_Ok or MB\_IconWArning);

end;

end

else

begin

res:=MessageBox(Handle, 'Уверены, что хотите закрыть без добавления?', 'Предупреждение',

Mb\_YESNO or MB\_ICONWARNING);

CanClose:=(Res=IDYES);

end;

end;

procedure TForm2.FormShow(Sender: TObject);

begin

if form1.lvReport.Tag=0 then

EditLbCode.SetFocus

else

if tag=3 then

EditLbPrice.SetFocus;

end;

function TForm2.GetAmount: integer;

begin

result:=StrToInt(EditSpinAmount.text);

end;

function TForm2.GetAmountSold: integer;

begin

result:=StrToInt(EditSpinAmountSold.text);

end;

function TForm2.GetCode: string;

begin

result:=EditlbCode.text;

end;

function TForm2.GetEndDate: TDate;

begin

result:=editDateEnd.Date;

end;

function TForm2.GetExpense: real;

begin

result:=StrTOFloat(EditlbPriceSold.text);

end;

function TForm2.GetIncome: real;

begin

result:=StrToFloat(EditlbPrice.text);

end;

function TForm2.GetIsMonth: boolean;

begin

result:=CheckBoxDate.Checked;

end;

function TForm2.GetName: string;

begin

result:=EditLbName.text;

end;

function TForm2.GetPrice: real;

begin

result:=StrToFLoat(EditLbPrice.text);

end;

function TForm2.GetProd: string;

begin

result:=EditLbProd.text;

end;

function TForm2.GetStartDate: TDate;

begin

result:=editDateStart.Date;

end;

function TForm2.GetVAT: integer;

begin

result:=StrToInt(EditLbVat.text);

end;

function TForm2.GetPriceSold: real;

begin

result:=StrToFloat(EditLbPriceSold.text);

end;

end.

unit Graphic;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ComCtrls, Vcl.ExtCtrls, Vcl.StdCtrls,

Vcl.ActnMan, Vcl.ActnColorMaps;

type

TDots = array of real;

TForm3 = class(TForm)

Panel1: TPanel;

Panel2: TPanel;

TrackBarDraw: TTrackBar;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

btnDraw: TButton;

ComBoxDraw: TComboBox;

Label5: TLabel;

Label6: TLabel;

Label7: TLabel;

Label8: TLabel;

ComBoxMethod: TComboBox;

Label9: TLabel;

Image1: TImage;

Panel3: TPanel;

PaintBox1: TPaintBox;

PaintBox2: TPaintBox;

lbFirstGr: TLabel;

lbSecondGr: TLabel;

btnSaveGraphic: TButton;

DlgSaveGraphic: TSaveDialog;

lbCurGraphic: TLabel;

Label10: TLabel;

Label11: TLabel;

Label12: TLabel;

procedure btnDrawClick(Sender: TObject);

procedure ComBoxMethodSelect(Sender: TObject);

procedure FormCreate(Sender: TObject);

private

DotsArray: TDots;

procedure FindMaxMin(var HighLim, LowLim:real);

procedure FindDotsExponential;

procedure FindDotsAverage;

procedure FindDotsLowSqr;

procedure BuildGraphic(HighLim, LowLim:real);

{ Private declarations }

public

{ Public declarations }

end;

var

Form3: TForm3;

Method, Column: byte;

implementation

uses MainUnit;

{$R \*.dfm}

{ TForm3 }

procedure TForm3.btnDrawClick(Sender: TObject);

var HighLim, LowLim: real;

begin

DotsArray:=nil;

method:=ComBoxMethod.ItemIndex;

Column:=ComBoxDraw.ItemIndex;

if method=0 then

begin

FindDotsAverage;

case column of

0: lbCurGraphic.Caption:='Текущий график: График по таблице с прогнозом по скользящим средним по прибыли';

1: lbCurGraphic.Caption:='Текущий график: График по таблице с прогнозом по скользящим средним по тратам';

2: lbCurGraphic.Caption:='Текущий график: График по таблице с прогнозом по скользящим средним по итогам';

end;

end

else

if method=1 then

begin

FindDotsExponential;

case column of

0: lbCurGraphic.Caption:='Текущий график: График по таблице с графиком экспоненциального сглаживания и прогнозом по прибыли';

1: lbCurGraphic.Caption:='Текущий график: График по таблице с графиком экспоненциального сглаживания и прогнозом по тратам';

2: lbCurGraphic.Caption:='Текущий график: График по таблице с графиком экспоненциального сглаживания и прогнозом по итогам';

end;

end

else

begin

FindDotsLowSqr;

case column of

0: lbCurGraphic.Caption:='Текущий график: График по таблице с прогнозом по методу наименьших квадратов по прибыли';

1: lbCurGraphic.Caption:='Текущий график: График по таблице с прогнозом по методу наименьших квадратов по тратам';

2: lbCurGraphic.Caption:='Текущий график: График по таблице с прогнозом по методу наименьших квадратов по итогам';

end;

end;

FindMaxMin(HighLim, LowLim);

BuildGraphic(HighLim, LowLim);

Form1.actSaveGraphic.Enabled:=True;

end;

procedure TForm3.BuildGraphic(HighLim, LowLim:real);

var x, i, y, step: integer;

m, m2: real;

begin

x:=80;

with image1 do

begin

Picture:=nil;

PaintBox2.Refresh;

step:=Round(width/length(DotsArray)/1.2);

if (HighLim>=0) and (LowLim>=0) and (HighLim+LowLim<>0) then

m:=height/ (HighLim+LowLim)/1.2

else

if (HighLim<0) and (LowLim<0) then

m:=-height/ (HighLim+LowLim)/1.2

else

if (HighLim>=0) and (LowLim<0) then

m:=height/(HighLim-LowLim)/1.2

else

m:=0;

with canvas do

begin

Pen.Color:=clBlack;

Brush.Color:=clBlack;

moveto(x,Height);

lineto(x,0);

Polygon([Point(x-5,10),Point(x+5,10),Point(x,0)]);

Brush.Color:=clWhite;

Font.Height:=16;

case column of

0: TextOut(10, 0, 'Прибыль');

1: TextOut(12, 0, 'Траты');

2: TextOut(14, 0, 'Итоги');

end;

Pen.Width:=1;

if method=1 then

begin

if column=2 then

begin

TextOut(7, Round((Height-Round(LowLim\*m))/2)-16, FloatToStr(Round(100\*LowLim)/100));

TextOut(7, Round(Height/2)-16, '0');

TextOut(7, Round((Height-Round(HighLim\*m))/2)-16, FloatToStr(Round(100\*HighLim)/100));

end

else

begin

TextOut(7, Height-Round(LowLim\*m)-16, FloatToStr(Round(100\*LowLim)/100));

TextOut(7, Height-Round(HighLim\*m)-16, FloatToStr(Round(100\*HighLim)/100));

end;

paintbox1.Canvas.Pen.Color:=clGreen;

paintbox1.Canvas.Brush.Color:=clGreen;

paintBox1.Canvas.Rectangle(0,0,paintBox1.Height,paintBox2.Width);

lbFirstGr.Caption:='График по таблице итогов';

case column of

0: y:= Height-Round(Form1.ReportTotal[0].Income\*m)-1;

1: y:= Height-Round(Form1.ReportTotal[0].Expense\*m)-1;

2: y:= Round((Height-Round(Form1.ReportTotal[0].Total\*m))/2)-1;

end;

Pen.Color:=clBlack;

Ellipse(x-1,y-2,x+1,y);

MoveTo(x, y);

for i := Low(Form1.ReportTotal) to High(Form1.ReportTotal)-1 do

begin

case column of

0: y:=Height-Round(Form1.ReportTotal[i+1].Income\*m)-1;

1: y:=Height-Round(Form1.ReportTotal[i+1].Expense\*m)-1;

2: y:=Round((Height-Round(Form1.ReportTotal[i+1].Total\*m))/2)-1

end;

Pen.Color:=clBlack;

Ellipse(x+step,y-2,x+Step+1,y);

Pen.Color:=clGreen;

lineto(x+step, y);

inc(x, step);

end;

Pen.Style:=psDot;

if column=2 then

begin

y:=Round((Height-Round(DotsArray[High(Form1.ReportTotal)+1]\*m))/2)-1;

lineto(x+step, y);

Pen.Color:=clBlack;

Ellipse(x+step,y-2,x+Step+1,y);

end

else

begin

y:=Height-Round(DotsArray[High(Form1.ReportTotal)+1]\*m)-1;

lineto(x+step, y);

Pen.Color:=clBlack;

Ellipse(x+step,y-2,x+Step+1,y);

end;

x:=80;

paintbox2.Canvas.Pen.Color:=clRed;

paintbox2.Canvas.Brush.Color:=clRed;

paintBox2.Canvas.Rectangle(0,0,paintBox1.Height,paintBox1.Width);

lbSecondGr.Caption:='График экспоненциального сглаживания';

Pen.Style:=psSolid;

if column=2 then

begin

y:= Round((Height-Round(DotsArray[0]\*m))/2)-1;

Pen.Color:=clBlack;

Ellipse(x,y-2,x+1,y);

moveto(x, y);

for i := Low(DotsArray) to High(DotsArray)-2 do

begin

y:=Round((Height-Round(DotsArray[i+1]\*m))/2)-1;

Pen.Color:=clBlack;

Ellipse(x+step,y-2,x+Step+1,y);

Pen.Color:=clRed;

lineto(x+step, y);

inc(x, step);

end;

Pen.Style:=psDot;

y:=Round((Height-Round(DotsArray[High(DotsArray)]\*m))/2)-1;

lineto(x+step,y);

Pen.Color:=clBlack;

Ellipse(x+step,y-2,x+Step+1,y);

end

else

begin

y:= Height-Round(DotsArray[0]\*m)-1;

Pen.Color:=clBlack;

Ellipse(x,y-2,x+1,y);

MoveTo(x, y);

for i := Low(DotsArray) to High(DotsArray)-2 do

begin

y:=Height-Round(DotsArray[i+1]\*m)-1;

Pen.Color:=clBlack;

Ellipse(x+step,y-2,x+Step+1,y);

Pen.Color:=clRed;

lineto(x+step, y);

inc(x, step);

end;

Pen.Style:=psDot;

y:=Height-Round(DotsArray[High(DotsArray)]\*m)-1;

Pen.Color:=clBlack;

Ellipse(x+step,y-2,x+Step+1,y);

Pen.Color:=clRed;

lineto(x+step, y);

end;

end

else

begin

paintbox1.Canvas.Pen.Color:=clGreen;

paintbox1.Canvas.Brush.Color:=clGreen;

paintBox1.Canvas.Rectangle(0,0,paintBox1.Height,paintBox1.Width);

lbFirstGr.Caption:='График по таблице итогов';

lbSecondGr.Caption:='';

Pen.Style:=psSolid;

if (column=2) or (method=2) then

begin

Pen.Color:=clBlack;

TextOut(7, Round((Height-Round(LowLim\*m))/2)-16, FloatToStr(Round(100\*LowLim)/100));

TextOut(7, Round(Height/2)-16, '0');

TextOut(7, Round((Height-Round(HighLim\*m))/2)-16, FloatToStr(Round(100\*HighLim)/100));

y:= Round((Height-Round(DotsArray[0]\*m))/2)-1;

Ellipse(x,y-2,x+1,y);

moveto(x, y);

for i := Low(DotsArray) to High(DotsArray)-7 do

begin

y:=Round((Height-Round(DotsArray[i+1]\*m))/2)-1;

Pen.Color:=clBlack;

Ellipse(x+step,y-2,x+Step+1,y);

Pen.Color:=clGreen;

lineto(x+step, y);

inc(x, step);

end;

Pen.Style:=psDot;

for i := High(DotsArray)-6 to High(DotsArray)-1 do

begin

y:=Round((Height-Round(DotsArray[i+1]\*m))/2)-1;

Pen.Color:=clBlack;

Ellipse(x+step,y-2,x+Step+1,y);

Pen.Color:=clGreen;

lineto(x+step, Round((Height-Round(DotsArray[i+1]\*m))/2)-1);

inc(x, step);

end;

end

else

begin

Pen.Color:=clBlack;

TextOut(7, Height-Round(LowLim\*m)-16, FloatToStr(Round(100\*LowLim)/100));

TextOut(7, Height-Round(HighLim\*m)-16, FloatToStr(Round(100\*HighLim)/100));

y:=Height-Round(DotsArray[0]\*m)-1;

Pen.Color:=clBlack;

Ellipse(x,y-2,x+1,y);

MoveTo(x, y);

for i := Low(DotsArray) to High(DotsArray)-7 do

begin

y:=Height-Round(DotsArray[i+1]\*m)-1;

Pen.Color:=clBlack;

Ellipse(x+step,y-2,x+step+1,y);

Pen.Color:=clGreen;

lineto(x+step, Height-Round(DotsArray[i+1]\*m)-1);

inc(x, step);

end;

Pen.Style:=psDot;

for i := High(DotsArray)-6 to High(DotsArray)-1 do

begin

y:=Height-Round(DotsArray[i+1]\*m)-1;

Pen.Color:=clBlack;

Ellipse(x+step,y-2,x+step+1,y);

Pen.Color:=clGreen;

lineTo(x+step, Height-Round(DotsArray[i+1]\*m)-1);

inc(x, step);

end;

end;

end;

end;

end;

end;

procedure TForm3.ComBoxMethodSelect(Sender: TObject);

begin

if ComBoxMethod.ItemIndex=1 then

TrackBarDraw.Enabled:=True

else

TrackBarDraw.Enabled:=False;

end;

procedure TForm3.FindDotsLowSqr;

var Total, PerTotal, Period, PeriodSqr, aCof, bCof: real;

i, N: integer;

begin

Total:=0;

PerTotal:=0;

PeriodSqr:=0;

setlength(DotsArray, length(Form1.ReportTotal)+6);

with Form1 do

begin

N:=length(ReportTotal);

case column of

0:

for i := Low(ReportTotal) to N-1 do

begin

DotsArray[i]:=ReportTotal[i].Income;

Total:=Total+ReportTotal[i].Income;

PerTotal:=PerTotal+ReportTotal[i].Income\*(i+1);

PeriodSqr:=sqr(i+1)+PeriodSqr;

end;

1:

for i := Low(ReportTotal) to N-1 do

begin

DotsArray[i]:=ReportTotal[i].Expense;

Total:=Total+ReportTotal[i].Expense;

PerTotal:=PerTotal+ReportTotal[i].Expense\*(i+1);

PeriodSqr:=sqr(i+1)+PeriodSqr;

end;

2:

for i := Low(ReportTotal) to N-1 do

begin

DotsArray[i]:=ReportTotal[i].Total;

Total:=Total+ReportTotal[i].Total;

PerTotal:=PerTotal+ReportTotal[i].Total\*(i+1);

PeriodSqr:=sqr(i+1)+PeriodSqr;

end;

end;

Period:=(N+1)\*N/2;

aCof:=(PerTotal-(Total\*Period)/N)/(PeriodSqr-sqr(Period)/N);

bCof:=(Total-aCof\*Period)/N;

for i := N to High(DotsArray) do

DotsArray[i]:=aCof\*i+bCof;

end;

end;

procedure TForm3.FindDotsAverage;

const N=6;

var AvgArray: array[0..N] of real;

i, index: Integer;

begin

setlength(DotsArray, length(Form1.ReportTotal)+6);

case column of

0:for i := low(Form1.ReportTotal) to High(Form1.ReportTotal) do

DotsArray[i]:=Form1.ReportTotal[i].Income;

1:for i := low(Form1.ReportTotal) to High(Form1.ReportTotal) do

DotsArray[i]:=Form1.ReportTotal[i].Expense;

2: for i := low(Form1.ReportTotal) to High(Form1.ReportTotal) do

DotsArray[i]:=Form1.ReportTotal[i].Total;

end;

index:= High(Form1.ReportTotal)-1;

AvgArray[0]:=(DotsArray[index-1]+DotsArray[index]+DotsArray[index+1])/3;

for i:=1 to N+1 do

begin

DotsArray[index+1]:=AvgArray[i-1]+1/3\*(DotsArray[index]-DotsArray[index-1]);

AvgArray[i]:=(DotsArray[index-1]+DotsArray[index]+DotsArray[index+1])/3;

inc(index);

end;

end;

procedure TForm3.FindDotsExponential;

var cof, average: real;

i: integer;

begin

SetLength(DotsArray, length(Form1.ReportTotal)+1);

cof:=TrackbarDraw.Position/10;

average:=0;

for i := low(Form1.ReportTotal) to High(Form1.ReportTotal) do

begin

case column of

0: average:=Form1.ReportTotal[i].Income+average;

1: average:=Form1.ReportTotal[i].Expense+average;

2: average:=Form1.ReportTotal[i].Total+average;

end;

end;

DotsArray[0]:=average/length(Form1.ReportTotal);

for i := low(Form1.ReportTotal) to High(Form1.ReportTotal) do

begin

case column of

0: DotsArray[i+1]:=cof\*Form1.ReportTotal[i].Income+(1-cof)\*DotsArray[i];

1: DotsArray[i+1]:=cof\*Form1.ReportTotal[i].Expense+(1-cof)\*DotsArray[i];

2: DotsArray[i+1]:=cof\*Form1.ReportTotal[i].Total+(1-cof)\*DotsArray[i];

end;

end;

end;

procedure TForm3.FindMaxMin(var HighLim, LowLim:real);

var i: Integer;

begin

with Form1 do

begin

if method<>2 then

case column of

0:

begin

HighLim:=ReportTotal[0].Income;

LowLim:=ReportTotal[0].Income;

for i := Low(ReportTotal)+1 to High(ReportTotal) do

begin

if ReportTotal[i].Income>HighLim then

HighLim:= ReportTotal[i].Income;

if ReportTotal[i].Income<LowLim then

LowLim:= ReportTotal[i].Income;

end;

end;

1:

begin

HighLim:=ReportTotal[0].Expense;

LowLim:=ReportTotal[0].Expense;

for i := Low(ReportTotal)+1 to High(ReportTotal) do

begin

if ReportTotal[i].Expense>HighLim then

HighLim:= ReportTotal[i].Expense;

if ReportTotal[i].Expense<LowLim then

LowLim:= ReportTotal[i].Expense;

end;

end;

2:

begin

HighLim:=ReportTotal[0].total;

LowLim:=ReportTotal[0].total;

for i := Low(ReportTotal)+1 to High(ReportTotal) do

begin

if ReportTotal[i].total>HighLim then

HighLim:= ReportTotal[i].total;

if ReportTotal[i].total<LowLim then

LowLim:= ReportTotal[i].total;

end;

end;

end

else

begin

HighLim:=DotsArray[0];

LowLim:=DotsArray[0];

for i := Low(DotsArray)+1 to High(DotsArray) do

begin

if DotsArray[i]>HighLim then

HighLim:= DotsArray[i];

if DotsArray[i]<LowLim then

LowLim:= DotsArray[i];

end;

end;

end;

end;

procedure TForm3.FormCreate(Sender: TObject);

begin

DlgSaveGraphic.InitialDir := ExtractFilePath(Application.ExeName);

end;

end.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Обозначение | | | | Наименование | | | | Дополнительные сведения | | | |
|  | | | | Текстовые документы | | | |  | | | |
|  | | | |  | | | |  | | | |
| БГУИР КР 1–40 01 01 951002 ПЗ | | | | Пояснительная записка | | | | 97 с. | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | | Графические документы | | | |  | | | |
|  | | | |  | | | |  | | | |
| ГУИР 951002 218 СП | | | | "Программное средство анализа продаж товаров и построения прогнозов с выводом графиков", А1, схема программы, чертеж | | | | Формат А1 | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  |  |  |  |  | БГУИР КР 1-40 01 01 218 Д1 | | | | | | |
|  |  |  |  |  |
|  |  |  |  |  |
| Изм. | Л. | № докум. | Подп. | Дата | “Программное средство анализа продаж товаров и построения прогнозов с выводом графиков”  Ведомость курсовой  работы |  | | | | Лист | Листов |
| Разраб. | | Махнач А. О. |  |  | Т |  | |  | 97 | 97 |
| Пров. | | Фадеева Е. Е. |  |  | Кафедра ПОИТ  гр. 951002 | | | | | |
|  | |  |  |  |
|  | |  |  |  |
|  | |  |  |  |