МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ



Кафедра «СМАРТ технологии»

Лабораторная работа № 2:

«Использование циклических алгоритмов для обработки данных и для

формирования консольных интерфейсов пользователя»

По дисциплине: «Программирование и алгоритмизация на языках

высокого уровня»

Группа 211-328

Студент Пан Филипп Юльевич

Дата \_\_\_\_\_\_\_\_\_\_\_

Преподаватель Логунова Елена Александровна

2022

Задание

L04\_13. Создайте базу данных и приложение C# для работы с ней

1. Наименование полей первой таблицы: код поставщика, название, телефон.

2. Наименование полей второй таблицы: код поставщика, шифр изделия, название изделия, дата поставки, размер поставки, стоимость поставки, номер склада.

3. Количество записей первой таблицы - 5, количество записей второй таблицы - 20. (Количество записей в таблицах студент может увеличить).

4. Содержание создаваемых запросов:

• данные о поставках, полученных в определенном диапазоне дат, для произвольно заданного склада;

• сведения о заданном изделии с расчетом дополнительного поля - цена изделия

• результаты определения средней и максимальной цены каждого изделия;

• замена шифра одного из изделий.

Ход работы

На Рисунке 1 представлена блок-схема, разработанная на основе контрольных вопросов. В соответствии с ней была написана программа (см. Приложение А). На Рисунке Б.2 представлен результат выполнения работы алгоритма отбора и подсчета элементов.

**Описание алгоритма работы программы**

При запуске программы открывается главное окно (Рисунок 1)

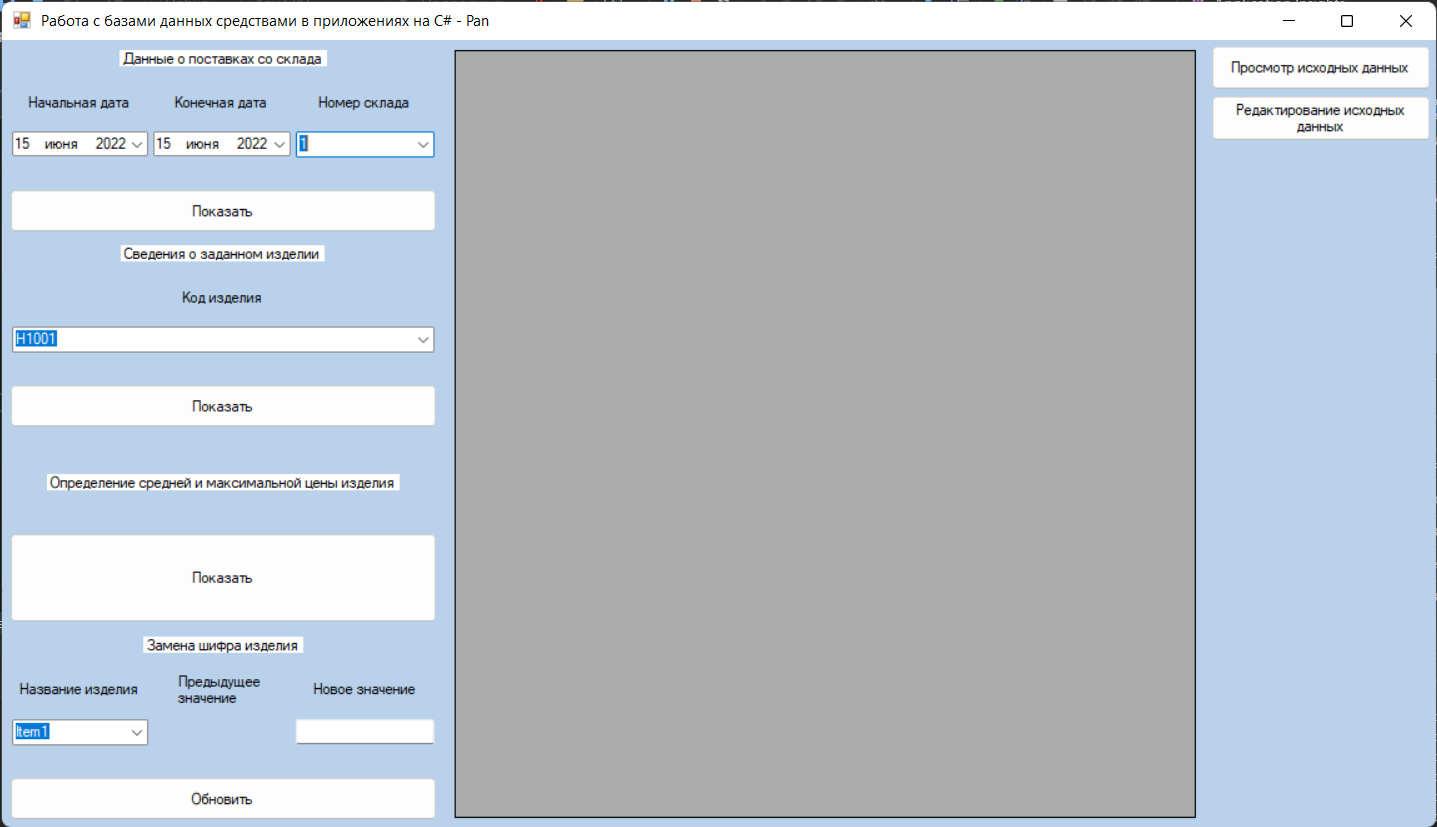


Рисунок - Главное окно

В правом верхнем углу располагаются кнопки для перехода к окнам просмотра и редактирования данных (Рисунок 2).

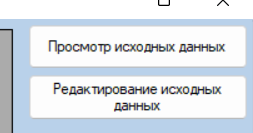


Рисунок - Кнопки на главном окне

При нажатии на кнопку «Просмотр исходных данных» откроется окно с таблицами (Рисунок 3)

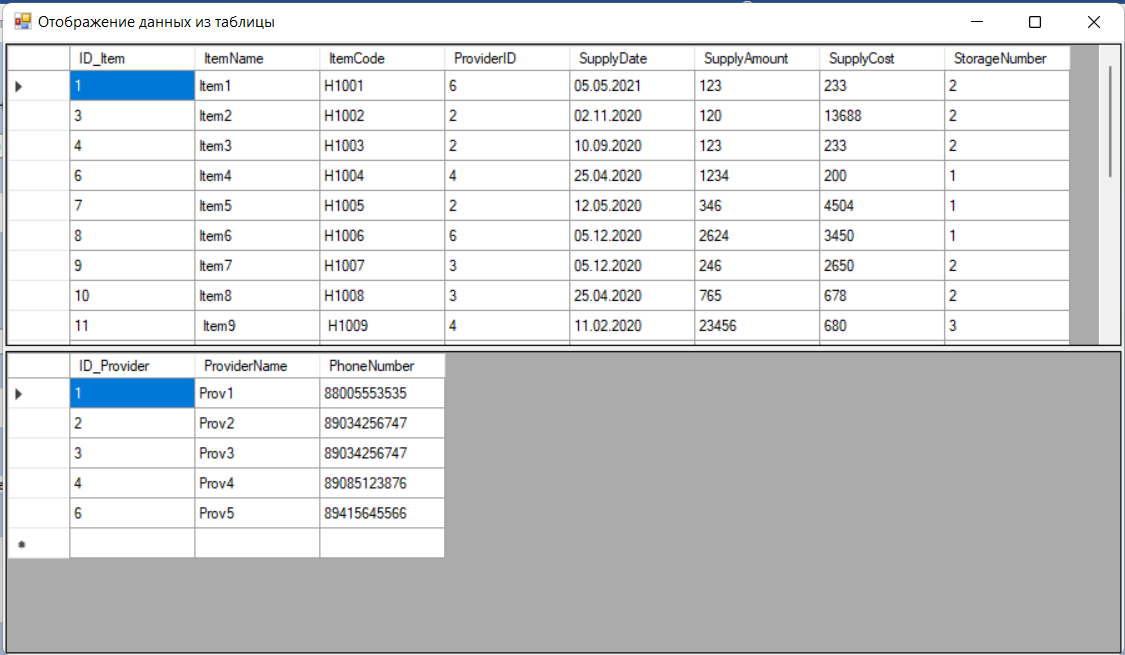


Рисунок - Таблица значений

При нажатии на кнопку «Редактирование исходных данных» откроется окно с возможностью изменить значения (Рисунок 4). Для этого необходимо нажать на строку в таблице, которую нужно изменить, и ввести в поля справа новые значения, а после нажать кнопку «Обновить».

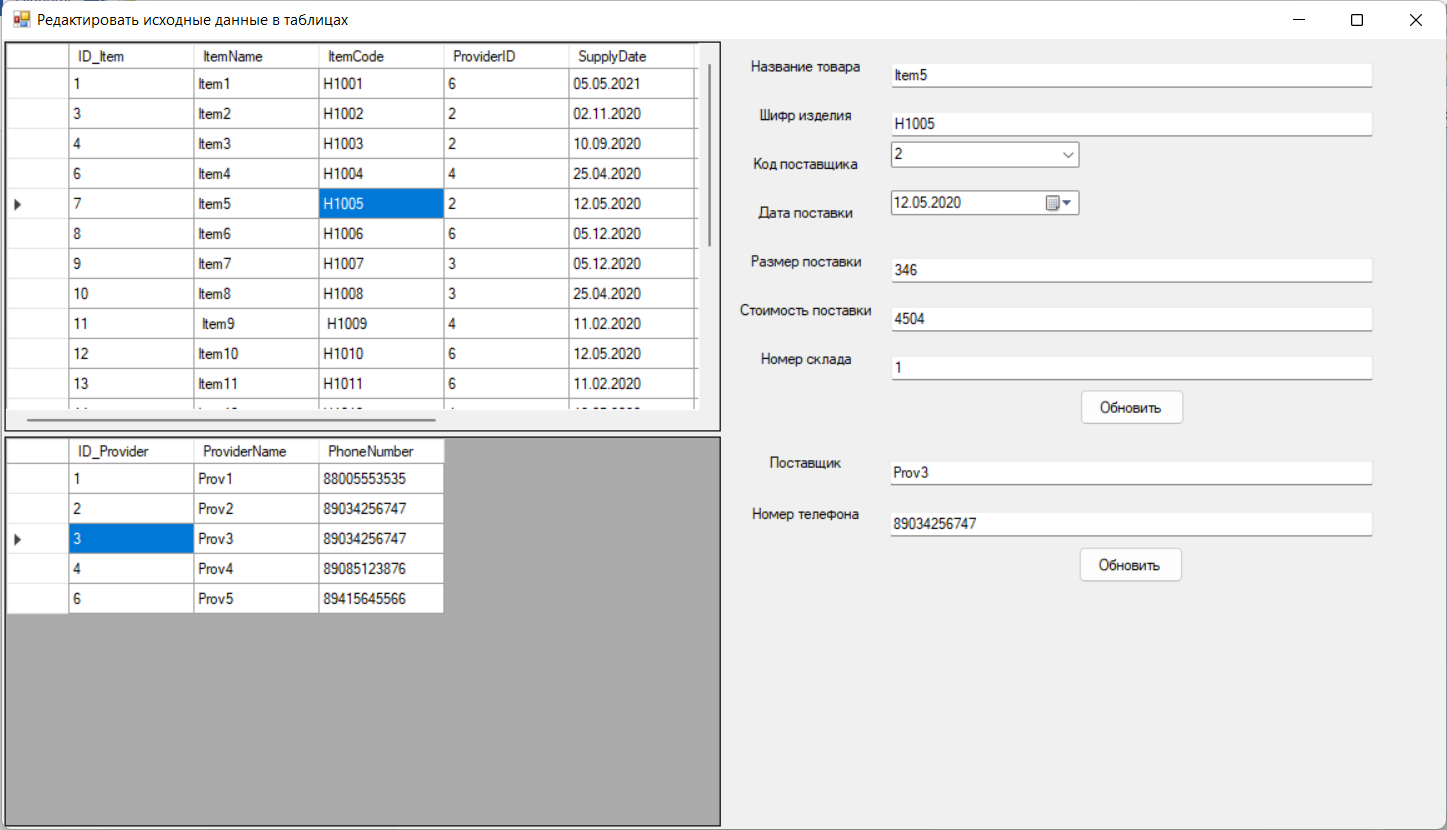


Рисунок - Окно редактирования значений

На главном окне также есть возможность отредактировать данные. Так, на рисунке 5 мы видим окно замены шифра изделия. Необходимо выбрать изделие по его названию (Рисунок 6), после чего станет видно текущий шифр изделия (Рисунок 7), затем можно ввести новое значение и увидеть, что данные обновились (Рисунок 8, 9).

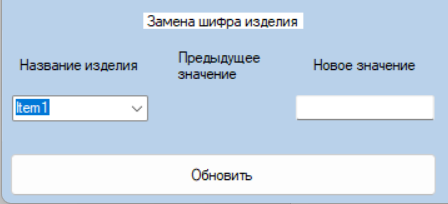


Рисунок - Замена шифра изделия

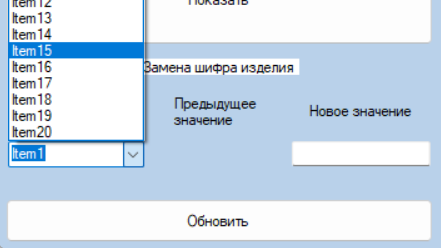


Рисунок - Замена шифра изделия

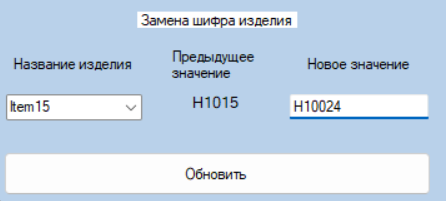


Рисунок - Замена шифра изделия

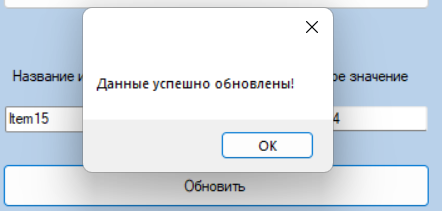


Рисунок - Замена шифра изделия

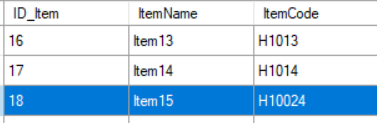


Рисунок - Проверка замены шифра изделия

Также на главном экране доступна функция расчета и просмотра средней и максимальной цены изделия (Рисунок 10, 11)

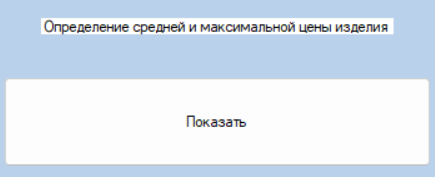


Рисунок - Определение средней и максимальной цены

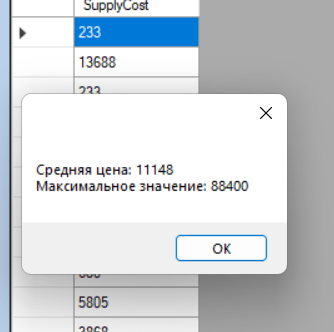


Рисунок - Определение средней и максимальной цены

Следующая функция – вывод данных об изделии по его коду (Рисунок 12).

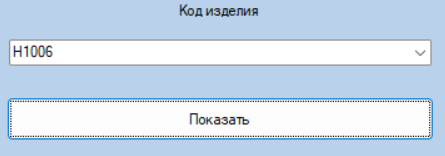


Рисунок - Вывод данных по коду изделия

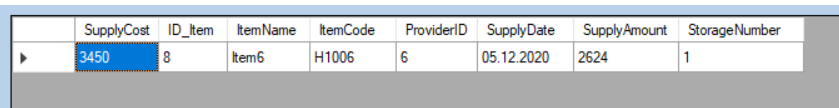


Рисунок - Вывод данных по коду изделия

А в левом верхнем углу главной формы реализован функционал отображения данных о поставках за определенный промежуток времени по номеру склада (Рисунок 14).

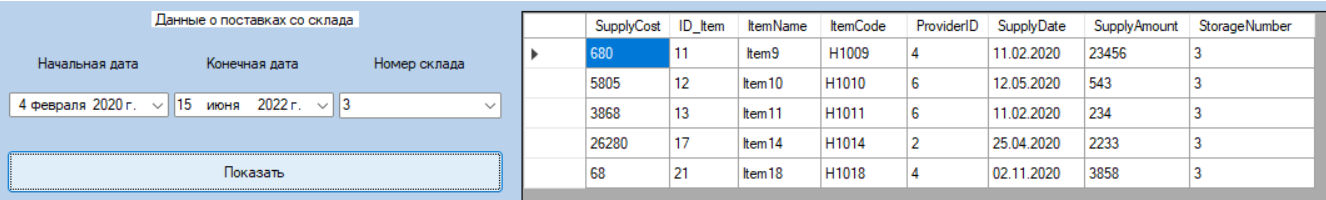


Рисунок - Показ данных о поставках со склада

Проверка работы системы

Проверка правильной работы разных функций в программе проводилась по следующему алгоритму:

1. Дополнить код проверяемой функции для учета исключительных ситуаций и вывода сообщений об ошибке. Добавить проверки на вводимые данные и выполнение запросов.
2. Запустить программу и открыть нужное окно. Ввести корректные данные и выполнить. Сделать все «как надо».
3. Оставить все поля для ввода пустыми и выполнить функцию.
4. Ввести в поля для ввода числовых значений символы и выполнить.
5. Ввести в поля для ввода строковых значений цифры и символы (кириллицу) и выполнить.
6. Ввести значение в выпадающий список вручную, а не выбирать из списка и выполнить.
7. Провести все возможные действия с таблицами данных: ввод значений вручную, реорганизация столбцов и ячеек, удаление строк, выбор пустой строки для выполнения запроса и т.д.

При возникновении ошибки на шагах 2-7, вернуться к коду программы и исправить проблему.

Приложение А. Листинг кода

class DBActions

{

SqlConnection con = new SqlConnection(@"Data Source=(localdb)\ph\_study; Initial Catalog=ProgLabs; Integrated Security=true;");

Encoding utf8 = Encoding.UTF8;

public void openConnection()

{

if (con.State == System.Data.ConnectionState.Closed) con.Open();

}

public void closeConnection()

{

if (con.State == System.Data.ConnectionState.Open) con.Close();

}

public DataSet getAllData()

{

openConnection();

SqlCommand command = new SqlCommand();

command.CommandText = "SELECT \* FROM lab4\_ItemsTable";

command.Connection = con;

SqlDataAdapter da = new SqlDataAdapter(command);

DataSet data = new DataSet();

DataTable dt1 = data.Tables.Add();

DataTable dt2 = data.Tables.Add();

da.Fill(data.Tables[0]);

command.CommandText = "SELECT \* FROM lab4\_ProvidersTable";

da = new SqlDataAdapter(command);

da.Fill(data.Tables[1]);

closeConnection();

return data;

}

public DataTable getPoviderID(string ColName, string TableName)

{

openConnection();

SqlCommand command = new SqlCommand();

command.CommandText = $"SELECT {ColName} FROM {TableName}";

command.Connection = con;

SqlDataAdapter da = new SqlDataAdapter(command);

DataTable data = new DataTable();

closeConnection();

da.Fill(data);

return data;

}

public string getDataByID(string ID, string TableName, string ColName, string IDColName)

{

try

{

openConnection();

SqlCommand command = new SqlCommand();

command.CommandText = $"SELECT {ColName} FROM {TableName} WHERE {IDColName} = '{ID}'";

command.Connection = con;

SqlDataAdapter da = new SqlDataAdapter(command);

SqlDataReader reader = command.ExecuteReader();

reader.Read();

string res = reader.GetValue(0).ToString();

reader.Close();

closeConnection();

return res;

}

catch (Exception e) { return $"error: {e}"; }

}

public DateTime getDatetimeByID(int ID, string TableName, string ColName)

{

try

{

openConnection();

SqlCommand command = new SqlCommand();

command.CommandText = $"SELECT {ColName} FROM {TableName} WHERE ID\_Item = '{ID}'";

command.Connection = con;

SqlDataAdapter da = new SqlDataAdapter(command);

SqlDataReader reader = command.ExecuteReader();

reader.Read();

DateTime res = (DateTime)reader.GetValue(0);

reader.Close();

closeConnection();

return res;

}

catch { return DateTime.Now; }

}

public string editDataInTable1(int ID, string TableName, string IN, string IC, int PI, DateTime SD, int SA, int SC, int SN)

{

try

{

openConnection();

SqlCommand command = new SqlCommand();

command.CommandText = $"UPDATE {TableName} SET " +

$"ItemName = '{IN}', " +

$"ItemCode = '{IC}', " +

$"ProviderID = {PI}, " +

$"SupplyDate = '{SD}', " +

$"SupplyAmount = {SA}, " +

$"SupplyCost = {SC}, " +

$"StorageNumber = {SN} " +

$"WHERE ID\_Item = '{ID}'";

command.Connection = con;

command.ExecuteNonQuery();

closeConnection();

return "Данные успешно обновлены!";

}

catch { return "Произошла ошибка при обновлении данных"; }

}

public string editDataInTable2(int ID, string TableName, string PN, long PH)

{

try

{

openConnection();

SqlCommand command = new SqlCommand();

command.CommandText = $"UPDATE {TableName} SET " +

$"ProviderName = '{PN}', " +

$"PhoneNumber = {PH} " +

$"WHERE ID\_Provider = '{ID}'";

command.Connection = con;

command.ExecuteNonQuery();

closeConnection();

return "Данные успешно обновлены!";

}

catch { return "Произошла ошибка при обновлении данных"; }

}

// Q2

public DataTable getDataSetByID(string What, string TableName, string IDCol, string ID)

{

openConnection();

SqlCommand command = new SqlCommand();

command.CommandText = $"SELECT {What} FROM {TableName} WHERE {IDCol} = '{ID}'";

command.Connection = con;

SqlDataAdapter da = new SqlDataAdapter(command);

DataTable data = new DataTable();

da.Fill(data);

closeConnection();

return data;

}

// Q4

public string editItemCode(string TableName, string ID, string IC)

{

try

{

openConnection();

SqlCommand command = new SqlCommand();

command.CommandText = $"UPDATE {TableName} SET ItemCode = '{IC}' WHERE ItemName = '{ID}'";

command.Connection = con;

command.ExecuteNonQuery();

closeConnection();

return "Данные успешно обновлены!";

}

catch { return "Произошла ошибка при обновлении данных"; }

}

// Q1

public DataTable getDataSetFromDateTime(DateTime Sd, DateTime Ed, string SN)

{

openConnection();

SqlCommand command = new SqlCommand();

command.CommandText = $"SELECT \* FROM lab4\_ItemsTable WHERE (SupplyDate BETWEEN @Sd AND @Ed) AND (StorageNumber = '{SN}')";

command.Connection = con;

SqlParameter Sparameter = new SqlParameter("@Sd", Sd);

command.Parameters.Add(Sparameter);

SqlParameter Eparameter = new SqlParameter("@Ed", Ed);

command.Parameters.Add(Eparameter);

SqlDataAdapter da = new SqlDataAdapter(command);

DataTable data = new DataTable();

da.Fill(data);

closeConnection();

return data;

}