СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Цель: составить приложение по выданному заданию.

Задание: поиск кратчайшего пути по технологической карте изделия. У изделия есть список операций (описанной в виде файла), которые должны быть выполнены определенными станками. У каждого завода есть несколько видов станков. Заводов всего 4 штуки. По составленной технологической карте списка операций вывести последовательность посещения изделием заводов и вывести эту последовательность в текстовый файл.

Задачи:

1) Составить алгоритм решения задачи;
2) Выбрать сущности, их методы и атрибуты, необходимые для решения задачи;
3) Составить диаграмму классов на основе выбранных сущностей;
4) Построить UML-диаграмму вариантов использования программы с инструкцией по их реализации пользователем;
5) Составить блок-схемы наиболее значимых процедур и методов;
6) По составленным блок-схемам написать код программы;
7) Сравнить результат, полученный с помощью программы, с ожидаемым результатом;
8) Написать заключение по лабораторной работе.

ОСНОВНАЯ ЧАСТЬ

1 Аналитическая часть

1.1 Алгоритм решения задачи

В приложении была создана форма с тремя кнопками, уточняющими надписями и полем для вывода ListBox1.

Задача была решена по следующему алгоритму:

1) Были созданы массивы станков и операций с сгенерированными данными. Дынные были сгенерированы таким образом, что на каждом станке можно сделать одну операцию из заданного файла и каждую операцию можно сделать хотя бы на одном станке. Каждый станок ссылается на один из четырёх заводов. Операции в заданном файле не повторяются.

2) По сгенерированным данным была определена кратчайшая последовательность посещения изделием заводов и выведена в текстовый фал.

3) В каждую строку поля вывода ListBox1 были выведены номера заводов, на которых можно выполнить соответственную операцию изделия из файла. Номер строки ListBox1 соответствует номеру операции в файле, начиная с нуля.

Третий пункт алгоритма необходим для ручной проверки пункта 2.

Каждому пункту составленного алгоритма была поставлена в соответствие кнопка на форме.

1.2 Диаграмма классов

В соответствии с составленным алгоритмом была сделана диаграмма классов (рисунок 1).

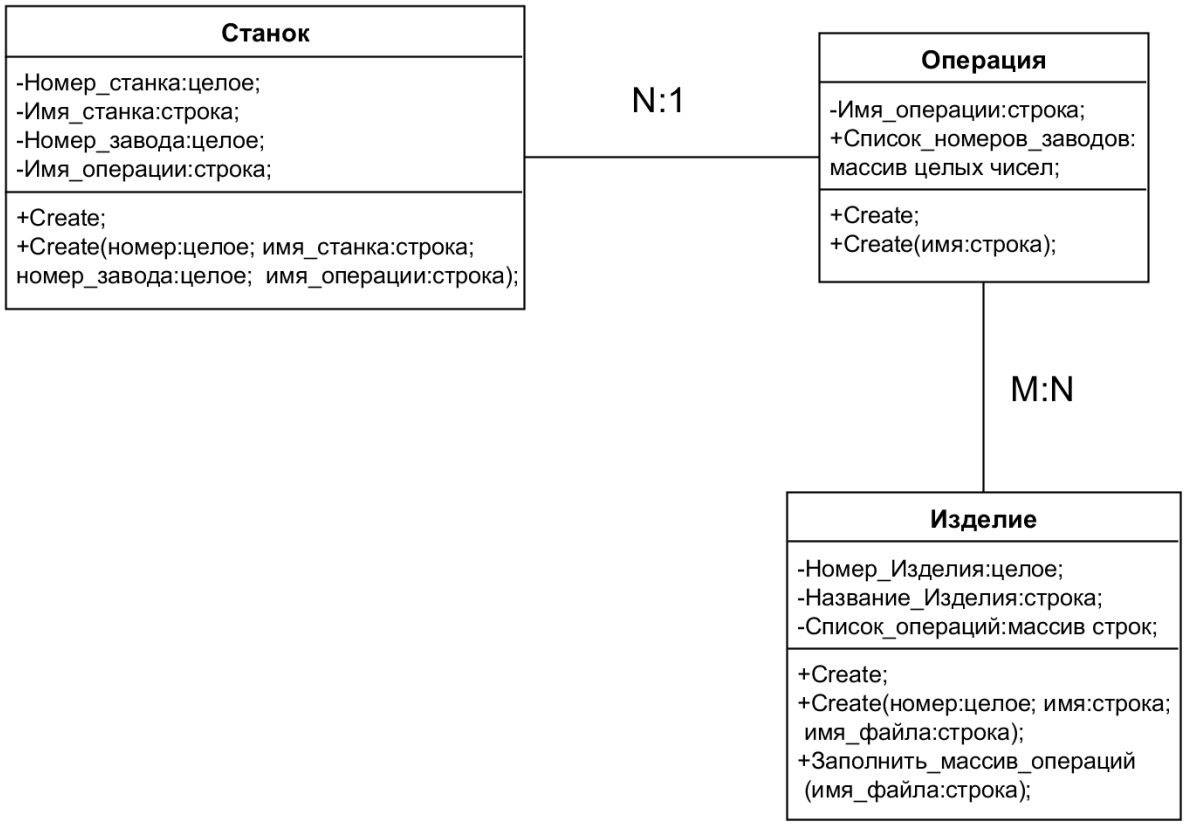

Рисунок 1 – Диаграмма классов

1.3 UML-диаграмма

В соответствии с составленным алгоритмом была сделана UML-диаграмма вариантов использования программы с инструкцией по их реализации пользователем (рисунок 2).

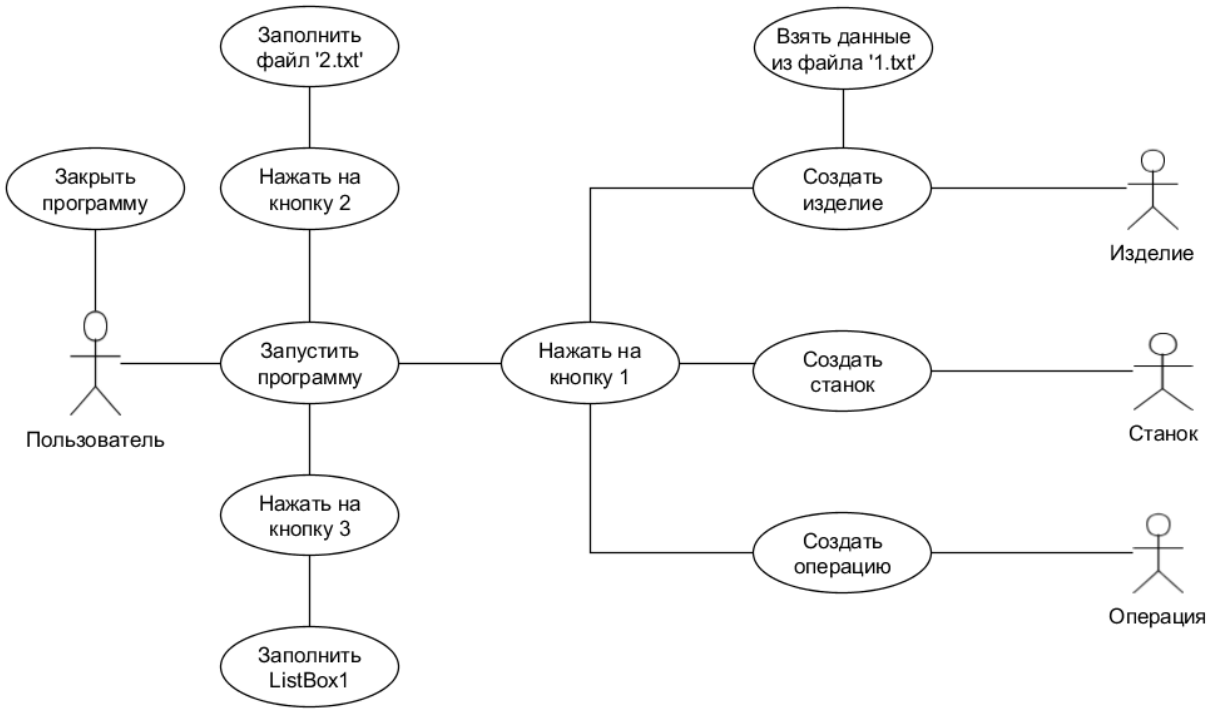

Рисунок 2 – UML-диаграмма

2 Блок-схемы

На основе написанного алгоритма и построенной диаграммы классов были составлены блок-схемы наиболее значимых методов и процедур, используемых в программе (рисунки 3-8).

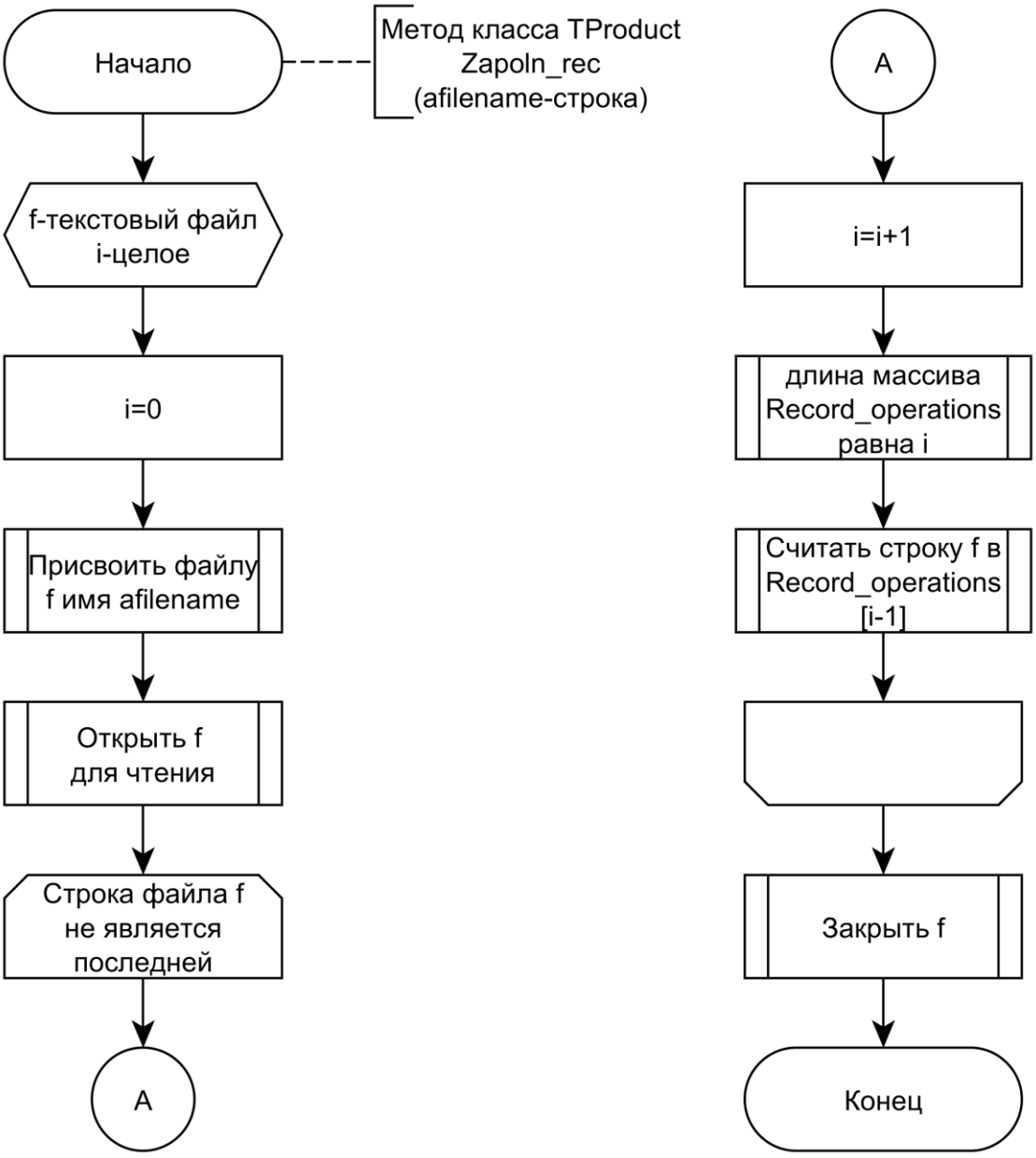

Рисунок 3 – Блок-схема метода «Заполнить_массив_операций» класса «Изделие»

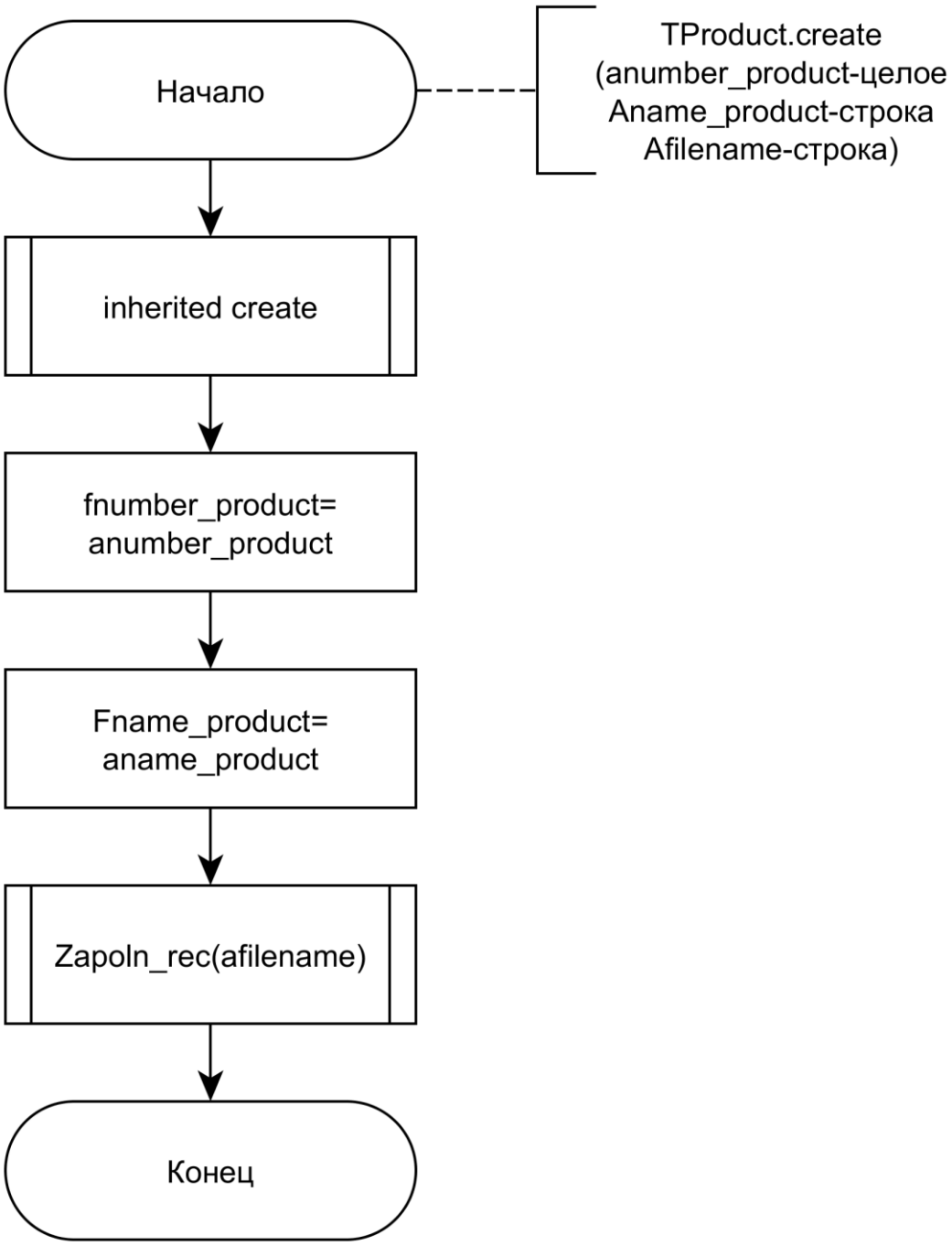

Рисунок 4 – Блок-схема конструктора «Create» класса «Изделие»

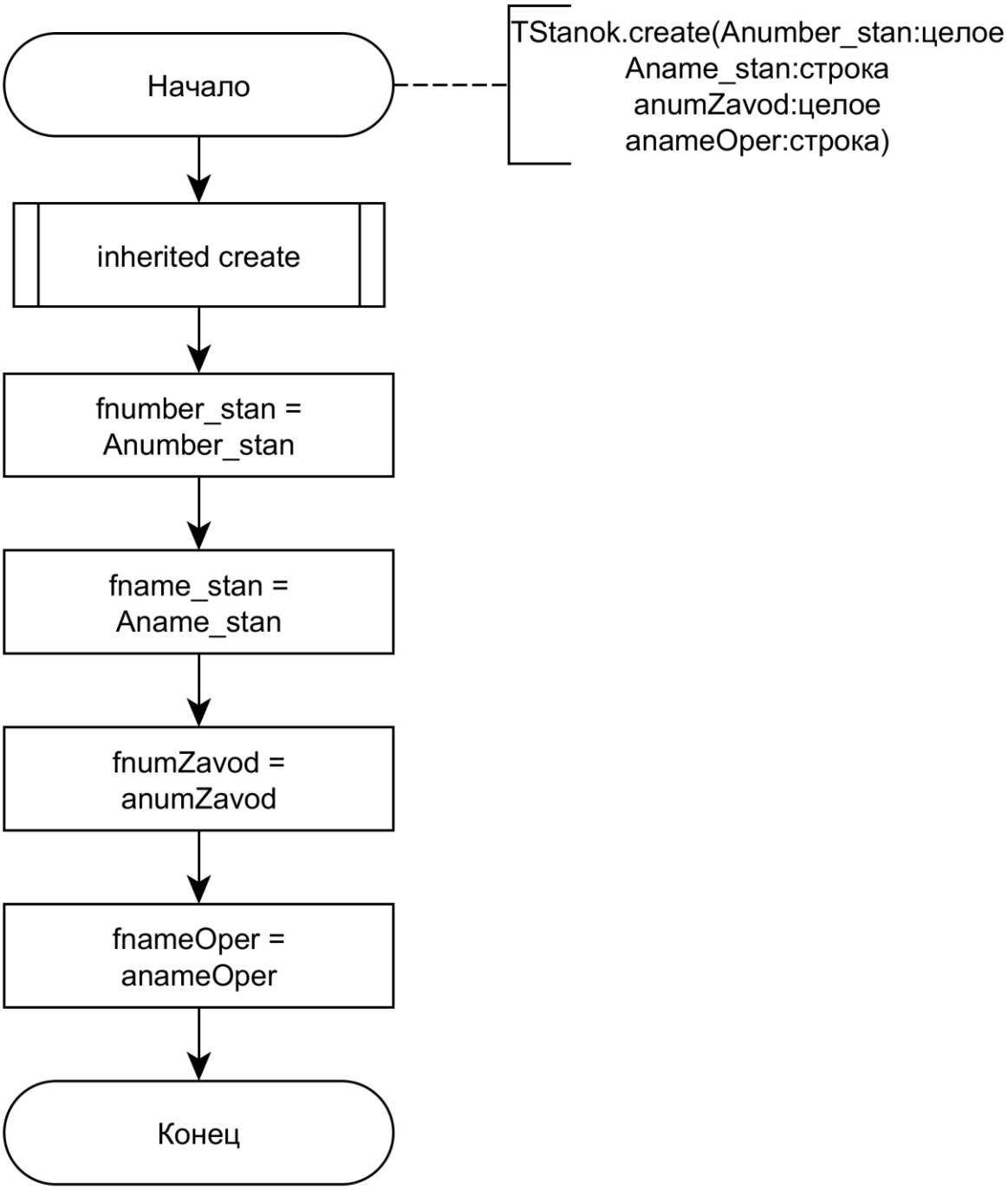

Рисунок 5 – Блок-схема конструктора «Create» класса «Станок»

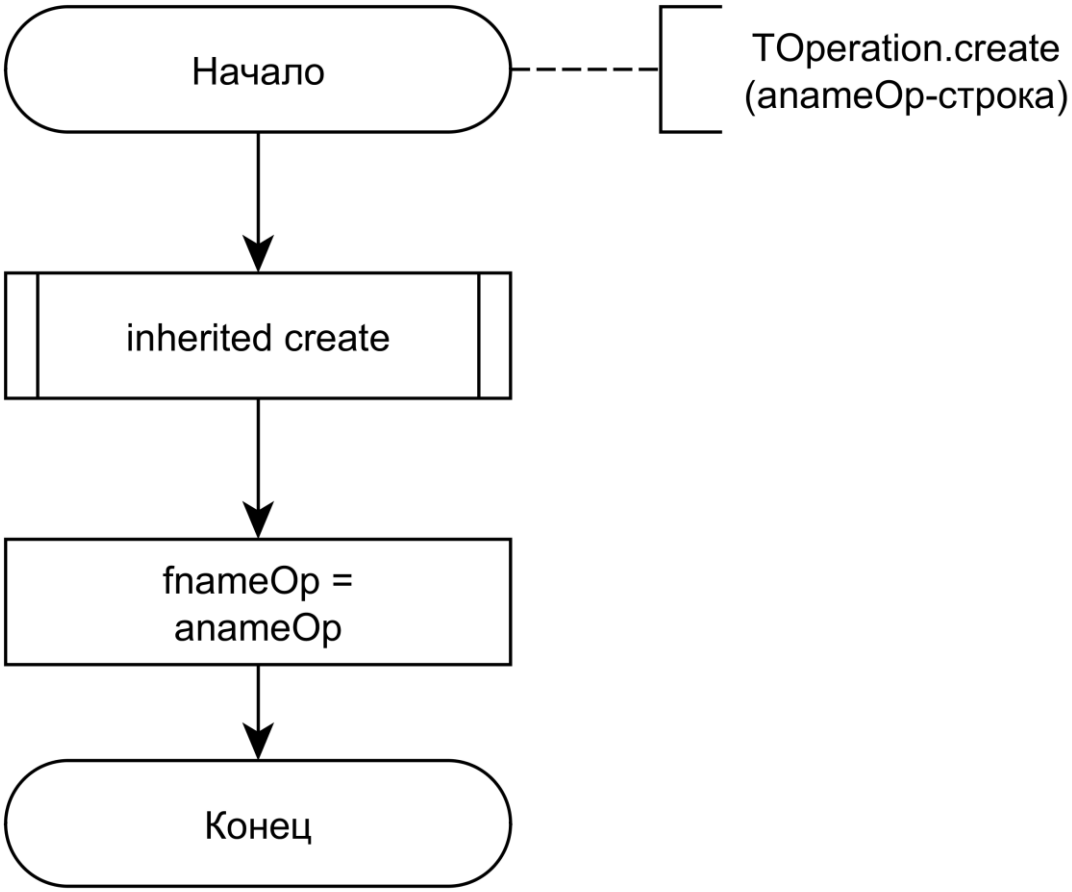

Рисунок 6 – Блок-схема конструктора «Create» класса «Операция»

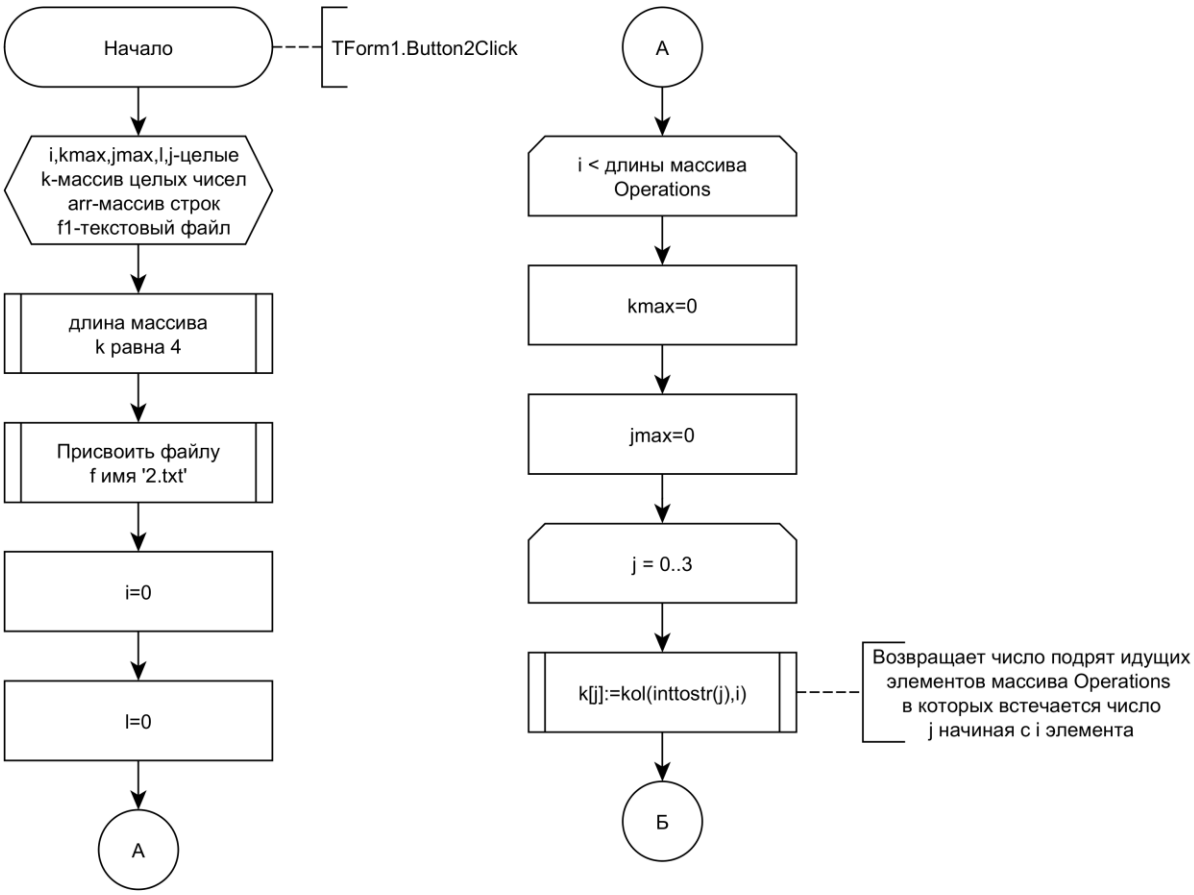

Рисунок 7 – Блок-схема процедуры нажатия на кнопку 2 на форме 1

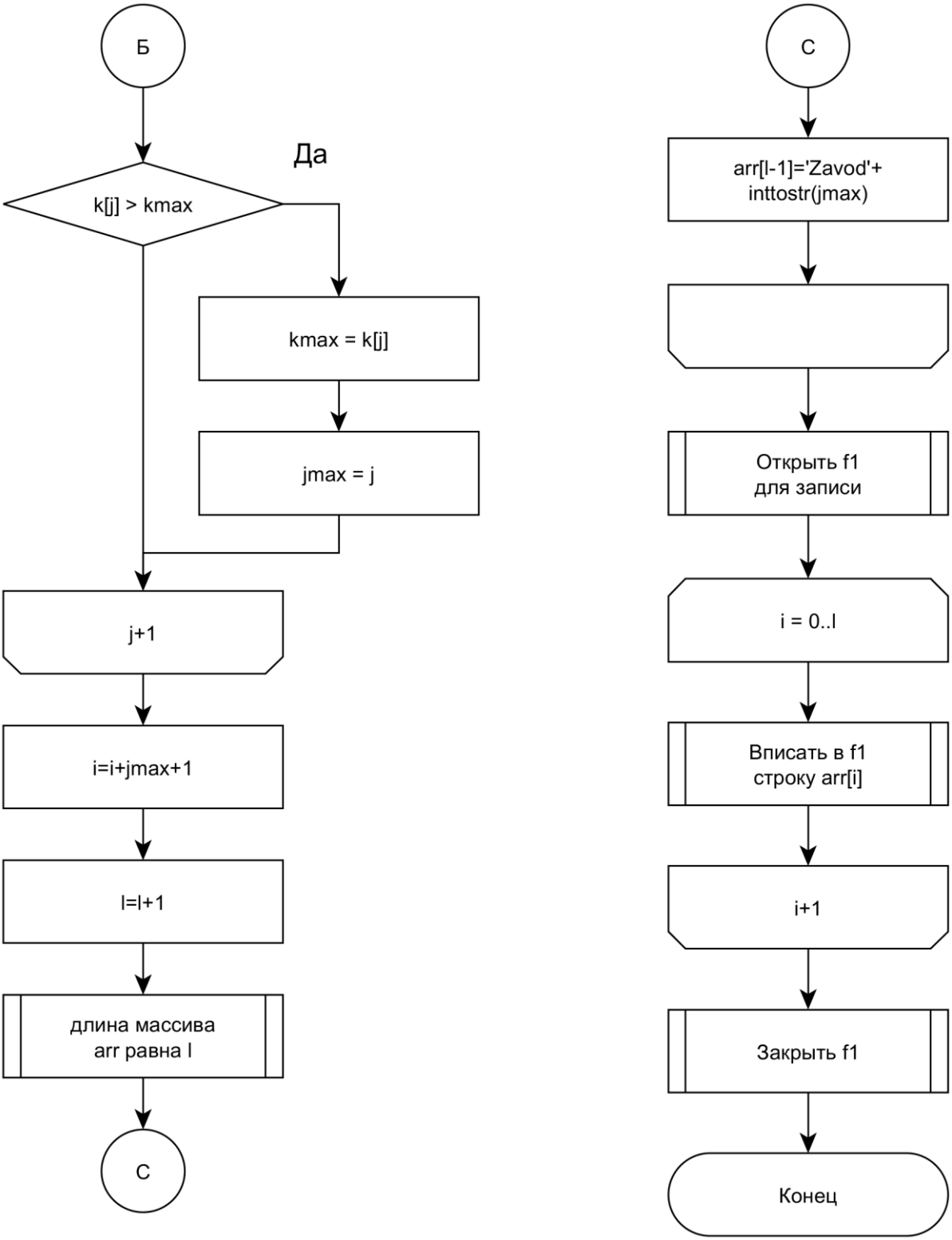

Рисунок 8 – Блок-схема процедуры нажатия на кнопку 2 на форме 1

3 Технологическая часть

3.1 Приложение

По написанному алгоритму, составленным диаграмме классов и блок-схемам было сделано приложение в программе Lazarus. Код программы представлен на листинге 1. Внешний вид формы представлен на рисунке 9.

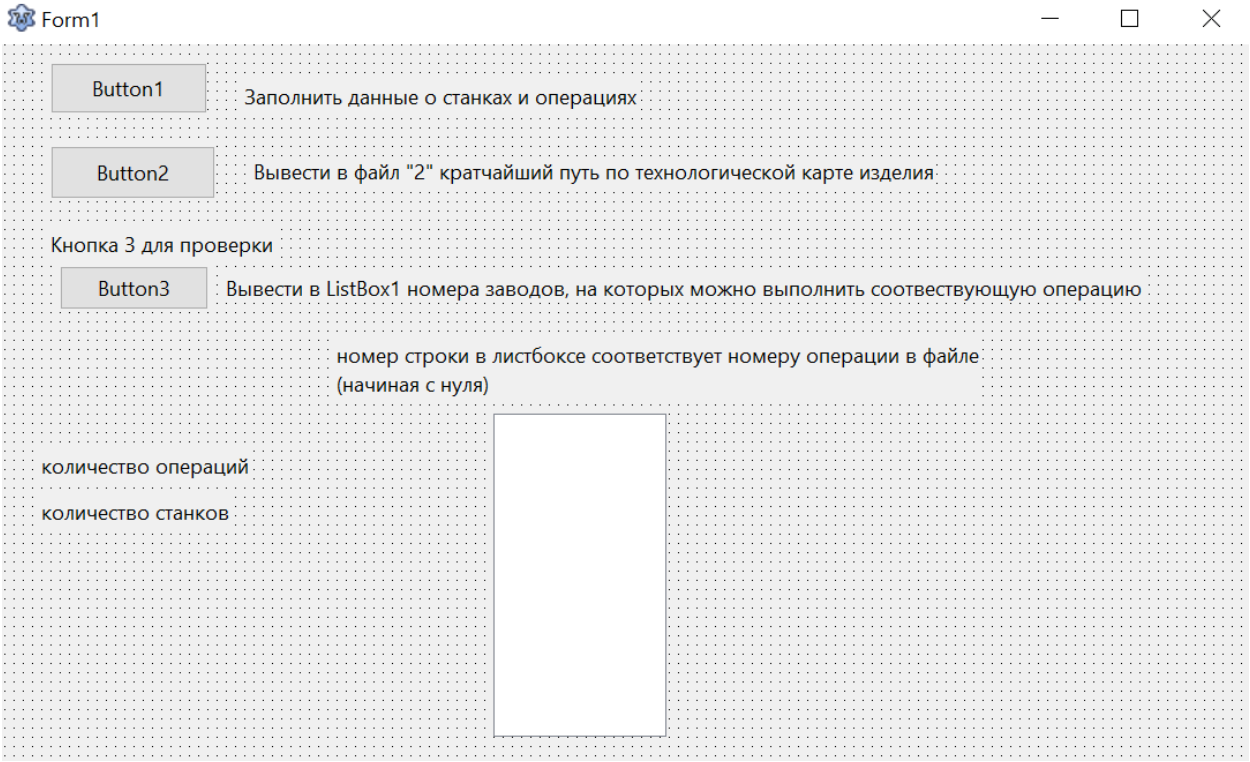

Рисунок 9 – Внешний вид формы 1

Листинг 1 – код программы решения задачи

```
unit Unit1;

{$mode objfpc}{$H+}

interface

uses
  Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls;

type

  { TForm1 }

  TForm1 = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
```

```
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    ListBox1: TListBox;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
  private

  public

  end;
type
  tmas=array of integer;
type Tproduct = class(Tobject)
private
   fnumber_product:integer;
   Fname_product:string;
   Record_operations:array of string;
public
   constructor create(anumber_product:integer; aname_product:string;
afilename:string);
   procedure Zapoln_rec(afilename:string);
end;
type TStanok = class(Tobject)
private
   fnumber_stan:integer;
   Fname_stan:string;
   fnumZavod:integer;
   fnameOper:string;
public
   constructor create(Anumber_stan:integer; Aname_stan:string;
anumZavod:integer; anameOper:string);
end;
type
  TStanoks=array of  TStanok;
type TOperation = class(Tobject)
private
  fnameOp:string;
public
  fRecZav:tmas;
  constructor create(anameOp:string);
end;
TArrayString=array of string;
TArrayOperation=array of TOperation;
function kol(s:string; i:integer):integer;
var
  Form1: TForm1;
  Stanoks:TStanoks;
  Operations:TArrayOperation;
implementation

{ TForm1 }

procedure TForm1.Button1Click(Sender: TObject);
var
  product:Tproduct;
  n,m,a,b,i,j,l:integer;
begin
  product:=Tproduct.create(0,'изделие', '1.txt');
  n:=length(product.Record_operations);
  label4.Caption:='Количество операций '+inttostr(n);
```

```
  m:=random(7)+n;
  label7.Caption:='Количество станков '+inttostr(m);
  setlength(Stanoks,m);
  setlength(Operations,n);
  i:=0;
  while i < n do
  begin
    a:=random(m);
    if Stanoks[a] = nil then
    begin
      b:=random(4);

stanoks[a]:=TStanok.create(a,'Stanok'+inttostr(a),b,product.Record_operation
s[i]);
      i:=i+1;
    end;
  end;
  for i:=0 to m-1 do
  begin
    if Stanoks[i] = nil then
    begin
      b:=random(4);
      a:=random(n);

stanoks[i]:=TStanok.create(i,'Stanok'+inttostr(i),b,product.Record_operation
s[a]);
    end;
  end;
  for i:=0 to n-1 do
  begin
    l:=0;
    Operations[i]:=TOperation.create(product.Record_operations[i]);
    for j:=0 to m-1 do
    begin
      if Stanoks[j].fnameOper = product.Record_operations[i] then
      begin
         l:=l+1;
         setlength(Operations[i].fRecZav,l);
         Operations[i].fRecZav[l-1]:=Stanoks[j].fnumZavod;
      end;
    end;
  end;
end;

procedure TForm1.Button2Click(Sender: TObject);
var
  i,kmax,jmax,l,j:integer;
  k:array of integer;
  arr:TArrayString;
  f1:text;
begin
  setlength(k,4);
  assignfile(f1,'2.txt');
  i:=0;
  l:=0;
  while i < length(Operations) do
  begin
    kmax:=0;
    jmax:=0;
    for j:=0 to 3 do
    begin
      k[j]:=kol(inttostr(j),i);
      if k[j] > kmax then
```

```
      begin
        kmax:=k[j];
        jmax:=j;
      end;
    end;
    i:=i+jmax+1;
    l:=l+1;
    setlength(arr,l);
    arr[l-1]:='Zavod'+inttostr(jmax);
  end;
  rewrite(f1);
  for i:=0 to l do
    writeln(f1,arr[i]);
  closefile(f1);
end;

procedure TForm1.Button3Click(Sender: TObject);
var
   j,k:integer;
   s:string;
begin
   for j:=0 to length(Operations)-1 do
   begin
     s:='';
     for k:=0 to length(Operations[j].fRecZav)-1 do
     begin
       s:=s+inttostr(Operations[j].fRecZav[k]);
       listbox1.Items[j]:=s;
     end;
   end;
end;

{ TForm1 }



procedure Tproduct.Zapoln_rec(afilename:string);
var
  f:text;
  i:integer;
begin
  i:=0;
  assignfile(f,afilename);
  reset(f);
  while not seekeof(f) do
  begin
    i:=i+1;
    setlength(Record_operations,i);
    readln(f,Record_operations[i-1]);
  end;
  closefile(f);
end;
constructor Tproduct.create(anumber_product:integer; aname_product:string;
afilename:string);
begin
  inherited create;
  fnumber_product:=anumber_product;
  Fname_product:=aname_product;
  Zapoln_rec(afilename);
end;
constructor TStanok.create(Anumber_stan:integer; Aname_stan:string;
anumZavod:integer; anameOper:string);
begin
```

```
  inherited create;
  fnumber_stan:=Anumber_stan;
  fname_stan:=Aname_stan;
  fnumZavod:=anumZavod;
  fnameOper:=anameOper;
end;
constructor TOperation.create(anameOp:string);
begin
 inherited create;
 fnameOp:=anameOp;
end;
function kol(s:string; i:integer):integer;
var
  Ars:TArrayString;
  j,k:integer;
begin
   setlength(Ars,length(Operations)+1);
   for j:=i to length(Operations)-1 do
   begin
     Ars[j]:='';
     for k:=0 to length(Operations[j].fRecZav)-1 do
     begin
       Ars[j]:=Ars[j]+inttostr(Operations[j].fRecZav[k]);
     end;
   end;
   j:=i;
   k:=0;
   while (pos(s,Ars[j]) <> 0) and (j<length(Ars)) do
   begin
      j:=j+1;
      k:=k+1;
   end;
   kol:=k;
end;

{$R *.lfm}

end.
```

3.2 Сравнение полученного и ожидаемого результатов

На рисунке 10 приведён пример содержимого файла «1.txt» со списком операций.

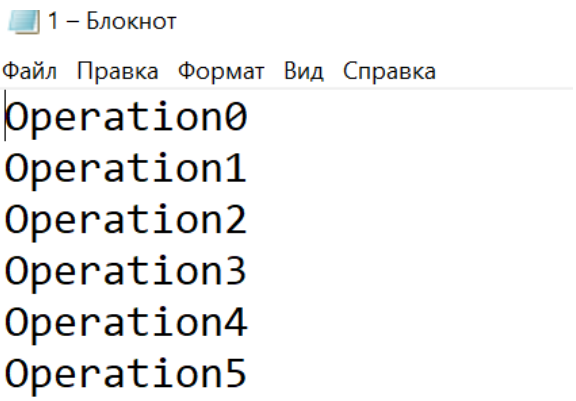

Рисунок 10 – пример содержимого файла «1.txt»

Вид приложения после последовательного нажатия на три кнопки расположенные на форме представлен на рисунке 11

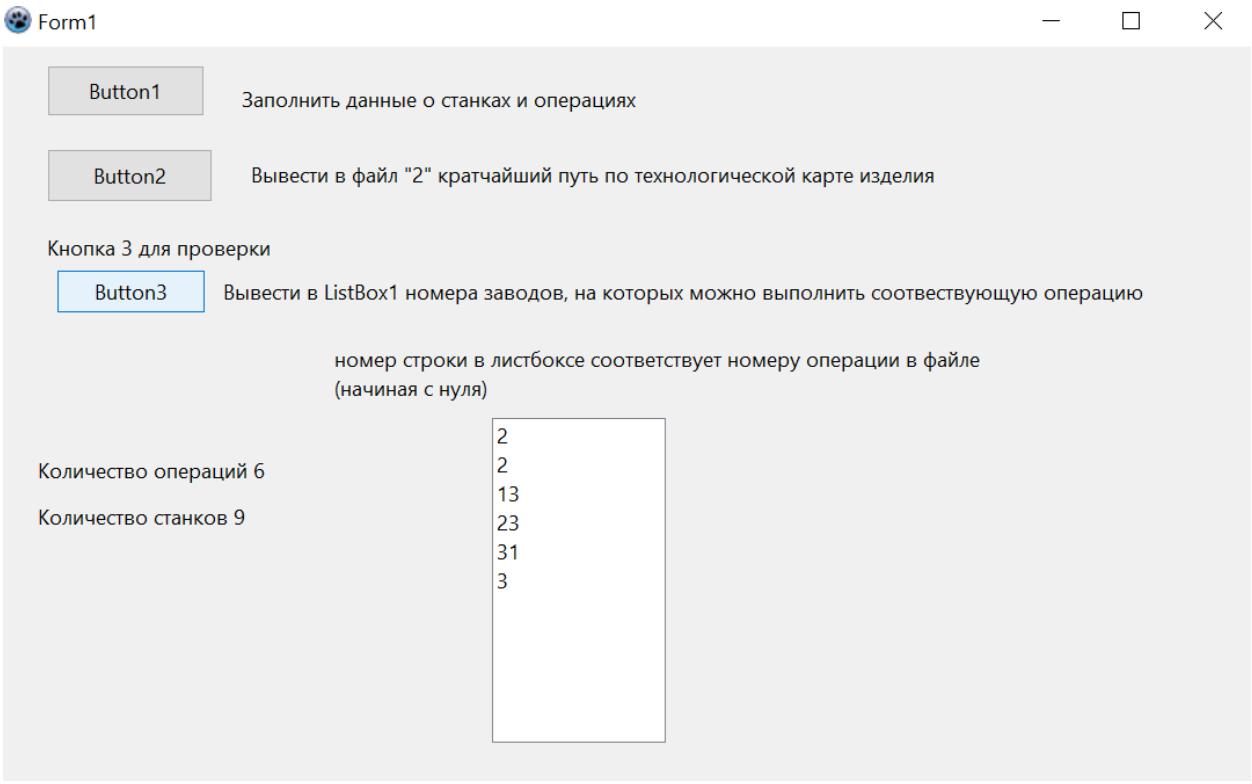

Рисунок 11 – Вид приложения после последовательного нажатия на три кнопки

Содержимое файла «2.txt» с кратчайшей последовательностью посещения изделием заводов представлено на рисунке 12.

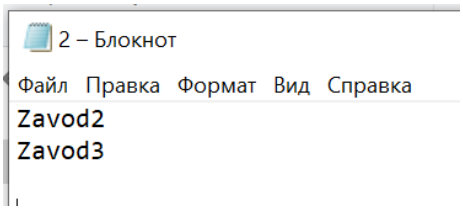

Рисунок 12 – Содержимое файла «2.txt»

Полученный результат совпадает с ожидаемым.

Для более тщательной проверки была составлена таблица ожидаемого и полученного значений (таблица 1).

Таблица 1 – сравнение ожидаемого результата и результата, полученного в ходе работы программы

Содержимое файла «1.txt»	Содержимое файла «1.txt» (результат, полученный в ходе работы программы)	Ожидаемый результат
OperationA OperationB OperationC OperationD OperationE	Zavod2 Zavod1	Zavod2 Zavod1
Operation0 Operation1 Operationa Operationb Operationc Operation5	Zavod2 Zavod3	Zavod2 Zavod3
Operation0 Operation1 Operation2	Zavod2	Zavod2

Программа работает исправно во всех случаях.

ЗАКЛЮЧЕНИЕ

В ходе данной лабораторной работы было составлено приложение по выданному заданию, были выполнены все поставленные задачи:

1) Был составлен алгоритм решения задачи;

2) Были выбраны сущности, их методы и атрибуты, необходимые для решения задачи;

3) Была составлена диаграмма классов на основе выбранных сущностей;

4) Была построена UML-диаграмма вариантов использования программы с инструкцией по их реализации пользователем;

5) Были составлены блок-схемы наиболее значимых процедур и методов;

6) По составленным блок-схемам был написан код программы;

7) Были сравнены результат, полученный с помощью программы, и ожидаемый результат;

Программа работает исправно во всех случаях.