Пермский филиал Федерального государственного автономного образовательного учреждения высшего образования

«Национальный исследовательский университет «Высшая школа экономики»

Факультет социально-экономических и компьютерных наук

«Программная инженерия» .

(Название ОП)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_бакалавриат\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(уровень образования)

**О Т Ч Е Т**

**по практике**

Вид практики: профессиональная

Тип практики: производственная

Выполнил студент гр.ПИ-21СВ

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

(ФИО)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

*(подпись)*

**Проверили:**

доцент кафедры информационных технологий в бизнесе*,* Шестакова Л.В.

*(должность, ФИО руководителя от* *НИУ ВШЭ – Пермь)*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(оценка по (подпись)*

*10-балльной шкале)*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(дата)*

Пермь 2023

**Оглавление**

[**Введение** 3](#_Toc139282792)

[**1.** **Описание структуры АО «Полиэкс»** 4](#_Toc139282793)

[**2.** **Описание проблемы, требующей решения** 6](#_Toc139282794)

[**3.** **Моделирование бизнес-процесса** 8](#_Toc139282795)

[**4.** **Выбор алгоритма** 9](#_Toc139282796)

[**5.** **Разработка алгоритма на основе метода ветвей и границ** 10](#_Toc139282797)

[**6.** **Реализация программы** 13](#_Toc139282798)

[ПРИЛОЖЕНИЕ А **Диаграмма бизнес-процесса «Выполнение химического исследования»** 14](#_Toc139282799)

[ПРИЛОЖЕНИЕ Б **Листинги программного кода** 15](#_Toc139282800)

# **Введение**

Местом прохождения производственной практики являлась компания «Полиэкс», а именно её структурное подразделение – химическая лаборатория Технической службы.

АО «Полиэкс» специализируется в области химических технологий и реагентов для повышения нефтеотдачи пластов (ПНП), текущего и капитального ремонта нефтегазодобывающих и нагнетательных скважин, а также реагентов для процессов добычи и транспорта нефти и газа.

Химическая лаборатория Технической службы занимается исследовательскими и аналитическими работами с целью разработки новой продукции, её усовершенствования и постановки на производство, а также техническим сопровождением других служб в составе компании (служба продаж, сервиса, по организации производства, коммерческой и др.) или компаний‑заказчиков.

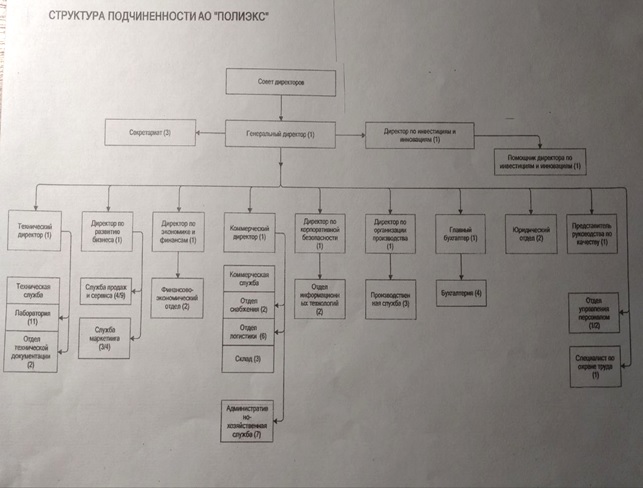
Цель практики – автоматизация деятельности структурного подразделения компании «Полиэкс».

Для достижения поставленной цели необходимо решить ряд задач.

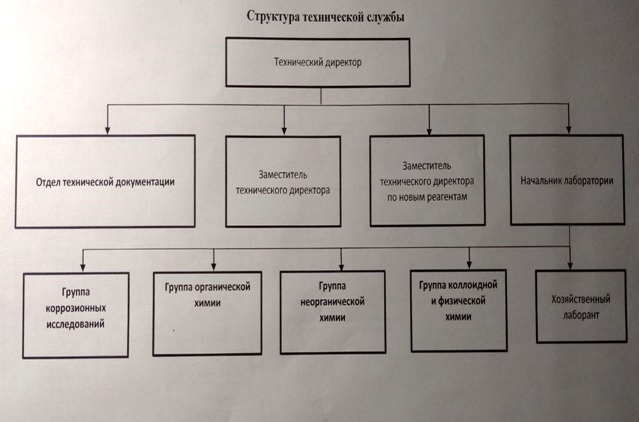
1. Описать организационную структуру компании «Полиэкс».
2. Выбрать бизнес-процесс, имеющий нерешенную проблему и описать эту проблему.
3. Смоделировать выбранный бизнес-процесс с помощью диаграммы в нотации BPMN.
4. Выбрать алгоритм для автоматизации выбранного бизнес-процесса и обосновать этот выбор.
5. Разработать алгоритм для реализации в программе.
6. Реализовать программу для автоматизации выбранного бизнес-процесса.

# **Описание структуры АО «Полиэкс»**

Структура компании «Полиэкс» представлена на рисунке 1. Одним из подразделений данной структуры является техническая служба, которая состоит из двух структурных групп: лаборатории и отела технической документации. Структура технической службы представлена на рисунке 2.



***Рисунок 1 – Организационная структура АО «Полиэкс»***



***Рисунок 2 – Организационная структура Технической службы АО «Полиэкс»***

# **Описание проблемы, требующей решения**

Входящая в состав химической лаборатории, группа по реагентам органической химии получает заказы на проведение химических экспериментов от различных внутренних заказчиков (службы продаж, сервиса и др., а также технического директора или начальника лаборатории).

Проводимые группой работы обладают разным приоритетом, в зависимости от степени их важности и срочности. Цели, достигаемые с помощью поставленных экспериментов, как правило придают им один и тот же стандартный приоритет. Например, проверка образцов сырья на применимость в производстве того или иного реагента, обычно имеет самый высокий приоритет, а текущая исследовательская деятельность – самый низкий. Приоритет работы может быть изменен руководством лаборатории в зависимости от сложившейся ситуации. В таблице 1 описана зависимость приоритета работы от её целей. Для удобства разработки алгоритма автоматизации вводится понятие «ранга» – величины, определяющей ценность работы.

Таблица 1 – Зависимость приоритета работы от её целей

|  |  |  |  |
| --- | --- | --- | --- |
| **Цель работы** | **Внутренний заказчик** | **Стандартный приоритет** | **Ранг** |
| Проверка образца сырья на пригодность в производстве реагента. | Служба по организации производства. Отдел снабжения коммерческой службы. | 1 | 5 |
| Получение исследовательской информации в рамках НИОКР. | Технический совет | 2 | 4 |
| Подбор технического решения проблемы, возникшей при применении реагента на стороне компании-заказчика. | Служба сервиса. | 3 | 3 |
| Приготовление образца реагента для отправки компании-заказчику. | Служба продаж. | 4 | 2 |
| Текущая исследовательская деятельность. | Технический директор.  Начальник лаборатории. | 5 | 1 |

При этом, для проведения химического эксперимента необходимо разместить на рабочем месте лабораторную установку, собранную из стеклянной посуды и оборудованную приборами для перемешивания, нагрева, охлаждения, подачи или отвода различных газов и т.д. Как правило, размер такой установки зависит от объема реакционной массы, вязкости среды (т.е. от размеров оборудования, необходимого для перемешивания). Необходимость проведения отгонки растворителя от реакционной массы требует размещения дополнительного оборудования для перегонки. В таблице 2 описаны варианты лабораторных установок с указанием ширины рабочей поверхности, занимаемой такой установкой.

Таблица 2 – Варианты лабораторных установок

|  |  |
| --- | --- |
| **Вариант лабораторной установки** | **Ширина, м** |
| Синтез в колбах, объёмом 50-250 мл, с применением колбонагревателя с магнитным перемешиванием. | 0,5 |
| Синтез в колбах, объёмом 50-2000 мл с применением верхнеприводной мешалки и нагревающей бани. | 0,75 |
| Синтез в колбах, объёмом 4000-6000 мл специального колбонагревателя и верхнеприводной мешалки. | 1 |
| Синтез с применением оборудования для перегонки. | дополнительно 0,5 |

Таким образом, лабораторная установка, в зависимости от характера эксперимента, может занимать 0,5; 0,75; 1,0; 1,25 или 1,5 метра.

Помещение группы по реагентам органической химии оборудовано вытяжным шкафом, ширина рабочей поверхности которого составляет 6 метров. Специфика экспериментов с применением органических химических реагентов не позволяет проводить их в помещениях других групп в составе лаборатории в связи с токсичностью, пожароопасностью и взрывоопасностью проводимых процессов.

Каждый эксперимент занимает по времени одну рабочую смену. Сотрудники группы могут проводить несколько экспериментов одновременно.

Имея определенный пул задач на момент начала смены, руководитель группы должен выбрать из них такой список задач, который позволит провести смену с наибольшей пользой.

Отсутствие автоматизированного алгоритма для выбора оптимального списка задач может привести к регулярной потере эффективности группы и, как следствие, несвоевременному решению острых проблем и накоплению нерешенных задач.

# **Моделирование бизнес-процесса**

Бизнес-процесс «Выполнение химического исследования» начинается с поступления в лабораторию технической службы документа-обоснования на проведение работ. Таким документом могут быть: заявка на испытание образца от службы по организации производства или от отдела снабжения коммерческой службы; служебная записка или техническое задание от службы сервиса и т.д. Также процесс «выполнения химического исследования» может быть инициирован устным распоряжением Технического директора или начальника лаборатории.

Начальник лаборатории может изменить приоритет поступившей задачи в зависимости от сложившейся оперативной ситуации. Руководитель группы по реагентам органической химии в начале смены выбирает из сложившегося пула задач список задач на сегодня и распределяет их среди исполнителей. Сам синтез проводится химиком-синтетиком, а свободный на данный момент химик-аналитик сопровождает синтез проведением анализов сырья и реакционной массы на полноту прохождения синтеза. Если синтез не прошел до конца, то химик-синтетик продолжает его.

По окончании синтеза химик-синтетик формирует отчет о проделанной работе включая в него данные анализов. Отчет согласуется руководителем группы и начальником лаборатории и отправляется на ознакомление внутреннему заказчику данной работы. При согласовании отчета руководство лаборатории может вернуть задачу на доработку.

Бизнес-процесс «Выполнение химического исследования» смоделирован в диаграмме BPMN, вынесенной в Приложение А.

# **Выбор алгоритма**

Решением, описанной в главе 1, проблемы должен быть наиболее оптимальный набор задач, взятый для решения исследовательской группой в рамках одной рабочей смены. Лабораторные установки для такого набора задач должны помещаться на имеющейся рабочей поверхности. Суммарная польза, которую компания получает в результате выполнения этих задач, оценивается суммарным «рангом» решаемых за смену задач.

Описанная проблема является классической «задачей о рюкзаке», где за «ёмкость рюкзака» можно принять ширину рабочей поверхности вытяжного шкафа, за «объём/вес предмета» – ширину лабораторной установки для решения конкретной задачи, а за «стоимость предмета» – «ранг» этой задачи. Таким образом, искомым ответом при решении данной задачи будем считать такой список задач из известного пула задач, который будет обладать наибольшим суммарным «рангом».

Для решения поставленной задачи достаточно применения «метода ветвей и границ». Этот метод (без дополнительных улучшений) успешно применяется на практике, когда он запускается многократно на разных наборах входных данных, т.к. «неудачные» наборы входных данных встречаются крайне редко, а в большинстве случаев алгоритм находит правильное решение гораздо быстрее, чем метод полного перебора.

Сложность данного алгоритма можно оценить, как О(2n). Но поставленная задача должна решаться многократно в процессе деятельности исследовательской группы, а объемы входных данных не так уж велики, чтобы использовать дополнительные улучшения для этого алгоритма (методами динамического программирования). Поэтому применение метода ветвей и границ, наряду с автоматизацией вычислений, значительно увеличит эффективность работы исследовательской группы.

# **Разработка алгоритма на основе метода ветвей и границ**

Входные данные для разрабатываемой программы состоят из общей ширины рабочей зоны (по умолчанию **SumWidth** = 6) и набора объектов класса Experiment. Каждый объект класса Experiment содержит в себе данные о названии эксперимента (**Title**), «ранге» (**Rank**) и ширине (**Width**).

После запуска программы, с помощью цикла со счетчиком, для всех объектов класса **Experiment**, входящих в исходный список, дополнительно вычисляется параметр **Profit** = Rank / Width. Далее список экспериментов сортируется по этому параметру, и начинается построение сортированного списка объектов структуры **Node**. Этот список будет хранить в себе информацию о «листьях» строящегося бинарного дерева. Т.к. информация об узлах дерева, не являющихся его «листьями», не имеет никакой ценности, такие узлы будут удаляться после построения на их основе двух новых «листьев».

Структура Node содержит в себе информацию:

* об индексе (**I**) текущего эксперимента в списке Experiments, решение о принятии которого в оптимальный список в этом узле принимается;
* список номеров экспериментов (**ExpList**), которые уже приняты в оптимальный список;
* ожидаемая суммарная ценность (**Result**) списка экспериментов, в случае продвижения дальше по этой ветви бинарного дерева.

В сортированный список объектов Node добавляется первый элемент, являющийся «корнем» формируемого бинарного дерева. Для него индекс текущего эксперимента равен 0, а список номеров уже принятых экспериментов пуст.

Запускается «вечный» цикл. В начале каждого его витка выделяется «максимальный» объект Node. Компаратор объектов Node выделяет «максимальный» объект **maxNode** из списка по следующим параметрам.

1. Ожидаемая суммарная ценность (**Result**).
2. Длина списка номеров отобранных экспериментов (**ExpList.Count**).
3. Индекс текущего эксперимента (**I**).
4. Сумма индексов в списке номеров отобранных экспериментов (**Sum(ExpList)**).

Создаются и добавляются в сортированный список **nodes** два новых объекта:

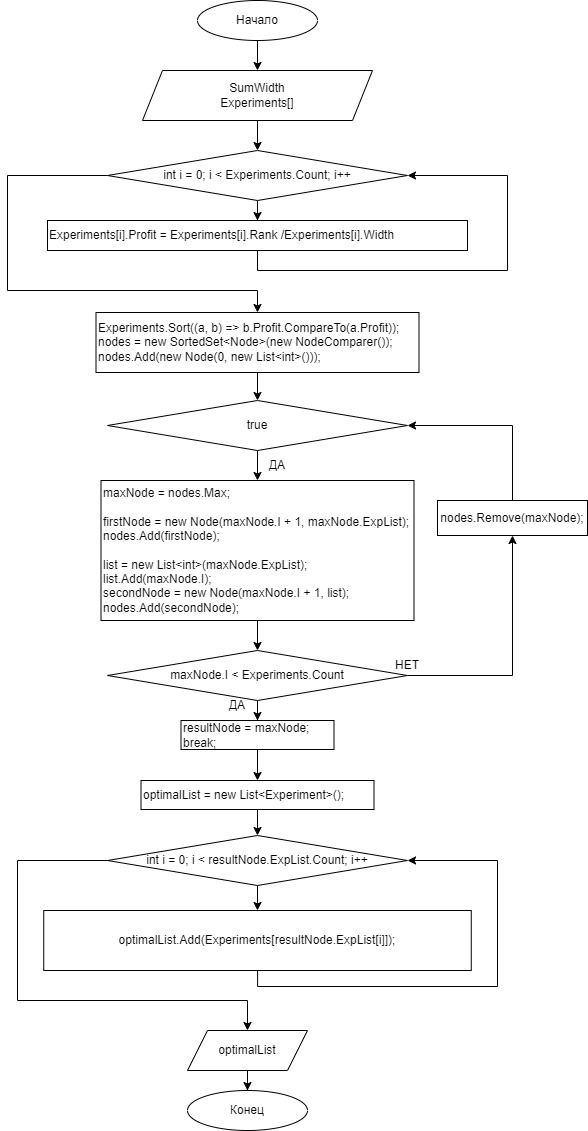
* **firstNode**, для которого индекс **I** увеличивается на 1, а список ExpList передается из текущего **maxNode**;
* **secondNode**, для которого индекс **I** увеличивается на 1, а список ExpList, передается из текущего **maxNode** после добавления к нему индекса **I**.

По сути **firstNode** – это узел, описывающий случай, когда текущий эксперимент не отбирается в оптимальный список, а **secondNode** – когда он отбирается.

Если на очередном витке «вечного» цикла индекс «максимального» объекта **maxNode** достигает длины списка экспериментов (**maxNode.I < Experiments.Count**), этот объект присваивается в переменную **resultNode** и цикл завершается с помощью оператора **break**. Иначе, объект **maxNode**, на основе которого строились новые «листья» бинарного дерева удаляется за ненадобностью из списка **nodes**.

По окончании «вечного» цикла формируется оптимальный список экспериментов **optimalList** на основе отобранного списка номеров экспериментов **resultNode.ExpList**. Список **optimalList** выводится в качестве ответа в новом окнеи программы.

Упрощенная блок-схема разработанного алгоритма приведена на рисунке 3.



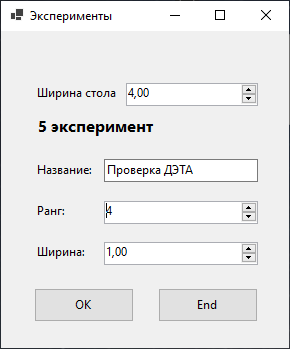
***Рисунок 3 – Упрощенная блок-схема алгоритма***

# **Реализация программы**

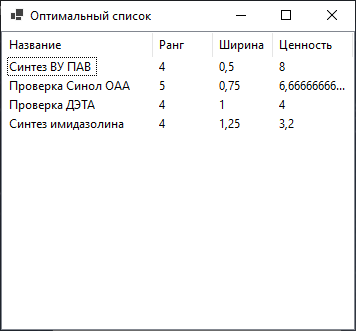
Реализованная программа представляет собой приложение оконного типа (WinForms). Оно содержит файлы окон **MainForm.cs** и **OptimalListForm.cs**, а также файлы классов **Experiment.cs**, **Node.cs**, **NodeComparer.cs** и **Solution.cs**.

Текст программы приведен в листингах в Приложении Б.

Пример работы программы изображен на рисунках 4-5.

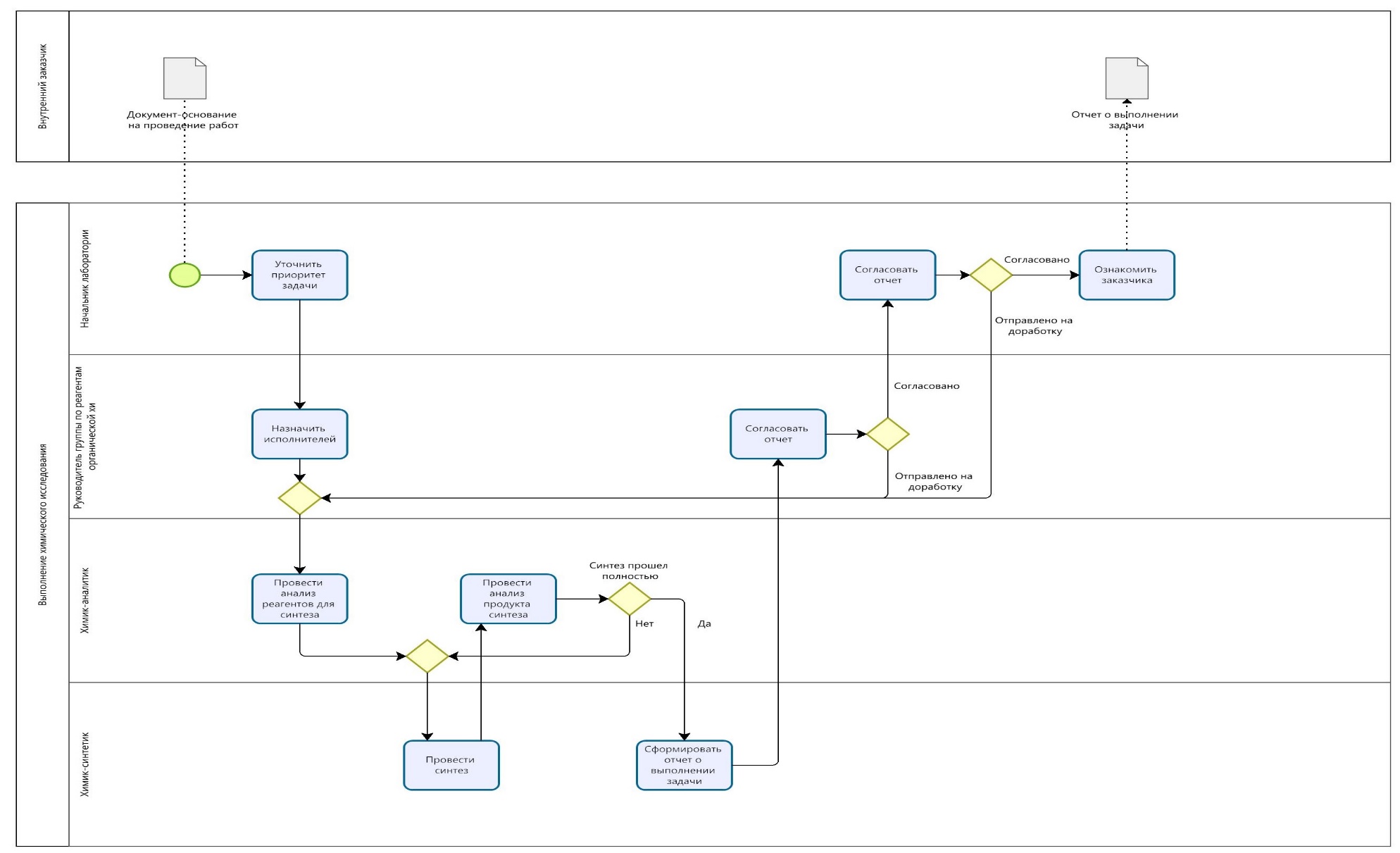


***Рисунок 4 – Пример окна ввода данных***

******

***Рисунок 5 – Пример окна вывода данных***

ПРИЛОЖЕНИЕ А  
**Диаграмма бизнес-процесса «Выполнение химического исследования»**



ПРИЛОЖЕНИЕ Б  
**Листинги программного кода**

**MainForm.cs**

public partial class MainForm : Form

{

private List<Experiment> Experiments = new List<Experiment>();

private int nums = 1;

public MainForm()

{

InitializeComponent();

}

private void btnOK\_Click(object sender, EventArgs e)

{

if (!string.IsNullOrEmpty(txtTitle.Text))

{

Experiment exp = new Experiment();

exp.Title = txtTitle.Text;

exp.Rank = (int)numRank.Value;

exp.Width = (double)numWidth.Value;

Experiments.Add(exp);

label1.Text = (Experiments.Count + 1).ToString() + label1.Text.Substring(nums);

if (Experiments.Count == 9)

nums = 2;

txtTitle.Clear();

numRank.Value = 1;

numWidth.Value = 0.5M;

}

else

{

MessageBox.Show("Введите название эксперимента", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

private void btnEnd\_Click(object sender, EventArgs e)

{

btnOK.Enabled = false;

Solution solution = new Solution(Experiments, (double)numSumWidth.Value);

OptimalListForm form = new OptimalListForm(solution.GetOptimalList());

form.Show();

}

}

**OptimalListForm.cs**

public partial class OptimalListForm : Form

{

private List<Experiment>? Experiments = null;

public OptimalListForm(List<Experiment> experiments)

{

InitializeComponent();

Experiments = experiments;

FillListView();

}

private void FillListView()

{

foreach (var exp in Experiments)

{

ListViewItem item = new ListViewItem(exp.Title);

item.SubItems.Add(exp.Rank.ToString());

item.SubItems.Add(exp.Width.ToString());

item.SubItems.Add(exp.Profit.ToString());

listView1.Items.Add(item);

}

}

}

**Experiment.cs**

public class Experiment

{

public string Title { get; set; }

public int Rank { get; set; }

public double Width { get; set; }

public double Profit { get; set; }

public void SetProfit()

{

Profit = Rank / Width;

}

}

**Node.cs**

public struct Node

{

public int? Id = null;

public int I;

public List<int> ExpList;

public double Result;

public Node(int id, int i, List<int> expList, double result)

{

Id = id;

I = i;

ExpList = expList;

Result = result;

}

}

**NodeComparer.cs**

public class NodeComparer : IComparer<Node>

{

public int Compare(Node x, Node y)

{

if (x.Id == y.Id)

{

return 0;

}

int a = x.Result.CompareTo(y.Result);

if (a != 0)

{

return a;

}

int b = x.ExpList.Count.CompareTo(y.ExpList.Count);

if (b != 0)

{

return b;

}

int c = x.I.CompareTo(y.I);

if (c != 0)

{

return c;

}

int sumX = 0;

foreach (int item in x.ExpList)

{

sumX += item;

}

int sumY = 0;

foreach (int item in y.ExpList)

{

sumY += item;

}

return sumY.CompareTo(sumX);

}

}

**Solution.cs**

internal class Solution

{

public double SumWidth { get; set; }

public List<Experiment> Experiments { get; set; }

private double Profits { get; set; } = new double();

private int nodeCounter = 0;

public Solution(List<Experiment> experiments, double sumWidth)

{

Experiments = experiments;

SumWidth = sumWidth;

}

public List<Experiment> GetOptimalList()

{

FillProfits();

Experiments.Sort((a, b) => b.Profit.CompareTo(a.Profit));

SortedSet<Node> nodes = new SortedSet<Node>(new NodeComparer());

var node = GetNode(0, new List<int>());

if (node.Id is not null)

{

nodes.Add(node);

}

Node resultNode = new Node();

while (true)

{

Node maxNode = nodes.Max;

if (maxNode.I >= Experiments.Count)

{

resultNode = maxNode;

break;

}

var firstNode = GetNode(maxNode.I + 1, maxNode.ExpList);

if (firstNode.Id is not null)

{

nodes.Add(firstNode);

}

List<int> list = new List<int>(maxNode.ExpList);

list.Add(maxNode.I);

var secondNode = GetNode(maxNode.I + 1, list);

if (secondNode.Id is not null)

{

nodes.Add(secondNode);

}

nodes.Remove(maxNode);

}

List<Experiment> optimalList = new List<Experiment>();

foreach (var item in resultNode.ExpList)

{

optimalList.Add(Experiments[item]);

}

return optimalList;

}

private void FillProfits()

{

for (int i = 0; i < Experiments.Count; i++)

{

Experiments[i].SetProfit();

}

}

private Node GetNode(int i, List<int> expNums)

{

double filledRank = 0;

double filledWidth = 0;

foreach (var num in expNums)

{

filledRank += Experiments[num].Rank;

filledWidth += Experiments[num].Width;

}

if (filledWidth > SumWidth)

{

return new Node();

}

double result = 0;

if (i >= Experiments.Count)

{

result = filledRank;

}

else

{

result = filledRank + Experiments[i].Profit \* (SumWidth - filledWidth);

}

nodeCounter++;

return new Node(nodeCounter, i, expNums, result);

}

}