Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Факультет «Математика, механика и компьютерные технологии»
Кафедра «Математическое и компьютерное моделирование»

РАБОТА ПРОВЕРЕНА
Рецензент
к.ф.-м.н., доцент,
_____А.Т. Латипова
«____»_____2017 г.

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой «МиКМ»
к.ф.-м.н., доцент,
_____ С.А. Загребина
«____»_____2017 г.

Разработка программного обеспечения для решения задачи выбора инвестиционной программы предприятия

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

ЮУрГУ–01.04.02.2017.052.04.000 МД

Научный руководитель
Профессор, д.ф. – м.н.,
_____ А.В. Панюков
« _____» _____ 2017 г.

Автор работы
студент группы ЕТ-224
_____ Е.А. Загирова
« _____» _____ 2017 г.

Нормоконтролер
Доцент, к.ф. – м.н.,
_____ Макаровских Т.А.
« _____» _____ 2017 г.

Челябинск
2017

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Факультет «Математика, механика и компьютерные технологии»
Кафедра «Математическое и компьютерное моделирование»

УТВЕРЖДАЮ
Заведующий кафедрой «МиКМ»
к.ф.-м.н., доцент,
_____ С.А. Загребина
«____»_____2017 г.

ЗАДАНИЕ

на выпускную квалификационную работу магистра

студенту группы ЕТ-224 Загировой Екатерине Альбертовне

1. **Тема работы**

   Разработка программного обеспечения для решения задачи выбора инвестиционной программы предприятия

   утверждена приказом по университету от _____ 2017г. №_____
2. **Срок сдачи студентом законченной работы** «___» _____ 2017г.
3. **Исходные данные к работе**

   3.1 Модельные данные;

   3.3. Самостоятельно сконструированные тестовые примеры.
4. **Перечень подлежащих разработке вопросов**

   4.1 Аналитический обзор литературы;

   4.2 Исследование видов и методов построения моделей по теме работы;

   4.3 Разработка библиотеки, решающее поставленную математическую задачу;

   4.4 Разработка алгоритмов решения выбранных математических моделей;

   4.5 Реализация интерфейса программного обеспечения.

5. **Перечень графического материала**
   Презентация (15 л.):
   стр. 1 – Титульный лист,
   стр. 2 – Цель и задачи исследования;
   стр. 3 – Описание переменных;
   стр. 4 – Максиминная стратегия;
   стр. 5 – Дисперсия как мера риска при неопределенности;
   стр. 6 – 7 – Вероятность недостижимости заданного уровня ЧДД как мера риска;
   стр. 8 – Алгоритм решения;
   стр. 9 – Диаграмма классов;
   стр. 10 – Схема данных условия задачи;
   стр. 11 – Пример условия задачи;
   стр. 12 – Интерфейс приложения;
   стр. 13 – 15 – Результаты решения;
   стр. 16 – Время выполнения;
   стр. 17 – Заключение и планы;
   стр. 18 – Конец презентации.

6. **Дата выдачи задания**: 2 сентября 2016 г.

Руководитель работы _____/А. В. Панюков/

Задание принял к исполнению _____/Е.А. Загирова/

КАЛЕНДАРНЫЙ ПЛАН

Наименование этапов дипломной работы	Срок выполнения этапов работы	Отметка о выполнении руководителя
1. Подбор литературы по теме дипломной работы	01.02.15 – 14.02.15	
2. Исследование видов и методов построения моделей по теме работы	15.02.15 – 15.03.15	
3. Разработка алгоритмов решения выбранных математических моделей	15.02.15 – 15.03.15	
4. . Программная реализация разработанных алгоритмов	15.03.15 – 30.03.15	
5. Реализация интерфейса программного обеспечения	01.04.15 – 25.05.15	
6. Подготовка пояснительной записки дипломной работы	10.02.15 – 30.05.15	
Написание главы 1	10.02.15 – 25.02.15	
Написание главы 2	25.02.15 – 10.03.15	
Написание главы 3	10.03.15 – 25.04.15	
7. Оформление пояснительной записки	05.06.15 – 10.06.15	
8. Получение отзыва руководителя	06.05.15	
9. Проверка работы руководителем, исправление замечаний	05.05.15 – 06.05.15	
10. Подготовка графического материала и доклада	07.06.15 – 08.06.15	
11. Нормоконтроль	10.06.2015	
12. Рецензирование, представление зав. кафедрой	10.06.2015	

Заведующий кафедрой _____/С.А. Загребина/

Руководитель работы _____/ А.В. Панюков /

Студент _____/ Е.А. Загирова /

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Факультет «Математика, механика и компьютерные технологии»
Кафедра «Математическое и компьютерное моделирование»

Аннотация

Загирова, Е.А. Разработка программного обеспечения для решения задачи выбора инвестиционной программы предприятия / Е.А. Загирова.– Челябинск: ЮУрГУ, Факультет Математики, механики и компьютерных технологий, 2017 , – 70 с., 8 ил., 1 табл., 4 прил., библиогр. список – 29 наименований.

Работа посвящена разработке программной реализации алгоритма для задачи выбора эффективного в долгосрочной перспективе инвестиционного пакета проектов в условиях интервальной неопределенности исходных данных, с критерием формирования пакета в виде чистого дисконтированного дохода всей программы вцелом. Предложены алгоритмы решения и реализация двух моделей на языке программирования C++: с условием гарантированности дохода (максиминная стратегия) и в условиях неопределенности (дисперсия дохода как мера его риска).

Приведена реализация программного приложения для применения библиотеки моделей в виде файла *.html с использованием языка Python.

Оглавление

Введение..... 8

1 Формирование инвестиционной программы предприятия..... 12

1.1 Методы стратегического формирования программы инвестирования промышленного предприятия 12

1.2 Практические подходы оптимизации к корпоративным инвестиционным программам 16

Выводы по главе один 20

2 Методы решения задачи эффективного управления инвестиционной политикой предприятия 21

2.1 Описание переменных и общая постановка задачи 21

2.2 Максиминная стратегия 22

2.3 Оптимальный инвестиционный пакет проектов в условиях риска……23

2.3.1 Дисперсия как мера риска..... 23

2.3.2 Вероятность недостижимости заданного уровня ЧДД как мера
риска……..24

Выводы по главе два..... 27

3 Реализация моделей 29

3.1 Алгоритмы решения 29

3.2 Реализация алгоритмов 30

3.3 Интеграция с Python 38

3.3.1 Язык программирования Python..... 38

3.4 Интеграция с Python 40

3.5 Примеры решения 43
Выводы по главе три 46
Заключение 47
Библиографический список 48
ПРИЛОЖЕНИЯ 52
Приложение А 53
Приложение Б ..... **Ошибка! Закладка не определена.**
Приложение В 60
Приложение Г ..... **Ошибка! Закладка не определена.**
Приложение Д 66
Приложение Е ..... **Ошибка! Закладка не определена.**

Введение

XXI век в России охарактеризовался началом экономических реформ, связанных с усиленным накоплением частного капитала в процессе приватизации, а также сокращением основных и оборотных фондов предприятий. Данные экономические процессы в сочетании с инфляции повлекли за собой экономический кризис, который нанес главный удар по сфере бизнеса и промышленности. Спад производства, а также снижение уровня управления инвестиционными процессами показало малоэффективность действующих бизнес-процессов предприятий, а также инвестиционного комплекса России в целом. Эффективная работа и развитие предприятия были возможны только при увеличением объема инвестиций. Понимая это, участники бизнес-процессов постепенно вывели эффективное управление инвестиционной деятельностью на первое место в стратегическом развитии предприятий [1].

Актуальность. Являясь одним из ключевых инструментов реализации стратегических целей предприятия, эффективное управление инвестиционной деятельностью оказывает большое влияние на социально-экономические, политические и другие факторы. Это влечет за собой повышенную заинтересованность всех участников инвестиционного процесса (государства, кредитных учреждений, частных инвесторов и др.) в улучшении качества разработки инвестиционных программ, повышения их эффективности [2]. Как следствие возникает необходимость в разработке надежного инструмента в виде методов оценки инвестиционных программ на основе экономико-математических моделей. Данные методы позволяют эффективно оценивать инвестиционные процессы и управлять ими.

На данный момент времени существует множество моделей выбора инвестиционной политики. На финансовых рынках наиболее известной является модель Марковица-Тобина управления портфелем ценных бумаг [3, 4], позволяющая с приемлемым риском максимизировать ожидаемую доход-

ность. Основным элементом управления является периодическая диверсификация портфеля.

Для реализации стратегических целей предприятия также необходимо эффективное управление его инвестиционной деятельностью, с целью наиболее эффективной реализации возможных проектов, приносящих с минимальным риском максимальный финансовый результат в условиях ограниченности инвестиционных ресурсов и неопределенности их объемов [5–11]. В данном случае инвестиции являются долгосрочными, а инвестиционная программа по своей сути является стратегическим планом развития предприятия [8–11]. Она рассчитывается на определенное время и включает список различных проектов инвестирования, в котором детально указаны объемы финансовых вложений.

В настоящее время существует ряд рекомендации по оценке инвестиционных проектов предприятий [12], выработанные министерством экономики Российской Федерации. В рекомендациях из других источников [9–11] указывается, что чистый дисконтированный доход (ЧДД) является агрегированным показателем эффективности. Стоит учитывать, что он может являться таковым как для отдельного проекта, так и для всей инвестиционной программы в целом. На основе данного вида дохода вычисляются следующие производные показатели эффективности:

– сроки окупаемости инвестиций,

– разность между суммой доходов и инвестиционными издержками (единовременными затратами) за весь срок использования инвестиционного проекта,

– норма прибыли на капитал,

– приведенные затрат на производство продукции и др.

В бизнес-плане отдельного проекта инвестирования должна быть четко обоснована экономическая целесообразность и описаны все действия по

его реализации [13–15]. При этом вычисляют потоки доходов/расходов и ЧДД для всех возможных расчетных периодов начала реализации.

Объектом данной работы является оптимальная инвестиционная политика предприятия. Разработка и реализация программного обеспечения, автоматизирующего поиск оптимального пакета инвестиционных проектов является **предметом** работы.

Целями работы являются:

– разработка алгоритмов решения задачи поиска эффективного пакета инвестиционных проектов;

– разработка программного обеспечения, позволяющего решать задачи данного направления любому пользователю.

В рамках работы стояли следующие **задачи**:

– подготовить теоретический материала по эффективному управлению инвестиционной деятельностью предприятия;

– разработать структуры программы для интервальных вычислений: спецификация переменных, структура исходных данных, основные классы;

– разработать программы для операционной системы Windows с последовательными алгоритмами с применением объектно-ориентированного подхода;

– разработать алгоритмы решения задачи поиска оптимальной инвестиционной программы.

Отчет состоит из четырех разделов, заключения и библиографического списка (30 назв.).

В разделе 1 рассматриваются некоторые теории и методы рассмотрения инвестиционных программ, определения их эффективности и целесообразности включения проектов в инвестиционный пакет.

В разделе 2 представлены математические модели поиска эффективных пакетов инвестиционных программ, а именно модель реализующая

принцип гарантированного результата и поиск эффективного портфеля в условиях риска [4], две математические модели, где в первом случае риск принимается как дисперсия ЧДД, во втором – как вероятность недостижимости ожидаемого ЧДД.

В разделе 3 рассмотрены основные классы и общая структура программного модуля решения задачи по трем математическим моделям, а также приведены результаты тестирования программы.

В заключении подведены итоги работы.

1 Формирование инвестиционной программы предприятия

1.1 Методы стратегического формирования программы инвестирования промышленного предприятия

Исследования показали, что одной из причин низкого уровня конкурентного статуса многих отечественных промышленных предприятий и их неспособности активно противостоять негативному влиянию экономического кризиса является отсутствие методологии формирования эффективных инвестиционных стратегий. В результате чего имеет место: дефицит финансовых ресурсов; нарастание задолженности перед поставщиками; высокий уровень себестоимости продукции; снижение показателей рыночной капитализации компаний. В условиях экономического кризиса, такие тенденции, прежде всего, касаются предприятий автомобилестроения, металлургии, химической промышленности, строительства. В свою очередь, хозяйствующие субъекты именно этих отраслей являются градообразующими, что предопределяет их особую экономическую и социальную роль.

Для решения этих проблем, в настоящем исследовании поставлено решение следующих задач: определены критерии формирования эффективных инвестиционных стратегий в условиях экономического кризиса; предложена методика стратегического анализа; даны рекомендации по принятию соответствующих управленческих решений на уровнях предприятий и отраслей.

С позиций исследуемой проблемы, под инвестиционной стратегией целесообразно понимать часть общей стратегии предприятия, устанавливающей основные принципы и граничения привлечения и использования денежных, материальных и информационных ресурсов в качестве инвестиций.

На основе экспертного опроса выявлены следующие критерии, которыми следует руководствоваться при принятии решений по инвестиционной стратегии в условиях экономического кризиса [26]:

– соответствие стратегии компании инвестиционным политикам регионов и страны;

– преимущественное использование внутренних источников при формировании инвестиционного фонда предприятия;

– использования родственной диверсификации и интеграции в качестве базовой стратегии роста;

– конкурентоспособность компании, СБЕ и инвестиционных проектов.

Методической базой, позволяющей исследовать потенциальные инвестиционные проекты на предмет соответствия этим критериям, является стратегический анализ. Далее представлена методика стратегического анализа промышленного предприятия с целью разработки эффективной инвестиционной стратегии, которая состоит из следующих этапов (рис. 1.1).

Рисунок 1.1 – Методика стратегического анализа предприятия

Этап 1. Анализ текущей стратегии компании - представляет собой исследование миссии и основных целей организации, параметров диверсифи-

кации, а так же характер и цель последних приобретений, разделений и слияний.

Этап 2. Анализ привлекательности отрасли бизнес-процессов (ПР отр) - должен осуществляться в связке с анализом конкурентоспособности компании и каждого ее подразделения по отдельности. Это позволит на последующих этапах определять перспективность стратегических бизнес-единиц и формировать для них соответствующие стратегии.

Для анализа привлекательности бизнес-процесса для компании могут использоваться следующие показатели: объем рынка и прогнозируемые темпы роста, интенсивность конкуренции, возможности и угрозы в отраслях, доступность ресурсной базы, межотраслевое стратегическое соответствие, структура издержек и прибыли, степень неопределенности и рисков и т.д. Здесь необходимым является анализ взаимосовпадения цепочек ценностей исследуемой корпорации и предприятий, которые рассматриваются в качестве объекта инвестиций. Для этого предлагается анализировать стратегическое соответствие подразделений по функциональным областям управления (производство, финансы, маркетинг, инновации, управление персоналом) на основе экспертного опроса. Здесь для каждого бизнес-процесса который осуществляет компании или имеет стратегические намерения на это, проставляются баллы от 0 до 10.

Этап 3. Оценка конкурентоспособности компании и ее подразделений. Основу для расчетов составляют показатели конкурентоспособности СБЕ по характерным конкурентным полям. Конкурентоспособность подразделения предлагается определять исходя из оценки эффективности его функциональных областей управления по отношению к конкурентам на основе экспертного опроса. Их суммирование с соответствующими весовыми коэффициентами позволяет определить интегральный показатель конкурентоспособности стратегической бизнес-единицы.

Если конкурентный статус исследуемой стратегической бизнес-единицы $КСП_{СБЕ}$ больше чем у всех рассматриваемых конкурентов, можно констатировать, что предприятие является лидером в данной сфере деятельности.

Если $КСП_{СБЕ}$ ниже чем у одного или двух конкурентов - несмотря на то, что не является лидером, занимает прочные позиции на рынке и система управления предприятием функционирует удовлетворительно.

Если $КСП_{СБЕ}$ меньше чем среднее значение по рынку, то уровень конкурентоспособности стратегической бизнес-единицы неудовлетворительный и необходимы срочные мероприятия по повышению эффективности наиболее важных в конкретном случае функциональных областей управления.

Расчет интегрального показателя конкурентоспособности для всего предприятия производится на основе суммирования показателей конкурентоспособностей стратегических бизнес-единиц с весовыми коэффициентами, отражающими значимость эффективности того или иного подразделения в стратегии предприятия.

Этап 4. Построение матрицы «Привлекательность отрасли – конкурентоспособность субъекта инвестиций» представлено на рис. 2. Здесь потенциальные инвестиционные проекты для компании получают определенные статус привлекательности (высокий, средний или низкий).

Этап 5. Разработка новых стратегических инициатив. На этом этапе, результатом стратегического анализа могут стать следующие управленческие решения: изменение стратегических планов некоторых или всех подразделений; включение в бизнес-портфель новых подразделений; отделение малоприбыльных или убыточных подразделений; повышение эффективности за счет создания альянсов; обновление ресурсной базы компании; снижение плановых показателей корпоративной эффективности.

Область применения данной методики охватывает три уровня и позволяет принимать следующие управленческие решения [27]:

1) на уровне корпорации – покупка или продажа СБЕ, создание стратегических альянсов, перераспределение ресурсов;

2) на уровне управления отраслями – формирование или включение компаний в отраслевой кластер, принятие решений о софинансировании проектов;

3) на региональном уровне управления инвестициями - отказ или признание инвестиционной проекта приоритетным для региона, оказание финансовой и нефинансовой поддержки, принятие решений о софинансировании проектов.

1.2 Практические оптимизационные подходы к корпоративным инвестиционным программам

Компании, являющиеся представителями крупного бизнеса, особенно контролируемые государством, обычно формируют свои инвестиционные программы. Основой для этого являются инвестиционные проекты, предложенные отдельными подразделениями корпорации. Естественно, что ее аппарат не в состоянии детально рассмотреть каждый такой проект и потому ориентируется на его общие характеристики, сведенные в некий общий документ («паспорт проекта»), составленный по унифицированной форме.

Принципов как правильно отбирать проекты для включения в программу, никто как правило не знает, поэтому большую роль в этом процессе чаще всего играют лоббистские усилия разработчиков проекта и подразделений – потенциальных исполнителей проекта.

Однако, если взглянуть на задачу формирования инвестиционной программы крупной корпорации с оптимизационных позиций, можно увидеть интересные моменты.

В целях конкретизации определенности будем рассматривать только ту часть инвестиционной программы, которая включает составными «стро-

ками» важнейшие стройки, объекты и мероприятия. Разработка такой программы в любой крупной корпоративной структуре не может являться формальным процессом. Как процедуры среднесрочного и годового планирования, так и их направленность во многом зависят от экономических интересов собственников корпорации и старших менеджеров, чем от общей корпоративной стратегии и от устремлений и грамотности той группы специалистов, которая непосредственно занята плановой работой.

Можно выделить следующие стадии работы [28]:

1) комплексный анализ исходной базы и состояния строительства, который часто объективно не проводится;

2) объективное определение общей потребности в капитальных вложениях, что часто подменяется сбором «заявок» с мест;

3) определение возможностей и источников финансирования капитальных вложений, которое, как правило, проводится в отрыве от действительной потребности и в значительной мере предопределяется рядом конъюнктурных, а не долговременных факторов и обстоятельств при распределении финансовых потоков;

4) установление лимита капитальных вложений на соответствующий период, что часто является результатом внутренних сделок между руководителями администрации корпорации и не вытекает ни из объективного анализа общей ситуации, ни из оптимизационных экономических расчетов;

5) распределение установленного лимита между важнейшими стройками и направлениями вложений, что зачастую делается на интуитивной, а не оптимизационной основе [29]. Чаще всего, при этом игнорируются предусмотренные проектами графики распределения инвестиций по периодам строительства, и упускаются из виду потребности в затратах на возмещение выбывающих основных фондов и на поддержание мощностей действующих предприятий.

В то же время инвестиционные программы крупной корпорации почти полностью контролируются ее высшими менеджерами. Это обстоятельство создает определенные возможности для оптимизации программ капитальных вложений уже на стадии их разработки. Важно иметь в виду, что формирование рациональной инвестиционной программы не сводится к механическому объединению предварительно отобранных проектов. Отдельный инвестиционный проект сам по себе может быть достаточно эффективным, однако не исключены ситуации, когда он по каким-либо причинам не «встраивается» в сводный инвестиционный портфель корпорации.

Наиболее же существенным является тот факт, что инвестиционную программу, в отличие от отдельного проекта, нельзя считать «малым» мероприятием. Строго говоря, в этом случае показатель ЧДД (и даже критерий DEI, сравнительно новый критериальный показатель эффективности – Дисконтированный эквивалентный доход, и входящая в его расчет ставка дисконта. Показатель DEI более точно по сравнению с ЧДД оценивает эффективность проекта. Метод дисконтированных денежных потоков при этом превращается в метод дисконтированных эквивалентных доходов.) для оценки эффективности программы не применим – ей необходимо давать «глобальную» оценку, исходя из влияния программы на всю деятельность корпорации. Это справедливо и по отношению к крупным инвестиционным проектам, включаемым в программу. Другими словами, частная, казалось бы, задача формирования оптимальной инвестиционной программы становится неотъемлемой составной частью стратегического плана развития корпорации и не может быть решена «локальными» оптимизационными методами. С этой точки зрения изложенные в литературе многочисленные модели оптимизации планов капитального строительства следует признать теоретически необоснованными: такие планы нельзя оптимизировать в отрыве от стратегического плана развития фирмы.

На практике сформировать оптимальную инвестиционную программу можно двумя способами [28].

1. Путем перебора. Здесь формируются различные варианты программы, и для каждого из них определяются денежные потоки (относящиеся не только к вошедшим в программу объектам, но и ко всей производственной деятельности корпорации). Затем решается оптимизационная задача (с дополнительными общекорпоративными ограничениями). Лучший вариант определяется по значению критерия оптимальности этой модели.

2. Итеративно. Выбирается один из вариантов программы. Решается двойственная модель с денежными потоками, отвечающими этому варианту, после чего оценивается DEI для всех предложенных к включению в программу проектов (за исключением включаемых по неэкономическим соображениям). Если выясняется, что в выбранном варианте программы присутствует проект с отрицательным DEI, он оттуда исключается, если же туда не был включен проект с положительным DEI, он вводится в программу. После каждого такого изменения программы (или после нескольких таких изменений) возникает ее новый вариант со своими ставками дисконта и оценками прав заимствования и других ограничений, для которого все предыдущие операции повторяются.

Рациональная инвестиционная программа должна формироваться итеративно, с использованием сведений о многих вариантах предлагаемых проектов. Поэтому говорить о том, что предложения о включении проекта в программу должны представляться в виде некоего унифицированного паспорта или, тем более, отдельной строки унифицированной таблицы, в современных условиях просто нецелесообразно. Наиболее приемлемо предоставление такой информации на цифровом носителе с описанием достаточно большого числа вариантов расчета, из которых в процессе итеративной разработки программы можно было бы выбирать наиболее подходящие.

Рассмотренные способы оценки эффективности, для корпоративных инвестиционных программам по «крупным» проектам, в процессе его реализации чаще всего предполагают существенное изменение целей и интересов компании. Поэтому задачей проектировщика-экономиста корпоративных инвестиционных программам, становится выявление необходимости пересмотра стратегии развития компании.

Выводы по главе один

Рассматриваемая в первой части главы модель описывает конкретную бизнес-единицу как возможный объект финансирования и оценивает целесообразность инвестирования средств в нее, но не затрагивает возможность инвестирования в другие проекты и бизнес-единицы, а также наличие у предприятия ресурсов на них.

Во второй части главы описывается очень сложный комплексный подход к поиску эффективных инвестиционных программ для корпораций, что значительно повышает сложность моделей и возможность их математического формулирования на данном этапе.

2 Математические методы решения задачи эффективного управления инвестиционной политикой предприятия

2.1 Описание переменных и общая постановка задачи

Инвестиционная программа представляет собой совокупность реализуемых инвестиционных проектов предприятия, состоящих из перечня объектов инвестиций, их характеристик и объемов финансирования [5].

В данной научно-исследовательской работе, в качестве основного критерия для формирования оптимальной инвестиционной программы предприятия, выбираем чистый дисконтированный доход (ЧДД) инвестиционной программы в целом. Под ЧДД будем подразумевать сумму чистых дисконтированных доходов проектов, которые были включены в инвестиционную программу.

Были введены следующие обозначения [21, 22]:

- $P = \{p_1, p_2, ..., p_n\}$ – множество из инвестиционных проектов, которые могут быть включены в состав инвестиционной программы;

$L = \{l_1, l_2, ..., l_n\}$ – множество продолжительностей реализации инвестиционных проектов (в расчетных периодах);

m – горизонт планирования (число расчетных периодов);

$R = \{r_1, r_2, ..., r_{m-1}\}$ – объемы финансирования инвестиционной программы предприятия по расчетным периодам.

Каждый из инвестиционных проектов p_j, $j = 1, 2, ..., n$ охарактеризован двумя показателями:

- NPV_{js} – величина ЧДД, приведенного к моменту начала реализации проекта p_j, если он был начат в период s.

- I_{jsi} – потребность в финансировании инвестиционного проекта j в расчетный период i от начала реализации инвестиционной программы, при условии, что он будет начат в период s.

Показатели доходов и расходов являются прогнозируемыми величинами и зависят от ряда факторов. Поэтому целесообразно считать NPV и I случайными величинами, а для каждого проекта и всех расчетных периодов получены интервальные оценки ЧДД $[\underline{NPV}_{js}, \overline{NPV}_{js}]$, потребностей $[\underline{I}_{jsi}, \overline{I}_{jsi}]$ и финансовых ресурсов предприятия $[\underline{r}_i, \overline{r}_i]$.

Также введена двоичная переменная x:

$$x_{js} = \begin{cases} 0, \text{ если проект } p_j \text{ не начинается в периоде } s; \\ 1, \text{ если проект } p_j \text{ начинается в периоде } s. \end{cases}$$

Поскольку реализация инвестиционного проекта p_j может начаться не позже чем в период $m - l_j$, то должно выполняться следующее условие:

$$\sum_{s=0}^{m-l_j} x_{js} \le 1, \ \ j = 1,2,...,n.$$

Условие реализуемости инвестиционной программы может быть записано в виде

$$\sum_{j=1}^{n} \sum_{s=0}^{i} I_{jsi} x_{js} \le r_i, \ \ j = 0,1,2,...,m-1.$$

ЧДД для всего портфеля проектов равен

$$NPV = \sum_{j=1}^{n} NPV_j = \sum_{j=1}^{n} \sum_{s=0}^{m-l_j} NPV_{js} x_{js}.$$

2.2 Максиминная стратегия

Максиминная (осторожная) стратегия направлена на получение максимального гарантированного результата. Применяя нижние оценки ЧДД и объемов финансирования по расчетным периодам и верхние оценки потреб-

ностей финансирования, гарантируется оптимальность значения следующей задачи:

$$\underline{NPV}(x) = \sum_{j=1}^{n} \sum_{s=0}^{m-l_j} \underline{NPV}_{js} x_{js} \to \max_{x \in D},$$

$$\sum_{j=1}^{n} \sum_{s=0}^{i} \bar{I}_{jsi} x_{js} \le \underline{r}_i, \ \ i = 0,1,2,...,m-1; \qquad (1)$$

$$\sum_{s=0}^{m-l_j} x_{js} \le 1, \ \ j = 1,2,...,n; \qquad (2)$$

$$x_{js} \in \{0,1\}, \ \ j = 1,2,...,n, \ \ s = 0,1,...,m-l_j. \qquad (3)$$

2.3 Оптимальный инвестиционный пакет проектов в условиях риска

2.3.1 Дисперсия как мера риска

В данной работе также рассматривается реализация модели с учетом риска. Дисперсия чистого дисконтированного дохода $\mathbf{D}\{NPV(x)\}$ используется в качестве меры риска по аналогии с задачей Марковица-Тобина [2] при формирования портфеля ценных бумаг. При этом учитывается независимость между ЧДД и булев характер переменной x.

В данном случае строятся эффективные инвестиционные программы в условиях риска (т.е. множество Парето в пространстве критериев «риск»-«ЧДД»). Наиболее подходящая инвестиционная программа выбирается на основании системы предпочтений лица принимающего решение при использовании систем поддержки принятия решений.

Ожидаемая величина ЧДД инвестиционной программы ищется как математическое ожидание $\mathbf{E}\{NPV(x)\}$, полагая, что чистый дисконтированный доход равномерно распределен на интервале $\left[\underline{NPV}_{js}, \overline{NPV}_{js}\right]$, $s = 1,2,...,l_j$.

В итоге получаем, что эффективную инвестиционную программу в условиях риска можно найти при решении задач [21, 22]:

$$\mathbf{E}\{NPV(x)\} = \sum_{j=1}^{n}\sum_{s=0}^{m-l_j}\left(\frac{\overline{NPV}_{js} + \underline{NPV}_{js}}{2}x_{js}\right) \to \max_{x\in D:\mathbf{D}\{NPV(x)\}\le d}, \quad (5)$$

$$\mathbf{D}\{NPV(x)\} = \sum_{j=1}^{n}\sum_{s=0}^{m-l_j}\left(\frac{\left(\overline{NPV}_{js} - \underline{NPV}_{js}\right)^2}{12}x_{js}\right) \to \min_{x\in D:\mathbf{E}\{NPV(x)\}\ge c}, \quad (6)$$

для которых выполняются условия (1) - (3), d и e – допустимые уровни дисперсии и математического ожидания соответственно.

Все представленные задачи являются задачами булева линейного программирования, поэтому решены алгоритмом на основе динамического программирования.

2.3.2 Вероятность недостижимости заданного уровня ЧДД как мера риска

В данном случае обеспечивается минимальный риск возможности выполнить программу для некоторого значения ожидаемого ЧДД. В качестве меры риска, используется вероятность недостижимости желаемого среднего значения ЧДД [18, 19]. Обозначим заданный уровень ЧДД как NPV, допустимый риск недостижимости NPV как α, допустимый риск превышения инвестиционной программой предприятия, требуемых ресурсов в расчетном периоде i как β_i.

Для определения расчетной формулы вероятности недостижимости заданного уровня необходимо было использовать детерминированный эквивалент задачи с вероятностными критериями [20]. Поэтому были рассмотрены события E_j, состоящие в том, что доход от проекта p_j будет не меньше величины y_j (где y_j - фиксированы). В результате математических преобразований была получена вероятность недостижимости пакетом программ заданного значения NPV, т.е. меры риска $\mathbf{P}\{NPV(x) < NPV\}$.

Также учитывается вероятность еще двух событий:

– финансовые ресурсы I_{jsi}, требуемые проекту, не превосходят z_{ji} (где z_{ji} - фиксированы), т.е. $P\{I_{jsi} \le z_{ji}\}$;

– ресурсы, требуемые *всем* выполняемым проектам, не превосходят объемов финансирования r_i, т.е. $P\left\{\sum_{j=1}^{n} z_{ji} \le r_i\right\}$.

Вследствие математических преобразований задача определения максимального ожидаемого дохода приняла следующий вид [21]:

$$NPV(x,y,z) = \sum_{j=1}^{n} y_j \to \max_{x,y,z}, \tag{7}$$

$$\sum_{s=0}^{m-l_j} NPV_{js} x_{js} = y_j, \ \ j = 1,2,...,n; \tag{8}$$

$$\sum_{j=1}^{n} \sum_{s=0}^{m-l_j} \left(x_{js} \frac{y_j - \underline{NPV}_{js}}{\overline{NPV}_{js} - \underline{NPV}_{js}} \right) \le \alpha; \tag{9}$$

$$\frac{\sum_{j=1}^{n} z_{ji} - \underline{r}_i}{\overline{r}_i - \underline{r}_i} + \sum_{j=1}^{n} \sum_{s=0}^{m-l_j} \left(x_{js} \frac{\overline{I}_{jsi} - z_{ji}}{\overline{I}_{jsi} - \underline{I}_{jsi}} \right) \le \beta_i, \ \ i = 1,2,...,m; \tag{10}$$

$$x_{js} z_{ji} \le \overline{I}_{jsi}, \ \ j = 1,2,...,n, \ \ s = 0,1,...,m-l_j, \ \ i = 1,2,...,m; \tag{11}$$

$$\sum_{s=0}^{m-l_j} x_{js} \le 1, \ \ j = 1,2,...,n; \tag{12}$$

$$x_{js} \in \{0,1\}, \ \ j = 1,2,...,n, \ \ s = 0,1,...,m-l_j. \tag{13}$$

Увеличение уровня вероятности недостижимости возможного ожидаемого значения ЧДД в стохастических моделях также ведет к инвестиционным программам с возрастающими средними ожидаемыми значениями ЧДД. Последняя стохастическая модель допускает риск превышения инвестиционной программой предприятия требуемых ресурсов по расчетным периодам, то возможные средние значения ожидаемых ЧДД в стохастической модели оказываются выше.

Следует обратить внимание, что данная задача, как и две предыдущие, является задачей булева линейного программирования. Соответственно, при переборе всех наборов проектов пакета, набор

$$X = \left\{ x_{js} : x_{js} \in \{0,1\},\ j = 1,2,...,n,\ s = 0,1,...,m - l_j \right\}$$

будет фиксироваться. С учетом данной особенности, задачу (7) – (13) можно разбить на две подзадачи. Во-первых, это система неравенств, составляющая ограничение на ресурсы проектов (10) – (13). Во-вторых, задача максимизации чистого дисконтированного дохода (7) – (9), (12), (13).

Рассмотрим первую подзадачу. Для применения к ней стандартных методов решения систем линейных неравенств сначала необходимо привести ее к классическому виду $Ax \le b$. Сначала отделим слагаемые с переменными z_{ji} от слагаемых с заданными значениями:

$$\frac{\sum_{j=1}^{n} z_{ji}}{\overline{r}_i - \underline{r}_i} - \frac{\underline{r}_i}{\overline{r}_i - \underline{r}_i} + \sum_{j=1}^{n} \sum_{s=0}^{m-l_j} \left(x_{js} \frac{\overline{I}_{jsi}}{\overline{I}_{jsi} - \underline{I}_{jsi}} \right) - \sum_{j=1}^{n} \sum_{s=0}^{m-l_j} \left(x_{js} \frac{z_{ji}}{\overline{I}_{jsi} - \underline{I}_{jsi}} \right) \le \beta_i,\ \ i = 1,2,...,m.$$

Для данной системы верны следующие утверждения:

$$\frac{\sum_{j=1}^{n} z_{ji}}{\overline{r}_i - \underline{r}_i} = \sum_{j=1}^{n} \left(\frac{z_{ji}}{\overline{r}_i - \underline{r}_i} \right),$$

$$\sum_{j=1}^{n} \sum_{s=0}^{m-l_j} \left(x_{js} \frac{z_{ji}}{\overline{I}_{jsi} - \underline{I}_{jsi}} \right) = \sum_{j=1}^{n} \left(z_{ji} \sum_{s=0}^{m-l_j} \left(x_{js} \frac{1}{\overline{I}_{jsi} - \underline{I}_{jsi}} \right) \right).$$

Применяя их к данной системе, вносим под единый знак суммы переменные z_{ji} с коэффициентами и выносим их за скобку:

$$\sum_{j=1}^{n} \left[z_{ji} \left(\frac{1}{\overline{r}_i - \underline{r}_i} - \sum_{s=0}^{m-l_j} \left(x_{js} \frac{1}{\overline{I}_{jsi} - \underline{I}_{jsi}} \right) \right) \right] - \frac{\underline{r}_i}{\overline{r}_i - \underline{r}_i} + \sum_{j=1}^{n} \sum_{s=0}^{m-l_j} \left(x_{js} \frac{\overline{I}_{jsi}}{\overline{I}_{jsi} - \underline{I}_{jsi}} \right) \le \beta_i$$

Слагаемые без переменной переносятся в правую часть системы неравенств, и подзадача принимает вид:

$$\sum_{j=1}^{n} \left[z_{ji} \left(\frac{1}{\overline{r}_i - \underline{r}_i} - \sum_{s=0}^{m-l_j} \left(x_{js} \frac{1}{\overline{I}_{jsi} - \underline{I}_{jsi}} \right) \right) \right] \le \frac{\underline{r}_i}{\overline{r}_i - \underline{r}_i} + \beta_i - \sum_{j=1}^{n} \sum_{s=0}^{m-l_j} \left(x_{js} \frac{\overline{I}_{jsi}}{\overline{I}_{jsi} - \underline{I}_{jsi}} \right), \quad (14)$$

при выполненных условиях (11) – (13).

Теперь рассмотрим вторую подзадачу. Аналогично приведем ее к виду $Ax \le b$. Сначала отделим слагаемые с переменными y_j от слагаемых с заданными значениями в неравенстве (9):

$$\sum_{j=1}^{n}\sum_{s=0}^{m-l_j}\left(x_{js}\frac{y_j}{\overline{NPV}_{js}-\underline{NPV}_{js}}\right)-\sum_{j=1}^{n}\sum_{s=0}^{m-l_j}\left(x_{js}\frac{\underline{NPV}_{js}}{\overline{NPV}_{js}-\underline{NPV}_{js}}\right)\le\alpha\,.$$

Вынесем переменные за знак суммы:

$$\sum_{j=1}^{n}\left(y_j\sum_{s=0}^{m-l_j}\left(x_{js}\frac{1}{\overline{NPV}_{js}-\underline{NPV}_{js}}\right)\right)-\sum_{j=1}^{n}\sum_{s=0}^{m-l_j}\left(x_{js}\frac{\underline{NPV}_{js}}{\overline{NPV}_{js}-\underline{NPV}_{js}}\right)\le\alpha\,.$$

Слагаемое без переменной переносится в правую часть, а целевая функция (7) переписывается с учетом (8). Подзадача принимает вид:

$$\sum_{j=1}^{n}\sum_{s=0}^{m-l_j}NPV_{js}x_{js}\to\max_{x,y,z},\tag{15}$$

$$\sum_{j=1}^{n}\left(y_j\sum_{s=0}^{m-l_j}\left(x_{js}\frac{1}{\overline{NPV}_{js}-\underline{NPV}_{js}}\right)\right)\le\alpha+\sum_{j=1}^{n}\sum_{s=0}^{m-l_j}\left(x_{js}\frac{\underline{NPV}_{js}}{\overline{NPV}_{js}-\underline{NPV}_{js}}\right),\tag{16}$$

При условиях (11) – (13).

Теперь задача (7) – (13) может быть решена посредством последовательного перебора и фиксирования пакета проектов X и следующего за этим разбиением на задачи линейного программирования (14) и (15) – (16), которые можно решить симплекс методом.

Выводы по главе два

В главе были рассмотрены модели оптимальной инвестиционной программы с фиксированными объемами финансирования. Данные модели позволяют формировать оптимальные по Парето инвестиционные программы, что возможно сделать при известном распределении финансовых средств по периодам. Для учета неопределенности финансовых ресурсов при поддержки инвестиционных проектов были представлены модификации данной модели и приведение их к стандартному виду.

Системную оценку инвестиционной привлекательности моделируемого предприятия дают решения соответствующих задач, которые в свою очередь могут быть использованы в интеллектуальных системах поддержки выбора эффективного портфеля с учетом производных показателей эффективности. В число данных показателей входят: (1) сроки окупаемости инвестиций, (2) норма прибыли на капитал, (3) разность между суммой доходов и инвестиционными издержками (единовременными затратами) за весь срок использования инвестиционного проекта, (4) приведенные затрат на производство продукции и др. и склонности к риску лица принимающего решение.

3 Реализация моделей

3.1 Алгоритмы решения

Решение задачи поиска оптимального пакета проектов проводится методом динамического программирования через полный перебор всех возможных комбинаций активных проектов (состояний системы) в каждый момент времени. Комбинация проектов с максимальным ЧДД и является оптимальной. Алгоритм решения задачи состоит из следующих основных шагов.

Шаг 0. Инициализация. Составить список всех состояний на текущий момент $i = 0$, $s = 0$. Это состояния, в которых проекты либо не начались, либо активировались в момент $s = 0$. Исключить из списка все состояния, которые не удовлетворяют условиям, т.е. комбинация требует инвестиций больше, чем предприятие может выделить (2) или какой-либо из проектов комбинации будет активен и за пределами горизонта событий. В задаче (5) также проверяется математическое ожидание ($\mathbf{D}\{NPV(x)\} \le d$).

Шаг 1. Динамика системы во времени. Начать цикл по времени. На каждой итерации переменная времени увеличивается на 1. Если данная переменная сравняется по значению с горизонтом событий, то цикл прекращается и происходит переход к шагу 3, иначе – к шагу 2.

Шаг 2. Тело цикла. Составить новый список состояний. Для этого просмотреть все состояния системы из списка состояний. Построить все возможные состояния системы исходя из предыдущего состояния, т.е. добавить в комбинацию уже активных проектов другие возможные комбинации проектов. При этом проведя такую же проверку каждого состояния, как и в шаге 0. Если не удалось активировать еще проекты, оставить текущее состояние системы. Удалить старый список состояний.

Шаг 3. Поиск оптимума. В зависимости от задачи, найти максимальное по ЧДД (1), по матожиданию (5) или минимальное по дисперсии (6) состояние системы. Это и будет решением задачи.

Замечание. Проверка условия $\mathbf{E}\{NPV(x)\} \geq e$ в задаче (6) проходит в самом конце алгоритма при переборе списка всех найденных состояний в поиске оптимального, т.к. при более раннем применении оно может отсечь возможные варианты, где основной доход от проектов приходит в более поздние периоды.

3.2 Реализация алгоритмов

Вся задача и ее решение представляется как динамическая система, изменяющаяся с течением времени. Описание всей системы представлено на рисунке 3.1 и в приложениях А – Б. Для быстрой обработки данных решение реализовано на языке программирования C++ в программе CLion.

На классе `InvestGraph`, строится все решение задачи. Переменная `data` хранит все входные данные задачи, в `answer` записывается состояние системы с максимальным ЧДД, `statusList` хранит в себе список всех состояний системы на текущей итерации цикла, в `statusListNextStep` записываются все новые состояния, полученные на данной итерации.

Листинг 3.1 – Описание класса `Status`

```
class Status
{
	vector <int> Prj;//активные, не активные, завершенные проекты
	double NPV;//чистый дисконтированный доход пакета проектов
	vector <double> inv;//ресурсы, затрачиваемые проектами в момент вре-
мени i,inv[i]
	double E;//мат. ожидание
	double D;//дисперсия
public:
	Status(void){};
	Status(const vector <int>& _Prj, const Data& data);
	~Status(void){};

	vector <int> get_Prj(){ return Prj;}
	vector <double> get_inv(){ return inv;}
	double get_NPV(){ return NPV;}
	double get_E(){ return E;}
	double get_D(){ return D;}

	vector <int> get_Prj() const { return Prj;}
	vector <double> get_inv() const { return inv;}
	double get_NPV() const { return NPV;}
```

```
        double get_E() const { return E;}
        double get_D() const { return D;}

        void writeCSV( string filename = "output_status.csv");

        bool const operator <(Status const &right) const;
        bool const operator >(Status const &right) const;
        bool operator ==(Status const &right) const;

    void writeStatus();

        }
};
```

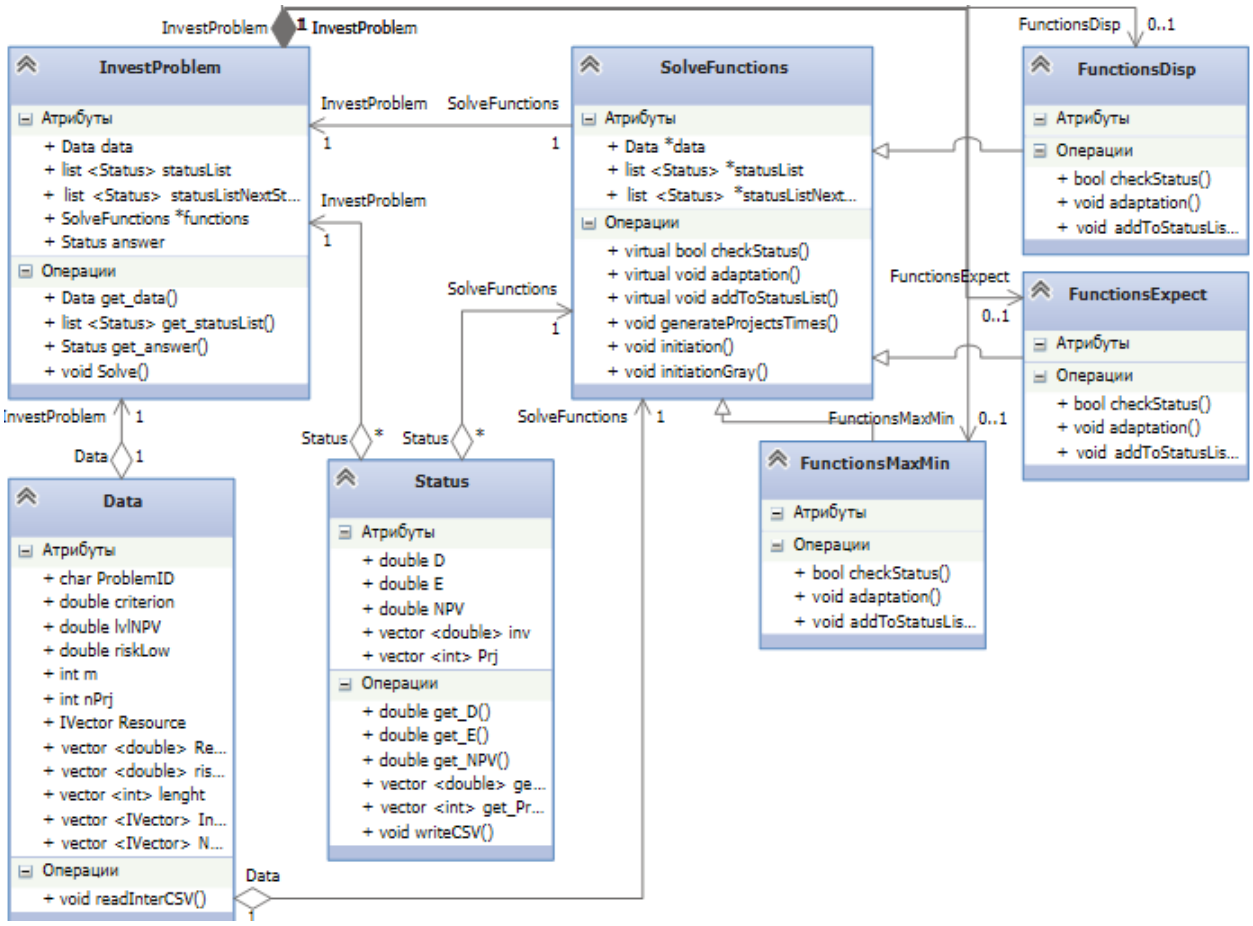

Рисунок 3.1 – Диаграмма классов

Для описания состояния системы используется класс `Status` (Листинг 3.1). В векторе `Prj` хранятся периоды начала реализации s проектов j или число -1, как идентификатор того, что проект не начался (не был активен за все время планирования). В переменной `NPV` хранится суммарный ЧДД по всем активным проектам данного набора. В векторе `inv` хранятся требуемые

суммарные инвестиции в проекты в каждый период времени $[0;m]$. В классе `Status` также хранится матожидание и дисперсия состояния, которые просчитываются при создании объекта класса.

В `Data` (Листинг 3.2) хранится идентификатор задачи и уровень показателя, т.е. для задачи (5) – верхняя граница дисперсии (максимально возможный риск), для задачи (6) – нижняя граница матожидания (минимальный ожидаемый ЧДД). Все переменный подробно описаны в комментариях листинга 3.2.

Листинг 3.2 – Класс `Data` для хранения данных задачи

```
class Data
{
public:
      char ProblemID;//идентификатор задачи: m - maxmin, e - expectation, d
- dispersion
   double criterion;//ограничения для дисперсии, матожидания
      int nPrj;//количество проектов
      int m; // горизонт планирования
      double riskLow, lvlNPV;//риск недостижимости уровня lvlNPV

      vector <double> ResSupBound;// риск превышения ресурсов, размера nPrj
      vector <double> riskUp; // риск превышения ресурсов, размера m
      vector <int> lenght; //длительность проектов, вектор размера nPrj
      IVector Resource; //финансовые ресурсы компании, вектор размера m

      vector <IVector> NPV; //net present value чистый дисконтированный до-
ход
      vector <IVector> Investments; //ежегодные инвестиции(вложения) в про-
ект

      Data(void){
    ProblemID = 'm';
  }
      Data( Data &a);

      Data(string filename){ readInterCSV(filename);}
      ~Data(void){};

      void readInterCSV (string filename);
};
```

Описание класса `InvestProblem`, на базе которого и строится все решение, представлено на рисунке 3.1 и листинге 3.3. Переменная `data` хранит все входные данные задачи, в `answer` записывается состояние системы с

максимальным ЧДД, `statusList` хранит в себе список всех состояний системы на текущей итерации цикла, в `statusListNextStep` записываются все новые состояния, полученные на данной итерации. Решение задачи запускается методом `Solve()`, все остальные методы, применяемые для решения задачи, сгруппированы в абстрактном классе `SolveFunctions` (листинг 3.4), где виртуальными методами объявлены `addToStatusList()` и `checkStatus()`. От него наследуют классы `FunctionsMaxMin`, `FunctionsExpect` и `FunctionsDisp`, в которых данные виртуальные методы переопределяются в соответствии с решаемой задачей, т.е. в условиях гарантированности результата, максимального матожидания или минимальной дисперсии соответственно.

Листинг 3.3 – Описание класса `InvestProblem`

```
class InvestProblem
{
      Data data;
      Status answer;
      list <Status> statusList;
      list <Status> statusListNextStep;

      SolveFunctions *functions;

public:
      InvestProblem(void){
    functions = new FunctionsMaxMin(data, statusList, statusListNextStep);
  }
      InvestProblem(string filename):data(filename){
            switch (data.ProblemID)
            {
            case 'm':{functions = new FunctionsMaxMin(data, statusList,
statusListNextStep); break;}
            case 'e':{functions = new FunctionsExpect(data, statusList,
statusListNextStep); break;}
            case 'd':{functions = new FunctionsDisp(data, statusList, sta-
tusListNextStep); break;}
            default:
                  break;
            }
      };
      ~InvestProblem(void){
            delete functions;
      };

      Data get_data() {return data;}
```

```
        Status get_answer() {return answer;}
        list <Status> get_statusList() {return statusList;}

        void Solve();
};
```

Задача решается в две строчки, в третьей результат выводится в файл:

```
InvestGraph system("input.csv");
system.Solve();
system.get_answer().writeCSV("output.csv");
```

Т.е. первоначально создается объект класса, где на вход подается файл формата *.csv, из которого читаются данные к задаче (по умолчанию файл «input.csv») и записываются в `data`. Затем вызывается главный метод `Solve()`, который и запускает алгоритм решения. Резельтат решения записывается в файл output.csv.

Листинг 3.4 – Описание класса `SolveFunctions`

```
class SolveFunctions
{
protected:
        Data *data;
        list <Status> *statusList;
        list <Status> *statusListNextStep;

public:
        void initiationGray(const int time, const int k, const vector <int>&
baseVec);
  void initiation(); //инициализация списка
  void nextStep( const int time); // следующий шаг в решении - новый список
состояний
        void generateProjectsTimes(int time, const vector <int>& baseVec,
vector<vector<int>>& List);

        virtual void addToStatusList(const Status &status) = 0;
        virtual void adaptation (const vector <int>& baseVec, const vector
<int>& g) = 0;
        virtual bool checkStatus (Status& status) = 0;

  SolveFunctions(){};
        SolveFunctions(Data &d,
                list <Status> &s,
                list <Status> &n): data(&d), statusList(&s), statusList-
NextStep(&n){};
        ~SolveFunctions(void){};

        Data& get_data(){ return *data;}
        list <Status>& stList(){ return *statusList;}
```

```
      list <Status>& stListNext(){ return *statusListNextStep;}
};
```

В первую очередь, метод `Solve()` (листинг 3.5) вызывает метод `initiation()` (листинг 3.6), который реализует шаг 0 алгоритма, и осуществляет временной цикл шага 1, где телом цикла является метод `NextStep()` (листинг 3.7). По завершении цикла получаем конечный список состояний, в котором находится максимальное по ЧДД и в переменную `Answer` записывается первый элемент полученного списка – состояние системы с оптимальным набором активных проектов.

Листинг 3.5 – Описание метода `Solve`

```
void InvestProblem::Solve(){
      ofstream write("time.csv", ios::app);

      unsigned int start_time = clock();//засекаем время

      functions->initiation();

      unsigned int end_time = clock();
      write << data.nPrj << " projects"<< ';' << (end_time - start_time) <<
';';//выводим время

  for (int i = 0; i < data.m; i++ ){
            cout << "NEXT STEP - " << i<< '\n';
    functions->nextStep(i+1);

            end_time = clock();
            write << (end_time - start_time) << ';';//выводим время
  }

      if (data.ProblemID == 'd'){
            for (list <Status>::iterator i = statusList.begin(); i != sta-
tusList.end();){
                  if ( (*i).get_E() < data.criterion){
                        list <Status>::iterator j = i;
                        i++;
                        statusList.erase(j);
                  }
                  else
                        i++;
            }
      }

      if (!statusList.empty()){
            answer = (*statusList.begin());
      }
```

```
      write << '\n';
      write.close();
       }
```

Листинг 3.6 – Описание метода `Initiation`

```
void SolveFunctions::initiation(){
      vector <int> prj(get_data().nPrj, -1);
      Status status(prj, get_data());
      stListNext().push_back(status);

      vector <vector <int>> List;
      generateProjectsTimes(0, prj, List);
      for (int j = 1; j < List.size(); j++){
            //преобразование нового варианта
            adaptation(prj, List[j]);
      }

  stList() = stListNext();
  stListNext().clear();
       }
```

Листинг 3.7– Описание метода `Initiation`

```
void SolveFunctions::nextStep( const int time){
      //Для каждого элемента списка состояний
  for (list <Status>::iterator i = stList().begin(); i != stList().end();
i++){
    vector <int> baseVec = (*i).get_Prj();
            stListNext().push_back(*i);

            vector<vector<int>> List;
            generateProjectsTimes(time, baseVec, List);
            for (int j = 1; j < List.size(); j++){
                  //преобразование нового варианта
                  adaptation(baseVec, List[j]);
            }
  }
  stList() = stListNext();
  stListNext().clear();
       }
```

В методах `initiation()` и `NextStep()` описывается подготовка данных к перебору всех вариантов состояний системы. Этот перебор является бинарным, т.к. у проектов только два возможных состояния: начаться в текущий период времени или не начаться.

За основу данного процесса, первоначально, был взят рекурсивный метод Грея [25, 26] с некоторой адаптацией под задачу, где на вход подаются идентификатор `k=0`, период времени, ссылка на перебираемый вектор и вектор проектов, относительно которого будут перебираться варианты. В мето-

дах `gray()` и `rGray()` (листинг 3.8) при получении новой комбинации вызывается метод `adaptation()`, который проводит окончательную обработку вектора проектов, формирует новое состояние, проверяет его на потребность в инвестициях (метод `checkStatus`) и добавляет в список новых состояний при удовлетворении состояния всем условиям.

Листинг 3.8– Описание методов `Gray` и `rGray`

```
void InvestGraph::gray (int time, int k, vector <int>& g, const vector
<int>& baseVec){
      if (k == g.size() ){//получили новый вектор по Грею
    adaptation(baseVec, g);
  }
  else{
    g[k] = -1;
    gray(time, k + 1, g, baseVec);
    g[k] = time;
    rGray(time, k + 1, g, baseVec);
  }
}

void InvestGraph::rGray(int time, int k, vector <int>& g, const vector
<int>& baseVec){
  if (k == g.size() ){//получили новый вектор по Грею
    adaptation(baseVec, g);
  }
  else{
    g[k] = time;
    gray(time, k + 1, g, baseVec);
    g[k] = -1;
    rGray(time, k + 1, g, baseVec);
}}
```

В окончательном варианте разработки генерацию новых вариантов состояний выполняет метод `generateProjectsTimes()`, который на вход получает кроме времени и базового вектора [6] список векторов, куда записываются новые варианты возможных изменений состояний (листинг 3.9). Благодаря такой реализации стала возможной параллельная обработка состояний, что должно значительно повысить скорость решения задач.

Листинг 3.7– Описание метода `generateProjectsTimes`

```
void SolveFunctions::generateProjectsTimes(
       int time,
       const vector <int>& baseVec,
```

```
       vector<vector<int>>& List){
      //количество проектов, которые никогда не активировались
  int k = 0; for (int i = 0; i < baseVec.size(); i++) if (baseVec[i] == -1)
k++;

      vector <int> empty(k, -1);
      List.push_back(empty);

      for (int i = 0; i < k; i++){
            int n = List.size();
            for (int j = 0; j < n; j++){
                  List.push_back(List[j]);
                  List[n+j][i] = time;
            }
      }
       }
```

3.3 Интеграция с Python

3.3.1 Язык программирования Python

Python [23] – высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объём полезных функций.

Python поддерживает несколько парадигм программирования, в том числе структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное. Основные архитектурные черты – динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений и удобные высокоуровневые структуры данных. Код в Python организовывается в функции и классы, которые могут объединяться в модули (они в свою очередь могут быть объединены в пакеты).

Эталонной реализацией Python является интерпретатор CPython, поддерживающий большинство активно используемых платформ [24]. Он распространяется под свободной лицензией Python Software Foundation License, позволяющей использовать его без ограничений в любых приложениях, включая проприетарные. Есть реализации интерпретаторов для JVM (с возможностью компиляции), MSIL (с возможностью компиляции), LLVM и дру-

гих. Проект PyPy предлагает реализацию Python на самом Python, что уменьшает затраты на изменения языка и постановку экспериментов над новыми возможностями.

Python – активно развивающийся язык программирования, новые версии (с добавлением/изменением языковых свойств) выходят примерно раз в два с половиной года. Вследствие этого и некоторых других причин на Python отсутствуют стандарт ANSI, ISO или другие официальные стандарты, их роль выполняет CPython.

Разработчики языка Python придерживаются определённой философии программирования, называемой «The Zen of Python» («Дзен Питона», или «Дзен Пайтона»). Автором этой философии считается Тим Петерс.

Python портирован и работает почти на всех известных платформах – от КПК до мейнфреймов. При этом Python имеет поддержку характерных для данной платформы технологий.

Python поддерживает динамическую типизацию, то есть тип переменной определяется только во время исполнения. Поэтому вместо «присваивания значения переменной» лучше говорить о «связывании значения с некоторым именем». Добавить новый тип можно либо написав класс (class), либо определив новый тип в модуле расширения (например, написанном на языке C).

Язык обладает чётким и последовательным синтаксисом, продуманной модульностью и масштабируемостью, благодаря чему исходный код написанных на Python программ легко читаем. При передаче аргументов в функции Python использует вызов по соиспользованию, т.е. при вызове функция получает копию ссылки на объект, сам объект не копируется.

В режиме отладки, интерпретатор Python имеет интерактивный режим работы, при котором введённые с клавиатуры операторы сразу же выполняются, а результат выводится на экран (REPL).

Python 3 это последняя версия языка Python, вышедшая в конце 2008 года. Релиз ставит целью упрощение и улучшение синтаксиса языка. Некоторые из этих изменений не совместимы с предыдущей версией, поэтому Python 3 имеет собственную метку. Несмотря на то, что Python 3 готов к использованию и рекомендован сообществом для новых проектов, многие используют Python 2 из-за наследованного кода.

3.4 Интеграция с Python

Язык программирования Python был выбран как кроссплатформенное средство разработки по нескольким причинам:

– Python является объектно-ориентированным языком и позволяет решать самые разнообразные задачи;

– интерпретатор языка реализован для большинства операционных систем и платформ;

– язык является открытым и постоянно совершенствуется миллионами разработчиков, постоянно расширяется его функционал и улучшаются потребительские свойства;

– Python имеет большое количество подключаемых модулей, предоставляющих самые разные дополнительные возможности.

– для реализации интерфейса было принято решение использовать Flask – фреймворк для создания веб-приложений на языке программирования Python.

Для интеграции имеющегося кода с python, была собрана библиотекf динамической компоновки (DLL) при помощи CMake – кроссплатформенной автоматизированной системы сборки проектов. Непосредственно сборкой она не занимается, а только генерирует Makefile, который потом будет выполнен утилитой make.

На данный момент, интерфейс библиотеки составляет только одна функция (листинг 3.8). На вход подается адрес файла с условием задачи. На

выходе возвращает массив чисел типа `double`, в котором хранятся значения `NPV, E, D`, а также массивов `prj` и `inv` найденного решения.

Листинг 3.8 – Интерфейс библиотеки

```
#ifndef TEST3_LIBRARY_H
#define TEST3_LIBRARY_H
#include <string>

extern "C" __declspec (dllexport) double* SolveProblem(
    char* filename);
#endif
```

```
double* SolveProblem( char* filename) {
  InvestProblem a(filename);
  std::cout << "Считали задачу.\nНачали решение" << std::endl;
  a.Solve();
  std::cout << "Закончили решение.\nформирование массива" <<
std::endl;
  vector <int> prj = a.get_answer().get_Prj();
  vector <double> inv = a.get_answer().get_inv();
  double* res = new double [prj.size() + inv.size() + 5];
  res[0] = prj.size();
  res[1] = inv.size();
  res[2] = a.get_answer().get_NPV();
  res[3] = a.get_answer().get_E();
  res[4] = a.get_answer().get_D();
  for (int i = 5, j = 0; j < prj.size(); i++, j++){
    res[j] = prj[j];
  }
  for (int i = 5 + prj.size(), j = 0; j < inv.size(); i++, j++){
    res[i] = inv[j];
  }
  for (int i = 0; i < prj.size() + inv.size() + 5; i++){
    std::cout << res[i] << " ";
  }
  std::cout << "\nЗакончили формирование массива.\nПередача данных"
<< std::endl;

  return res;}
```

Способ обращения python к библиотеке и распознание массива представлен в листинге 3.9. Формирование данных для сайта и его активация представлены в приложении В.

Далее при использовании Flask – фреймворк создаем web-страничку, на которой выбираем файл с условием задачи (рис. 3.2). После нажатия кнопки «Решить», программа перебрасывает пользователя на другую web-

страницу, где в графическом виде представлены результаты вычислений (рис. 3.2, 3.3).

Реализация сайта и графиков приведена в приложении Г.

Рисунок 3.2 – Интерфейс приложения и график выполнения проектов

Листинг 3.9 – Обращение к библиотеке

```
from ctypes import *

func = cdll.LoadLibrary("C:/Users/zagir/CLionProjects/test3/cmake-
build-debug/libtest3.dll").SolveProblem

func.restype = POINTER(c_double * 20)

res = func( c_char_p(
"D:\Programming\BaseInvestGraph\BaseInvestGraph\input_03.csv".encode()
)).contents
```

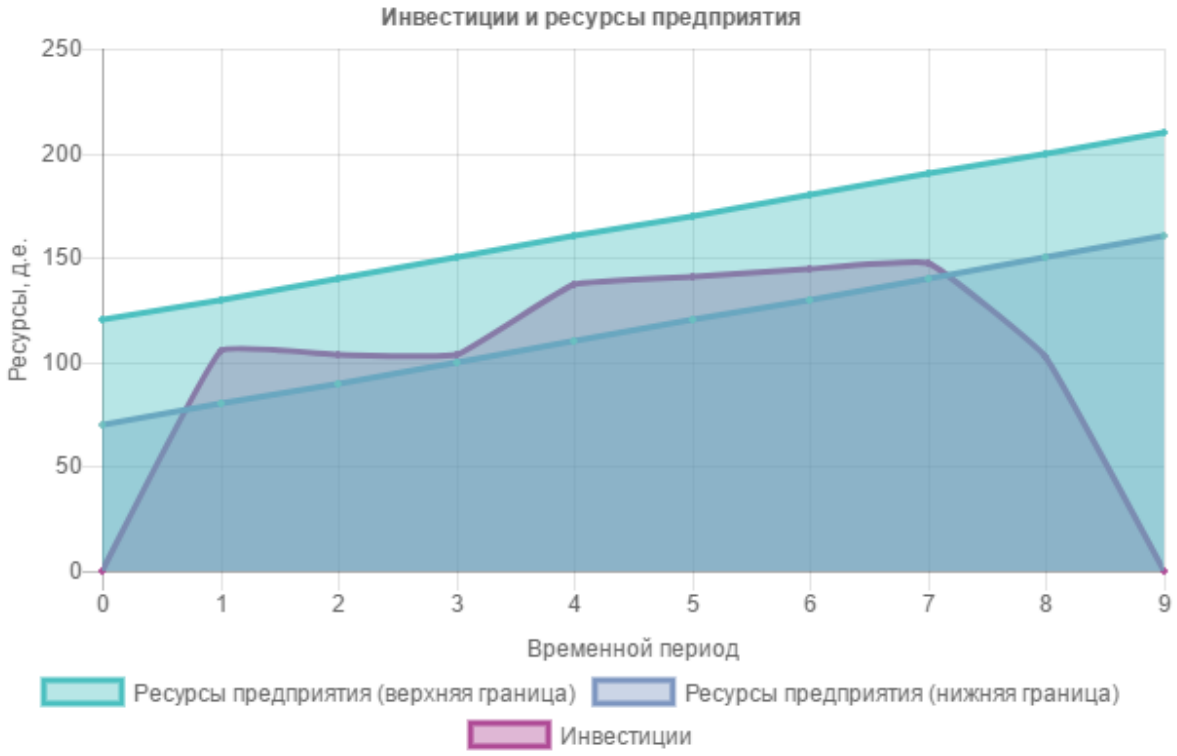

Рисунок 3.3 – Результаты вычисления в виде графика

3.5 Примеры решения

Рассмотрим применение данного кода к решению задачи. В таблице 3.1 представлена схема того, как задается условие задачи.

Данные разделены пустой строкой на пять блоков. В первом блоке четыре строки. В первой строке записан идентификатор задачи ID, количество проектов n, горизонт планирования m (начиная с нулевого периода) и значение критерия e/d, т.е. ограничение для дисперсии (5) или матожидания (6). Во второй – продолжительность каждого проекта L, в третьей и в чет-вертой – нижняя Rl и верхняя Ru граница имеющихся у предприятия ресурсов в каждый период времени. Во втором блоке приводятся нижние оценки ЧДД проектов NPV_l в каждый период времени возможного начала реализации. В каждой строке отображена информация по одному проекту. В третьем блоке приводятся верхние оценки ЧДД проек-тов NPV_u. В четвертом блоке записан нижний уровень затрат Inv_l на каждый проект в каждый период времени, начиная с момента активации проекта по тому же принципу, что и

предыдущий блок. В пятом блоке записана верхняя граница затрат на каждый проект Inv_u.

Таблица 3.1 – Схема условия задачи

ID	n	m	e / d
L1	L2	…	Ln
Rl 1	Rl 2	…	Rl m
Ru 1	Ru 2	…	Ru m
NPV_l_1_0	NPV_l_1_1	…	NPV_l_1_m-L1
…	…	…	…
NPV_l_n_0	NPV_l_n_1	…	NPV_l_n_m-Ln
NPV_u_1_0	NPV_u_1_1	…	NPV_u_1_m-L1
…	…	…	…
NPV_u_n_0	NPV_u_n_1	…	NPV_u_n_m-Ln
Inv_l_1_0	Inv_l_1_1	…	Inv_l_1_L1
…	…	…	…
Inv_l_n_0	Inv_l_n_1	…	Inv_l_n_Ln
Inv_u_1_0	Inv_u_1_1	…	Inv_u_1_L1
…	…	…	…
Inv_u_n_0	Inv_u_n_1	…	Inv_u_n_Ln

Для проверки быстродействия программы были сгенерированы тестовые задачи. В данных задачах требовалось спланировать инвестиционную программу предприятия на 10 лет вперед для разного количества учтенных в пакете проектов: начиная от трех и заканчивая восемью. Пример условия одной из тестовых задач представлен на рисунке 3.4.

Результаты решения задачи для пакета из шести инновационных проектов представлены на рисунке 3.5. В первом столбце показан номер проекта, во втором – период, в который следует начать проект. *-1* означает то, что проект не должен быть выполнен. При формировании инвестиционной программы по максиминной стратегии гарантированный ЧДД программы равен 80 (рис. 5.а). Это решение совпало с решением задачи поиска минимальной

дисперсии при условии, что математическое ожидание будет больше 80 (рис. 5.в). При решении задачи поиска максимального математического ожидания с ограничением дисперсии в пределах 40 получили гарантированный ЧДД 47, ожидаемую прибыль – 57 (рис. 5.б).

	A	B	C	D	E	F	G	H	I	J
1	m	4	10	100						
2	6	7	8	6						
3	70	80	90	100	110	120	130	140	150	160
4	120	130	140	150	160	170	180	190	200	210
5										
6	30	26	23	20	18					
7	40	35	32	29						
8	47	43	40							
9	40	36	33	30	28					
10										
11	40	36	33	34	31					
12	55	46	42	43						
13	57	58	54							
14	55	51	48	45	43					
15										
16	50	55	50	55	50	55				
17	40	45	35	40	45	35	45			
18	43	43	43	43	43	43	43	43		
19	47	52	47	52	47	52				
20										
21	61	68	63	66	65	70				
22	53	59	50	50	57	46	55			
23	55	54	54	54	55	56	55	58		
24	53	58	53	58	53	58				

Рисунок 3.4 – Пример условия задачи на 3 проекта, 10 периодов

NPV	80
Expectancy	95
Dispersion	41.6667
Project	time
1	-1
2	-1
3	0
4	-1
5	-1
6	4

а)

NPV	47
Expectancy	57
Dispersion	33.3333
Project	time
1	-1
2	-1
3	-1
4	-1
5	0
6	-1

б)

NPV	80
Expectancy	95
Dispersion	41.6667
Project	time
1	-1
2	-1
3	0
4	-1
5	-1
6	4

в)

Рисунок 3.5 – Решение задачи
а) максимин, б) матожидание, в) дисперсия

Выводы по главе три

Предложенная реализация позволяет найти оптимальный пакет инвестиционных проектов при условии гарантированности результата, а также в условиях неопределенности. Данная реализация не позволяет решать задачи большой размерности, что показано на рисунке 3.6. Заметим, что в течении работа цикла, каждая итерация продолжается дольше, чем предыдущая, но скорость роста уменьшается до 20% к количеству проектов равному 8 (рис. 3.7).

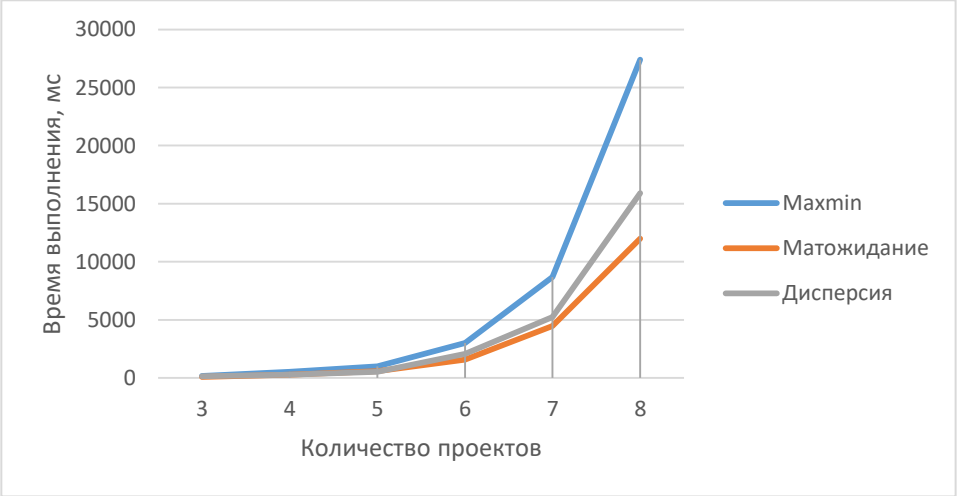

Рисунок 3.6 – Время выполнения программы

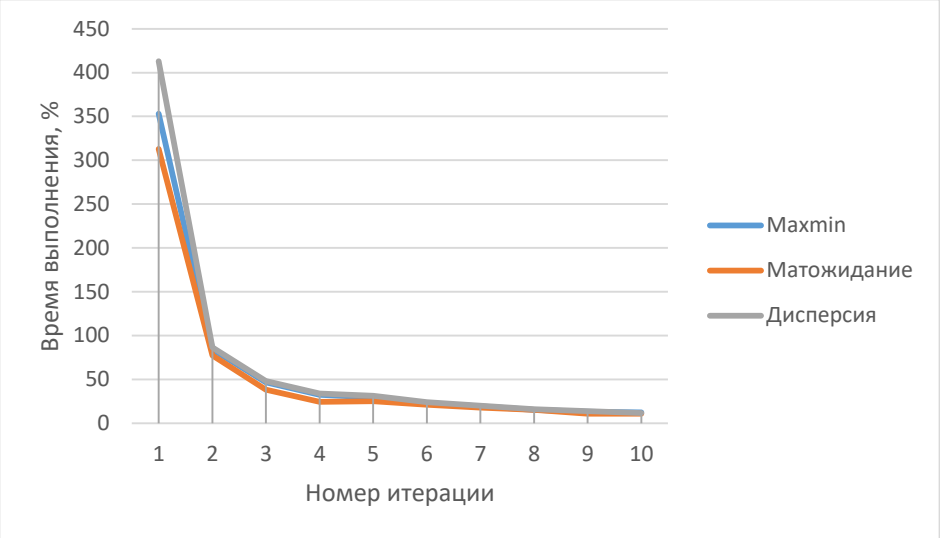

Рисунок 3.7 – Скорость выполнения итерации цикла по отношению к предыдущей итерации, %

Заключение

В работе были рассмотрены некоторые модели поиска оптимальной стратегии инвестирования предприятия во множество проектов в течении продолжительного периода времени. Поставленные **цели** были достигнуты, а именно была выполнена:

– разработка алгоритмов решения задачи поиска оптимального инвестиционного пакета проектов как при условии гарантированности результата, так и в условии неопределенности,

– разработка программного приложения, позволяющего решать задачи данного направления любому пользователю,

Также, в процессе работы были выполнены следующие **задачи**:

– подготовка теоритического обзора материала по данной теме;

– разработка структуры программы для вычислений: спецификация переменных, структура исходных данных, основные классы;

– разработка программы для операционной системы Windows с последовательными алгоритмами с применением объектно-ориентированного подхода;

– разработка алгоритмы решения задачи поиска оптимальной инвестиционной программы.

Работа может быть использована для решения задач поиска оптимального пакета инвестиционных проектов, с вложенной возможностью графического представления решения. Далее планируется реализация программного обеспечения, позволяющего решать задачи большой размерности за счет распараллеливания процессов, а также реализовать методы поиска оптимального портфеля проектов с учетом вероятности недостижимости заданного уровня чистого дисконтированного дохода.

Библиографический список

1. Аристов, С.А. Принципы моделирования инвестиционных программ предприятия /С.А. Аристов // Экономика и математические методы. – 2009. – № 3. – С. 74–83.

2. Афанасьев М.А. Оптимальная инвестиционная программа / М.А. Афанасьев // Инвестиции в России. – 2002. – № 12. – С. 75–79.

3. Кибзун, А.И. Алгоритм решения обобщенной задачи Марковица / А. И. Кибзун, А. И. Чернобровов // Автоматика и телемехника. – 2011. – № 2. – С. 77–92.

4. Новоселов, А.А. Математическое моделирование финансовых рисков: Теория измерения. / А.А. Новоселов – Новосибирск: Наука, 2001. – 102 с.

5. Dilger R. J., Gonzales O. R. SBA Small Business Investment Company Program // Journal of Current Issues in Finance, Business and Economics. 2012. Vol. 5. № 4. C. 407-417.

6. Михалина, Л.М. Изменение инвестиционной активности субъектов малого бизнеса в условиях коррупционной среды / Л. М. Михалина, Е. Б. Голованов // Вестник Южно-Уральского государственного университета. Серия: Экономика и менеджмент. 2014. Том 8 (3). С. 41-47.

7. Schlegel D., Frank F., Britzelmaier B. Investment decisions and capital budgeting practices in German manufacturing companies. // International Journal of Business and Globalisation. 2016. Vol. 16. №. 1. C. 66-78.

8. Юзвович, Л.И. Экономическая природа и роль инвестиций в национальной экономической системе / Л.И. Юзвович // Финансы и кредит. 2010. № 9. С. 48-52.

9. Гамилова, Д.А. Обоснование эффективности проведения инвестиционной политики предприятия / Д.А. Гамилова // Инновации и инвестиции. 2010. № 2. С. 37-41.

10. Сергеева, Д.П. Методы оценки эффективности инвестиционных проектов с учетом рекомендаций минэкономики / Д.П. Сергеева // Инновационная наука. 2015. № 9. С. 201-204.

11. Желнова, К.В. Анализ практики принятия решений в области инвестиционной политики / К.В. Желнова // Вопросы современной экономики. 2013. № 4. С. 38-46.

12. Методические рекомендации по оценке эффективности инвестиционных проектов (утв. Минэкономики РФ, Минфином РФ, Госстроем РФ 21.06.1999 N ВК 477). URL: //http: www.consultant.ru/document/cons_doc_LAW_28224/ (дата обращения: 30.07.2016).

13. Панюков, А.В. Математическое моделирование экономических процессов. / А.В. Панюков – Москва: URSS, 2009. –191 с.

14. Panyukov A. V. Teleghin V.A. Forming of Discrete Mechanical Assembly Production Program // Journal of Computational and Engineering Mathematics. 2015. Vol. 2. № 1. С. 57-64.

15. Афанасьев М. А. Оптимальная инвестиционная программа / М.А. Афанасьев // Инвестиции в России. 2002. № 12. С. 50-54.

16. Глухов, В.В. Математические методы и модели для менеджмента. / В.В. Глухов – СПб: Лань, 2005. – 528 с.

17. Козина, Е.Н. Разработка экономико-математической модели инвестиционной программы предприятия / Е. Н. Козина // Сб. трудов международной научно-практической конференции «Современные тенденции экономики, права, управления, математики: новый взгляд. Санкт-Петербург, 01-03 декабря». – СПб.: Изд-во «КультИнформПресс», 2012. С. 64-67.

18. Кибзун, А.И. Задачи стохастического программирования с вероятностными критериями. / Кибзун А.И. – М.: Физматлит, 2009. – 372 с.

19. Козина, Е.Н. Математическая модель инвестиционной программы предприятия / Е.Н. Козина //Сб. трудов XII Всероссийского совещания по проблемам управления (Москва, 16-19 июня 2014). М.: ИПУ РАН, 2014. С. 1250-1253.

20. Вишняков, Б.В. Детерминированные эквиваленты для задач стохастического программирования с вероятностными критериями / Б.В. Вишняков, А.И. Кибзун // Автоматика и телемеханика. – 2006. – № 6. – С. 126–143.

21. A. V. Panyukov and E. N. Kozina Forming of the Competitive Investment Programs for Enterprises Workshop on Computer Modelling in Decision Making (CMDM 2016). CEUR Workshop Proceedings. Vol. 1726. pp. 89-99. URL http://ceur-ws.org/Vol-1726/paper-09.pdf

22. Панюков, А.В. Программная реализация максиминной стратегии управления инвестиционной деятельностью предприятия [текст]/ А.В. Панюков, Е.А. Загирова // Информационные технологии и системы. – 2017. – С. 211-219.

23. Python // Официальный сайт Python [Эл. ресурс]/ Режим доступа: https://www.python.org/about/, свободный. – Загл. с экрана.

24. Python 3.6.1 documentation [Электронный ресурс] / Режим доступа: https://docs.python.org/3/, свободный. – Загл. с экрана.

25. Аникушин, М. Коды Грея и задачи перебора [Электронный ресурс] / М. Аникушин. URL: https://habrahabr.ru/post/200806/

26. Рейнгольд, Э. Комбинаторные алгоритмы. Теория и практика / Э. Рейнгольд, Ю. Нивергельт, Н. Део. – М. : Мир, 1980. – 476 с.

27. Англичанинов, В. В. Развитие методологии формирования инвестиционной программы промышленного предприятия [Эл. ресурс]/ В.В. Англичанинов // Вестник Казанского технологического университета. – 2009 – С. 222 – 228.

28. Покровский, А.М. Практические оптимизационные подходы к корпоративным инвестиционным программам [Эл. ресурс] / А.М. Покровский // Транспортное дело России. – 2011 – с. 38 – 41.

29. Брейли, Р. Принципы корпоративных финансов / Брейли Р., Майерс С. – М.: Олимп-бизнес, – 1997 – 1088с.

ПРИЛОЖЕНИЯ

Приложение А

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Факультет «Математика, механика и компьютерные технологии»
Кафедра «Математическое и компьютерное моделирование»

Разработка программного обеспечения для решения задачи выбора инвестиционной программы предприятия

ТЕКСТ ПРОГРАММЫ
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

ЮУрГУ–01.04.02.2017:115-052.04.000 МД

Руководитель проекта:
Профессор, д.ф. – м.н.,
_____ А.В. Панюков
« _____» _____ 2017 г.

Автор работы
студент группы ЕТ-224
_____ Е.А. Загирова
« _____» _____ 2017 г.

Нормоконтролер
Доцент, к.ф. – м.н.,
_____ Макаровских Т.А.
« _____» _____ 2017 г.

Челябинск
2017

Файл BaseInvGraph.h

```
#pragma once
#include "Data.h"
#include <vector>
#include <list>
using namespace std;

class SolveFunctions
{
protected:
   Data *data;
   list <Status> *statusList;
   list <Status> *statusListNextStep;

public:
   void initiationGray(const int time, const int k, const vector <int>&
baseVec);
    void initiation(); //инициализация списка
    void nextStep( const int time); // следующий шаг в решении - новый список
состояний
   void generateProjectsTimes(int time, const vector <int>& baseVec, vec-
tor<vector<int>>& List);

   virtual void addToStatusList(const Status &status) = 0;
   virtual void adaptation (const vector <int>& baseVec, const vector <int>&
g) = 0;
   virtual bool checkStatus (Status& status) = 0;

    SolveFunctions(){};
   SolveFunctions(Data &d,
      list <Status> &s,
      list <Status> &n): data(&d), statusList(&s), statusListNextStep(&n){};
   ~SolveFunctions(void){};

   Data& get_data(){ return *data;}
   list <Status>& stList(){ return *statusList;}
   list <Status>& stListNext(){ return *statusListNextStep;}
};

class FunctionsMaxMin: public SolveFunctions
{
public:
   void addToStatusList(const Status &status);
   void adaptation (const vector <int>& baseVec, const vector <int>& g);
   bool checkStatus (Status& status);

    FunctionsMaxMin(){};
   FunctionsMaxMin( SolveFunctions &a){
      //data = (a).get data();

   };
   FunctionsMaxMin(Data &d,
      list <Status> &s,
      list <Status> &n): SolveFunctions( d, s, n){};
   ~FunctionsMaxMin(void){};
};

class FunctionsExpect: public SolveFunctions
```

```
{
public:
   void addToStatusList(const Status &status);
   void adaptation (const vector <int>& baseVec, const vector <int>& g);
   bool checkStatus (Status& status);
    FunctionsExpect(){};
   FunctionsExpect(Data &d,
      list <Status> &s,
      list <Status> &n): SolveFunctions( d, s, n){};
   ~FunctionsExpect(void){};
};

class FunctionsDisp: public SolveFunctions
{
public:
   void addToStatusList(const Status &status);
   void adaptation (const vector <int>& baseVec, const vector <int>& g);
   bool checkStatus (Status& status);
    FunctionsDisp(){};
   FunctionsDisp(Data &d,
      list <Status> &s,
      list <Status> &n): SolveFunctions( d, s, n){};
   ~FunctionsDisp(void){};
};

class InvestProblem
{
   Data data;
   Status answer;
   list <Status> statusList;
   list <Status> statusListNextStep;

   SolveFunctions *functions;

public:
   InvestProblem(void){
        functions = new FunctionsMaxMin(data, statusList, statusList-
NextStep);
    }
   InvestProblem(string filename):data(filename){
      switch (data.ProblemID)
      {
      case 'm':{functions = new FunctionsMaxMin(data, statusList, statusList-
NextStep); break;}
      case 'e':{functions = new FunctionsExpect(data, statusList, statusList-
NextStep); break;}
      case 'd':{functions = new FunctionsDisp(data, statusList, statusList-
NextStep); break;}
      default:
         break;
      }
   };
   ~InvestProblem(void){
      delete functions;
   };

   Data get_data() {return data;}
   Status get_answer() {return answer;}
   list <Status> get_statusList() {return statusList;}
   void Solve();
};
```

Файл BaseInvGraph.cpp

```
#include "BaseInvGraph.h"
#include <iostream>
#include <ctime>
#include <fstream>
using namespace std;

void InvestProblem::Solve(){
   ofstream write("time.csv", ios::app);

   unsigned int start_time =  clock();//засекаем время

   functions->initiation();

   unsigned int end_time = clock();
   write << data.nPrj << " projects"<< ';' << (end_time - start_time) <<
';';//выводим время

    for (int i = 0; i < data.m; i++ ){
      cout << "NEXT STEP - " << i<< '\n';
        functions->nextStep(i+1);

      end_time = clock();
      write << (end_time - start_time) << ';';//выводим время
    }

   if (data.ProblemID == 'd'){
      for (list <Status>::iterator i = statusList.begin(); i != sta-
tusList.end();){
         if ( (*i).get_E() < data.criterion){
            list <Status>::iterator j = i;
            i++;
            statusList.erase(j);
         }
         else
            i++;
      }
   }

   if (!statusList.empty()){
      answer = (*statusList.begin());
   }
   write << '\n';
   write.close();
}
void SolveFunctions::initiation(){
   vector <int> prj(get_data().nPrj, -1);
   Status status(prj, get_data());
   stListNext().push_back(status);

   vector <vector <int>> List;
   generateProjectsTimes(0, prj, List);
   for (int j = 1; j < List.size(); j++){
      //преобразование нового варианта
      adaptation(prj, List[j]);
   }

    stList() = stListNext();
```

```
    stListNext().clear();
}
void SolveFunctions::nextStep( const int time){
   //Для каждого элемента списка состояний
    for (list <Status>::iterator i = stList().begin(); i != stList().end();
i++){
        vector <int> baseVec = (*i).get_Prj();
      stListNext().push_back(*i);

      vector<vector<int>> List;
      generateProjectsTimes(time, baseVec, List);
      for (int j = 1; j < List.size(); j++){
         //преобразование нового варианта
         adaptation(baseVec, List[j]);
      }}
    stList() = stListNext();
    stListNext().clear();
}
 void SolveFunctions::generateProjectsTimes(
    int time,
    const vector <int>& baseVec,
    vector<vector<int>>& List){
   //количество проектов, которые никогда не активировались
    int k = 0; for (int i = 0; i < baseVec.size(); i++) if (baseVec[i] == -1)
k++;

   vector <int> empty(k, -1);
   List.push_back(empty);

   for (int i = 0; i < k; i++){
      int n = List.size();
      for (int j = 0; j < n; j++){
         List.push_back(List[j]);
         List[n+j][i] = time;
      }}}
//////// MAXIMIN
void FunctionsMaxMin::addToStatusList(const Status &status){
   if ((*(stListNext().begin())).get_NPV() > status.get_NPV())
      //добавляем в список состояний
      stListNext().push_back(status);
   else
      stListNext().push_front(status);
}

void FunctionsMaxMin::adaptation (const vector <int>& baseVec, const vector
<int>& g){
   vector <int> vec;
    bool endPrj = true;
    //составляем новый вектор состояния проектов
    for (int i = 0, j = 0; i < baseVec.size(); i++){
        vec.push_back(baseVec[i]);
        if (baseVec[i] == -1){
            //если проект умещается в сроки
            if ((get_data().m) - g[j] > (get_data().lenght[i])){
                vec[i] = g[j];
                j++;
            }
            else{
                endPrj = false;//проект вышел за горизонт планирования
                break;
            }
```

```
            }
        }
        if (endPrj){
            Status status(vec, (get_data()) );
          if (checkStatus(status))
             addToStatusList(status);
        }
}
bool FunctionsMaxMin::checkStatus (Status& status){
   bool result = true;
    vector <double> Res = status.get_inv();

    for (int i = 0; i < Res.size(); i++){
      if ( Res[i] > get_data().Resource[i].Inf()) {
            result = false;
            break;
        }
    }
   return result;
}
////////  EXPECTANCY
void FunctionsExpect::addToStatusList(const Status &status){
   if ((*(stListNext().begin())).get_E() > status.get_E())
      //добавляем в список состояний
      stListNext().push_back(status);
   else
      stListNext().push_front(status);
}
void FunctionsExpect::adaptation (const vector <int>& baseVec, const vector
<int>& g){
   vector <int> vec;
    bool endPrj = true;
    //составляем новый вектор состояния проектов
    for (int i = 0, j = 0; i < baseVec.size(); i++){
        vec.push_back(baseVec[i]);
        if (baseVec[i] == -1){
            //если проект умещается в сроки
            if ((get_data().m) - g[j] > (get_data().lenght[i])){
                vec[i] = g[j];
                j++;
            }
            else{
                endPrj = false;//проект вышел за горизонт планирования
                break;
            }
        }
    }
    if (endPrj){
        Status status(vec, (get_data()) );
      if (checkStatus(status))
         addToStatusList(status);
    }
}
bool FunctionsExpect::checkStatus (Status& status){
   bool result = true;
    vector <double> Res = status.get_inv();

    for (int i = 0; i < Res.size(); i++){
      if ( Res[i] > get_data().Resource[i].Inf()) {
            result = false;
            break;
```

```
        }
    }

   if (status.get_D() > get_data().criterion)
      result = false;

   return result;
}
////////  DISPERSION
void FunctionsDisp::addToStatusList(const Status &status){
   bool push = false;
   for (list <Status>::iterator i = stListNext().begin(); i != stList-
Next().end(); i++){
      if ((*(i)).get_D() > status.get_D()){
         //добавляем в список состояний
         stListNext().insert(i, status);
         push = true;
         break;
      }
   }
   if (!push)
      stListNext().push_back(status);
}
void FunctionsDisp::adaptation (const vector <int>& baseVec, const vector
<int>& g){
   vector <int> vec;
    bool endPrj = true;
    //составляем новый вектор состояния проектов
    for (int i = 0, j = 0; i < baseVec.size(); i++){
        vec.push_back(baseVec[i]);
        if (baseVec[i] == -1){
            //если проект умещается в сроки
            if ((get_data().m) - g[j] > (get_data().lenght[i])){
                vec[i] = g[j];
                j++;
            }
            else{
                endPrj = false;//проект вышел за горизонт планирования
                break;
            }}
    }
    if (endPrj){
        Status status(vec, (get_data()) );
      if (checkStatus(status))
         addToStatusList(status);
    }}
bool FunctionsDisp::checkStatus (Status& status){
   bool result = true;
    vector <double> Res = status.get_inv();

    for (int i = 0; i < Res.size(); i++){
      if ( Res[i] > get_data().Resource[i].Inf()) {
            result = false;
            break;
        }}
return result;
}
```

Приложение Б

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Факультет «Математика, механика и компьютерные технологии»
Кафедра «Математическое и компьютерное моделирование»

Разработка программного обеспечения для решения задачи выбора инвестиционной программы предприятия

ТЕКСТ ПРОГРАММЫ
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

ЮУрГУ–01.04.02.2017:115-052.04.000 МД

Руководитель проекта:
Профессор, д.ф. – м.н.,
_____ А.В. Панюков
« _____» _____ 2017 г.

Автор работы
студент группы ЕТ-224
_____ Е.А. Загирова
« _____» _____ 2017 г.

Нормоконтролер
Доцент, к.ф. – м.н.,
_____ Макаровских Т.А.
« _____» _____ 2017 г.

Челябинск
2017

Файл Data.h

```
#pragma once
#include <vector>
#include <list>
#include <string>
#include "IntervalClasses.h"
using namespace std;



class Data
{
public:
   char ProblemID;//идентификатор задачи: m - maxmin, e - expectation, d - 
dispersion, p - prob
   double criterion;//ограничения для дисперсии, матожидания
   int nPrj;//количество проектов ИЛИ вектор названий проектов???
   int m; // горизонт планирования
   double riskLow, lvlNPV;//риск недостижимости уровня lvlNPV

   vector <double> ResSupBound;// риск превышения ресурсов, размера nPrj
   vector <double> riskUp; // риск превышения ресурсов, размера m
   vector <int> lenght; //длительность проектов, вектор размера nPrj
   IVector Resource; //финансовые ресурсы компании, вектор размера m

   vector <IVector> NPV; //net present value чистый дисконтированный доход
   vector <IVector> Investments; //ежегодные инвестиции(вложения) в проект

   Data(void){
        ProblemID = 'm';
        //readInterCSV("input.csv");
    }
   Data( Data &a){
      ProblemID = a.ProblemID;
      criterion = a.criterion;
      nPrj = a.nPrj;
      m = a.m;
      riskLow = a.riskLow;
      lvlNPV = a.lvlNPV;
      ResSupBound = a.ResSupBound;
      riskUp = a.riskUp;
      lenght = a.lenght;
      Resource = a.Resource;
      NPV = a.NPV;
      Investments = a.Investments;
   }

   Data(string filename){ readInterCSV(filename);}
   ~Data(void){};

   void readInterCSV (string filename);
};
```

```
class Status
{
   vector <int> Prj;//активные, не активные, завершенные проекты
   double NPV;//чистый дисконтированный доход пакета проектов
   vector <double> inv;//ресурсы, затрачиваемые проектами в момент времени i,
inv[i]
   double E;//мат. ожидание
   double D;//дисперсия
public:
   Status(void){};
   /*Status(Status &a){
      Prj = a.Prj;
      NPV = a.NPV;
      inv = a.inv;
      E = a.E;
      D = a.D;
   };*/
   Status(const vector <int>& _Prj, const Data& data);
   ~Status(void){};

   vector <int> get_Prj(){ return Prj;}
   vector <double> get_inv(){ return inv;}
   double get_NPV(){ return NPV;}
   double get_E(){ return E;}
   double get_D(){ return D;}

   vector <int> get_Prj() const { return Prj;}
   vector <double> get_inv() const { return inv;}
   double get_NPV() const { return NPV;}
   double get_E() const { return E;}
   double get_D() const { return D;}

   void writeCSV( string filename = "output_status.csv");

   bool const operator <(Status const &right) const {
      if ((this->NPV) < right.get_NPV())
         return 1;
      else
         return 0;
   }
   bool const operator >(Status const &right) const {
      if ((this->NPV) > right.get_NPV())
         return 1;
      else
         return 0;
   }
   bool operator ==(Status const &right) const{
       bool result = true;
      if ((this->inv) == right.inv
      && (this->NPV) == right.NPV
      && (this->Prj)== right.Prj){
          return 1;
      }
      else{
        return 0;
      }

      void writeStatus();

   }
};
```

Файл Data.cpp

```
#include "Data.h"
#include <fstream>
#include <string>
#include <sstream>
#include <stdio.h>
#include <stdlib.h>


Status::Status(const vector <int>& _Prj, const Data& data){
   Prj = _Prj;
   NPV = 0;
   E = 0;
   D = 0;
   //ресурсы, требуемые данным составом проектов в каждый момент времени
   for (int i = 0; i < data.m; i++) inv.push_back(0);
   for (int i = 0; i < _Prj.size(); i++){
      //если проект был начат
        if (Prj[i] != -1){
         int j = _Prj[i] + data.lenght[i];
            //во время выполнения проекта требуются ресурсы
            for (int k = _Prj[i]; k < j; k++)
            inv[k] += data.Investments[i][k - _Prj[i]].Sup();
         NPV += data.NPV[i][ Prj[i]].Inf();//находим общий ЧДД
         E += (data.NPV[i][_Prj[i]].Inf() + data.NPV[i][_Prj[i]].Sup())/2;

         double a = data.NPV[i][_Prj[i]].Sup() - data.NPV[i][_Prj[i]].Inf();
         D += (a*a)/12;
            }
       }
}


void Status::writeCSV(string filename){
   ofstream write(filename, ios::app);
   write << "NPV;" << NPV <<'\n';
   write << "Expectancy;" << E <<'\n';
   write << "Dispersion;" << D <<'\n';
   write << "Project;time"<<'\n';
   for (int i = 0; i < Prj.size(); i++){
      write << (i + 1) << ";" << Prj[i] <<'\n';
   }
   write.close();
}


void Data::readInterCSV( string filename)
{
   int x;
   char h = ';';
   ifstream read(filename);
   if (read.is_open()){
      char separator = ';';
      string str, s;

      std::getline(read, str);
      stringstream line_streamed(str);
      std::getline(line_streamed, s, separator);
      ProblemID = s[0];
      std::getline(line_streamed, s, separator);
```

```
nPrj = atoi(s.c_str());
std::getline(line_streamed, s, separator);
m = atoi(s.c_str());
std::getline(line_streamed, s, separator);
criterion = atoi(s.c_str());

//считываем продолжительность проектов
std::getline(read, str);
line_streamed << str;
for (int i = 0; i < nPrj; i++){
   std::getline(line_streamed, s, separator);
   lenght.push_back(atoi(s.c_str()));
}
line_streamed.clear();

vector <double> infRes;
vector <double> supRes;
//считываем имеющиеся ресурсы нижняя граница
std::getline(read, str);
line_streamed << str;
for (int i = 0; i < m; i++){
   std::getline(line_streamed, s, separator);
   infRes.push_back(atoi(s.c_str()));
}
line_streamed.clear();

//считываем имеющиеся ресурсы верхняя граница
std::getline(read, str);
line_streamed << str;
for (int i = 0; i < m; i++){
   std::getline(line_streamed, s, separator);
   supRes.push_back(atoi(s.c_str()));
}
line_streamed.clear();

//Записываем ресурсы
for (int i = 0; i < m; i++){
   Interval a(infRes[i], supRes[i]);
   Resource.addInterval(a);
}
std::getline(read, str);

vector <vector <double> > inf;
vector <vector <double> > sup;
//считываем нижнюю границу ЧДД
for (int i = 0; i < nPrj; i++){
   int a = m - lenght[i] + 1;
   vector <double> v;
   std::getline(read, str);
   stringstream line(str);
   for (int j = 0; j < a; j++){
      std::getline(line, s, separator);
      v.push_back(atoi(s.c_str()));
   }
   inf.push_back(v);
}
std::getline(read, str);
//h = read.get(); while (h != '\n') h = read.get();

//считываем верхнюю границу ЧДД
for (int i = 0; i < nPrj; i++){
```

```
            std::getline(read, str);
            stringstream line(str);
            int a = m - lenght[i] + 1;
            vector <double> v;
            for (int j = 0; j < a; j++){
               std::getline(line, s, separator);
               v.push_back(atoi(s.c_str()));
            }
            line_streamed.clear();
            sup.push_back(v);
         }
         std::getline(read, str);
         //запись ЧДД
         for (int i = 0; i < nPrj; i++){
            int s = m - lenght[i] + 1;
            IVector v;
            for (int j = 0; j < s; j++){
               Interval a(inf[i][j], sup[i][j]);
               v.addInterval(a);
            }
            NPV.push_back(v);}

         inf.clear();
         sup.clear();
         //считываем инвестиции в проект НИЖНЯЯ граница
         for (int i = 0; i < nPrj; i++){
            std::getline(read, str);
            stringstream line(str);
            vector <double> v;
            for (int j = 0; j < lenght[i]; j++){
               std::getline(line, s, separator);
               v.push_back(atoi(s.c_str()));
            }
            inf.push_back(v);
         }
         std::getline(read, str);
         //считываем инвестиции в проект ВЕРХНЯЯ граница
         for (int i = 0; i < nPrj; i++){
            std::getline(read, str);
            stringstream line(str);
            vector <double> v;
            for (int j = 0; j < lenght[i]; j++){
               std::getline(line, s, separator);
               v.push_back(atoi(s.c_str()));
            }
            sup.push_back(v);
         }
         //запись инвестиций
         for (int i = 0; i < nPrj; i++){
            IVector v;
            for (int j = 0; j < lenght[i]; j++){
               Interval a(inf[i][j], sup[i][j]);
               v.addInterval(a);
            }
            Investments.push_back(v);
         }}
read.close();
}
```

Приложение В

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Факультет «Математика, механика и компьютерные технологии»
Кафедра «Математическое и компьютерное моделирование»

Разработка программного обеспечения для решения задачи выбора инвестиционной программы предприятия

ТЕКСТ ПРОГРАММЫ
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

ЮУрГУ–01.04.02.2017:115-052.04.000 МД

Руководитель проекта:
Профессор, д.ф. – м.н.,
_____ А.В. Панюков
« _____» _____ 2017 г.

Автор работы
студент группы ЕТ-224
_____ Е.А. Загирова
« _____» _____ 2017 г.

Нормоконтролер
Доцент, к.ф. – м.н.,
_____ Макаровских Т.А.
« _____» _____ 2017 г.

Челябинск
2017

Файл main.py

```
from ctypes import *
from flask import Flask, render_template


def data_work():
    path =
"D:\Programming\BaseInvestGraph\BaseInvestGraph\input_07.csv"
    try:
        f = open(path, 'r')
    except IOError:
        print('Не может открыть файл')
    row = f.readline().split(';')
    n = int(row[1])
    m = int(row[2])
    lenght, ResLow, ResUp = [], [], []
    lenghtS, ResLowS, ResUpS = [], [], []
    lenghtS = f.readline().split(';')
    ResLowS = f.readline().split(';')
    ResUpS = f.readline().split(';')
    i = 0
    while i < n:
        lenght.append(int(lenghtS[i]))
        i = i + 1
    i = 0
    while i < m:
        ResLow.append(int(ResLowS[i]))
        i = i + 1
    i = 0
    while i < m:
        ResUp.append(int(ResUpS[i]))
        i = i + 1
    MasSize = 3 + n + m
    func =
cdll.LoadLibrary("C:/Users/zagir/CLionProjects/test3/cmake-
build-debug/libtest3.dll").SolveProblem
    func.restype = POINTER(c_double * MasSize)
    res = func(c_char_p(path.encode())).contents
    print("Передача данных выполнена")

    return [res[0:3], lenght, res[3:(3 + n)], res[(3 + n):],
ResLow, ResUp]


app = Flask(__name__)

@app.route("/")
def chart():
    data = data_work()
```

```
    context = {}
    context['values'] = [data[5], data[4], data[3]]
    context['labels'] = [c for c in range(len(data[5]))]
    context['legend'] = ['Ресурсы предприятия (верхняя грани-
ца)', 'Ресурсы предприятия (нижняя граница)', 'Инвестиции']
    context['data'] = data[0]
    context["data"][2] = round( context["data"][2], 2)

    rangeBar = []
    v = {}
    j = 0
    for i in data[2]:
        if int(i) != -1:
            v["low"] = int(i)
            v["high"] = int(i) + int(data[1][j]) - 1
        else:
            v["low"] = int(0)
            v["high"] = int(0)
        rangeBar.append(v.copy())
        j = j + 1
    context["rangebar"] = rangeBar

    return render_template('page.html', context = context)


if __name__ == "__main__":
    app.run(debug=True)
```

Приложение Г

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Факультет «Математика, механика и компьютерные технологии»
Кафедра «Математическое и компьютерное моделирование»

Разработка программного обеспечения для решения задачи выбора инвестиционной программы предприятия

ТЕКСТ ПРОГРАММЫ
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

ЮУрГУ–01.04.02.2017:115-052.04.000 МД

Руководитель проекта:
Профессор, д.ф. – м.н.,
_____ А.В. Панюков
« _____» _____ 2017 г.

Автор работы
студент группы ЕТ-224
_____ Е.А. Загирова
« _____» _____ 2017 г.

Нормоконтролер
Доцент, к.ф. – м.н.,
_____ Макаровских Т.А.
« _____» _____ 2017 г.

Челябинск
2017

Файл base.html

```
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; char-
set=utf-8" />
    <title> {% block title %}{% endblock %}</title>
    <!-- import plugin script -->
    <script src='/static/js/Chart.js'></script>
    <script src="/static/js/anychart.min.js"
type="text/javascript"></script>
    <!-- 1. Подключаем скомпилированный и минимизированный файл
CSS Bootstrap 3 -->
    <link href="/static/css/bootstrap.min.css" rel="stylesheet">
</head>
<body>

{% block content %}
{% endblock %}

<!-- 2. Подключаем библиотеку jQuery, необходимую для работы
скриптов Bootstrap 3 -->
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.
min.js"></script>
<!-- 3. Подключаем скомпилированный и минимизированный файл Ja-
vaScript платформы Bootstrap 3 -->
<script src="/static/js/bootstrap.min.js"></script>


</body>
</html>
```

Файл page.html

```
{% extends "base.html" %}
{% block title %}Задача{% endblock %}
{% block content %}
<center>
    <h3>Задача оптимизации управления пакетом инвестиционных проектов</h3>

            <form id = "auth" method="get" action="/" class="form-horizontal">
            <p><h5>Выберите файл формата *.csv с условием задачи</h5></p>
            <center>
                <div class="form-group">
                    <div class="col-xs-2 col-sm-2 col-md-2 col-lg-2"></div>
                        <label for="Stocks" class="col-xs-2 control-label"></label>
                            <div class="col-xs-4 clearfix">
                            <p><input type = "file" name="problem" form = "auth" class="form-control" id="prob"></p>
                                <!--<p><input type = "text" name="path" ></p>-->
                                <p><input type="submit" value="Решить" form = "auth" class="form-control"></p>
                            </div>
                    </div>
                </div>
            </center>
            </form>
</center>

{% if context %}
    <center>
        <p>Гарантированный доход пакета проектов составляет: {{context['data'][0]}} д.е.</p>
        <p>Ожидаемый доход: {{context['data'][1]}} д.е.</p>
        <p>Дисперсия дохода: {{ context['data'][2]}} д.е.</p>
        <div id="container" style="width: 700px; height: 400px;"></div>
        <!--<canvas id="container" width="600" height="400"></canvas>-->
        <canvas id="Chart1" width="600" height="400"> </canvas>
    </center>
            <script>
            Chart.defaults.global.responsive = false;
            var chartData = {
                title: 'Инвестиции и ресурсы предприятия',
                labels : [{% for item in context['labels'] %}
"{{item}}",
                            {% endfor %}],
                datasets : [{label: '{{ context['legend'][0] }}',
                  fill: true,
                  lineTension: 0.1,
                  backgroundColor: "rgba(75,192,192,0.4)",
```

```
  borderColor: "rgba(75,192,192,1)",
  borderCapStyle: 'butt',
  borderDash: [],
  borderDashOffset: 0.0,
  borderJoinStyle: 'miter',
  pointBorderColor: "rgba(75,192,192,1)",
  pointBackgroundColor: "#fff",
  pointBorderWidth: 1,
  pointHoverRadius: 5,
  pointHoverBackgroundColor: "rgba(75,192,192,1)",
  pointHoverBorderColor: "rgba(220,220,220,1)",
  pointHoverBorderWidth: 2,
  pointRadius: 1,
  pointHitRadius: 10,
  data : [{% for item in context['values'][0] %}
            {{item}},
          {% endfor %}],
  spanGaps: false
},
{label: '{{ context['legend'][1] }}',
  fill: true,
  lineTension: 0.1,
  backgroundColor: "rgba(125,150,192,0.4)",
  borderColor: "rgba(125,150,192,1)",
  borderCapStyle: 'butt',
  borderDash: [],
  borderDashOffset: 0.0,
  borderJoinStyle: 'miter',
  pointBorderColor: "rgba(125,192,192,1)",
  pointBackgroundColor: "#fff",
  pointBorderWidth: 1,
  pointHoverRadius: 5,
  pointHoverBackgroundColor: "rgba(125,150,192,1)",
  pointHoverBorderColor: "rgba(220,220,220,1)",
  pointHoverBorderWidth: 2,
  pointRadius: 1,
  pointHitRadius: 10,
  data : [{% for item in context['values'][1] %}
            {{item}},
          {% endfor %}],
  spanGaps: false
},
{label: '{{ context['legend'][2] }}',
  fill: true,
  lineTension: 0.1,
  backgroundColor: "rgba(175,75,150,0.4)",
  borderColor: "rgba(175,75,150,1)",
  borderCapStyle: 'butt',
  borderDash: [],
  borderDashOffset: 0.0,
  borderJoinStyle: 'miter',
  pointBorderColor: "rgba(175,75,150,1)",
  pointBackgroundColor: "#fff",
  pointBorderWidth: 1,
  pointHoverRadius: 5,
  pointHoverBackgroundColor: "rgba(175,75,150,1)",
```

```
                pointHoverBorderColor: "rgba(220,220,220,1)",
                pointHoverBorderWidth: 2,
                pointRadius: 1,
                pointHitRadius: 10,
                data : [{% for item in context['values'][2] %}
                          {{item}},
                        {% endfor %}],
                spanGaps: false
            }
            ]
            }
        var                ctx                =                docu-
ment.getElementById("Chart1").getContext("2d");
        var Chart1 = new Chart(ctx, {
        type: 'line',
        data: chartData,
        options: {
            title: {text: "Инвестиции и ресурсы предприятия", dis-
play: true},
            legend: {hidden: true, position: "bottom"},
            scales: {
                yAxes: [{scaleLabel: {display: true, labelString:
"Ресурсы, д.е."}}],
                xAxes: [{scaleLabel: {display: true, labelString:
"Временной период"}}],
                },
            tooltips: {
                enabled: true,
                mode: 'single',
                callbacks: { label: function(tooltipItems, data)
{return tooltipItems.yLabel + ' д.е.';}}
        }}});

        var holder = document.getElementById("Chart1");
        var            pointSelected            =            docu-
ment.getElementById("pointSelected");
        holder.onclick = function(evt){
            var activePoint = Chart1.getElementAtEvent(evt);
            pointSelected.innerHTML = 'Point selected... index: '
+ activePoint[0]._index;
        };


        var data1 = [{% for item in context['rangebar'] %}
                    { low: {{item["low"]}}, high: {{item["high"]}}
},
                {% endfor %}
            ];

        anychart.onDocumentLoad(function() {
                chart = anychart.barChart();
                chart.title().text('Распределение проектов по вре-
мени');
                chart.xAxis().title("Номер проекта");
                chart.yAxis().title("Период");
                chart.rangeBar(data1).hoverStroke("darkred", 4);
```

```
                    chart.container('container'); // pass the contain-
er id, chart will be displayed there
                    chart.draw(); // call the chart draw() method to
initiate chart display
                });
            </script>
{% endif %}
{% endblock %}
```