

# РЕФЕРАТ

Дипломная работа: 57 с., 0 рис., 94 табл., 0 источников. Ключевые слова: ГЕНЕТИЧЕСКИЕ АЛГОРИТМЫ, МАШИННОЕ ОБУЧЕНИЕ, NP-ПОЛНЫЕ ЗАДАЧИ, ЗАДАЧА О РЮКЗАКЕ, ЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ, МНОГОМЕРНЫЙ РЮКЗАК  
Объект исследования - задача о многомерном рюкзаке  
Цель работы - исследование применения генетического алгоритма к задаче о многомерном рюкзаке. Результатом работы является программа, реализующая генетический алгоритм на языке C# и его модификации. Разработанная программа позволяет найти точные максимумы для малых наборов предметов(<50 предметов) в 100% случаев и для больших(100 предметов) - в 43,3% случаев. В остальных случаях программа находит локальный максимум, различающий с исходным не более чем на 0,21%.

# Содержание

<b>ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ</b>	<b>4</b>
<b>ВВЕДЕНИЕ</b>	<b>5</b>
<b>1 Начало</b>	<b>6</b>
1.1 Структура текста .....	6
1.2 Символы .....	6
1.3 Команды Latex .....	7
1.4 Пробелы, строки и абзацы .....	7
1.5 Промежутки между словами и вертикальные отступы .....	8
<b>2 Управление шрифтом</b>	<b>9</b>
2.1 Команды переключения формы шрифта .....	9
2.2 Размеры шрифта .....	9
2.3 Расстояние между строками - интерлиньяж .....	10
<b>3 Рубрикация и оглавление</b>	<b>11</b>
3.1 Команды рубрикации .....	11
3.2 Оглавление .....	11
3.3 Перекрестные ссылки .....	11
3.4 Приложения .....	12
3.5 Ненумерованные заголовки .....	12
<b>4 Список литературы и ссылки на него</b>	<b>13</b>
4.1 Накопить библиографию .....	13
4.2 Файл с библиографией .....	14
4.3 Как формируется список .....	14
4.4 Ссылки .....	14
4.5 Англоязычные документы .....	15
<b>5 Списки, сноски, дословная печать</b>	<b>16</b>
5.1 Списки .....	16
5.1.1 Простые перечисления .....	16
5.1.2 Перечисления с нумерацией .....	16
5.1.3 Перечни без пустот .....	17

5.2	Сноски .....	17
5.3	Гиперссылки .....	18
5.4	Воспроизведение «как есть» .....	18
<b>6</b>	<b>Таблицы</b>	<b>19</b>
6.1	Описание таблицы .....	19
6.2	Дополнительные возможности .....	19
6.3	Плавающие объекты .....	20
6.4	Не плавающие таблицы .....	21
<b>7</b>	<b>Рисунки</b>	<b>23</b>
7.1	Описание рисунка .....	23
7.2	Плавающие рисунки .....	23
7.3	Не плавающие рисунки .....	24
<b>8</b>	<b>Формулы</b>	<b>25</b>
<b>9</b>	<b>Теоретическая часть</b>	<b>26</b>
9.1	Задача о многомерном рюкзаке .....	26
9.2	Генетические алгоритмы .....	26
9.2.1	Этапы работы алгоритма .....	27
9.2.2	Выбор этапов .....	28
9.2.3	Дополнения алгоритма .....	29
<b>10</b>	<b>Практическая часть</b>	<b>30</b>
10.1	Наборы тестов .....	30
10.2	Первый набор тестов .....	31
10.2.1	Мутация в одной позиции .....	31
10.2.2	Инверсионная мутация .....	32
10.2.3	Промежуточные выводы .....	33
10.3	Второй набор тестов .....	33
	<b>ЗАКЛЮЧЕНИЕ</b>	<b>36</b>
	<b>ПРИЛОЖЕНИЯ</b>	<b>38</b>
<b>A</b>	<b>Исходный код программы</b>	<b>39</b>

# ОБОЗНАЧЕНИЯ

В данной работе использованы следующие обозначения и сокращения.

Класс проблем NP (nondeterministic polynomial) – множество проблем разрешимости, решение которых возможно проверить на машине Тьюринга за время, не превосходящее полинома от размера входных данных .

NP-полная проблема – проблема из класса NP, к которой можно свести любую другую проблему из этого класса за полиномиальное время.

# ВВЕДЕНИЕ

В ходе развития компьютерных наук человечеством встретилось с классом NP-полных задач. Такие задачи решаются алгоритмически за недетерминированное полиномиальное время, что существенно затрудняет поиск решения таких задач в приемлемые сроки. Одной из таких задач является задача о рюкзаке (Knapsack problem) и её модификации. Объектом исследования данной работы является задача о многомерном рюкзаке (Multidimensional knapsack problem). Предмет исследования - решение задачи о многомерном рюкзаке с использованием генетического алгоритма. Многие прикладные проблемы могут быть формализованы в виде рассматриваемой задачи. Примерами таких проблем являются размещение процессоров и баз данных в системе распределенных вычислений (см. [1]), погрузка груза и контроль бюджета (см. [2]), задачи раскройки (см. [3]) и др. Решение этих проблем обуславливает актуальность решения задачи о рюкзаке.

Цель исследования - решить задачу о многомерном рюкзаке с использованием генетического алгоритма. Для достижения цели были поставлены следующие задачи

- Исследовать генетические алгоритмы в применении к NP-полным задачам
- Спроектировать и реализовать генетический алгоритм, решающий задачу о многомерном рюкзаке
- Провести оценку эффективности генетического алгоритма в решении поставленной задачи, используя набор готовых тестов

# 1 Начало

Создание выпускной работы в Latex - это подготовка текста, в котором встречаются команды Latex. Во многом в этом помогают предоставляемые шаблоны, например, готовая верстка титульной страницы, в которой нужно только заменить слова. Также помогают примеры, которые позволяют выполнять работу по аналогии или также, заменяя отдельные фрагменты.

Пример текста (с командами курсива и полужирный):

У попа {\it была} {\bf собака}

У попа *была* **собака**

## 1.1 Структура текста

Исходный файл для Latex состоит из *преамбулы* и *основного текста*. Преамбула, располагающаяся в начале текста, задает класс документа, используемые пакеты и прочие технические параметры [\_\_2010].

После преамбулы идет строка `\begin{document}`, с которой и начинается основной текст. В самом конце файла нужно поместить строку

`\end{document}`

Пример (от знака % идет комментарий):

`\documentclass[a4paper,14pt]{article}`

% здесь преамбула

`\begin{document}`

Привет мир

`\end{document}`

## 1.2 Символы

Большинство символов в основном тексте обозначают сами себя, набирайте их прямо с клавиатуры. Спецсимволы говорят Latex что-то специальное:

{ } \$ & # % \_ ^ ~ \

Первые семь печатаются с помощью односимвольной команды, впереди добавляется \, например, % печатается командой \%.  
Для \ команда `\textbackslash`.

Символы ^ и ~ можно напечатать, например, с помощью команды `\verb`, смотрите

раздел 5.4.

Существует множество команд, которые печатают небуквенные символы, смотрите, например, [\_\_2003].

## 1.3 Команды Latex

Некоторые команды требуют обязательные параметры, например, `\section{Название раздела}`. У некоторых команд могут быть необязательные параметры, они заключаются в квадратные скобки:

`\documentclass[a4paper, 14pt]{article}`, здесь `a4paper` - размер бумаги, `14pt` - нормальный размер шрифта 14. Название команды Latex может быть одним небуквенным символом или состоять из нескольких букв. Во втором случае команда без параметров ограничивается справа пробелом, который при этом не печатается. Если это неудобно, то конец команды можно указать пустыми фигурными скобками `}`.

Кроме команд в языке Latex используются *окружения* - фрагменты текста, заключенные между командами `\begin{}` и `\end{}`. Обе команды в качестве параметра принимают *имя окружения*. Окружение позволяет распространить свойство или действие на весь заключенный в окружение текст. Например, окружение `large` позволяет напечатать фрагмент текста увеличенным шрифтом. Для этого фрагмент заключается между `\begin{large}` и `\end{large}` [\_\_2010].

## 1.4 Пробелы, строки и абзацы

Любое количество идущих подряд *пробельных символов*, то есть пробелов, табуляций и переводов строки, Latex воспринимает как один пробел. Размер пробела между словами Latex выбирает сам, причем так, чтобы все пробелы в одной строке были одинаковы.

Однако пустая строка, то есть два подряд символа перевода строки, обозначает *разделитель абзацев*. Команда `\noindent`, вставленная непосредственно перед абзацем, отменяет отступ (красную строку) для этого абзаца. Комбинация из двух символов `\\` принудительно завершает строку, но не абзац. Команда `\par` означает то же, что пустая строка, то есть отделяет один абзац от другого.

Вставить между абзацами вертикальный интервал можно, например, командами `\bigskip`, `\medskip` и `\smallskip` [\_\_2010].

## 1.5 Промежутки между словами и вертикальные отступы

Некоторые слова нежелательно разрывать на разные строки, тогда между ними нужно ставить неразрывный пробел `\,~`. Пример: на странице~5.

Команда `\quad` дает промежуток в `1em`, команда `\qquad` - промежуток в `2em` (`em` - промежуток, равный примерно ширине латинской буквы `M` текущего шрифта).

Для небольшого пробела используется команда `\,`,

Если необходимо задать промежуток с указанием конкретной длины, можно воспользоваться командой

`\hspace{длина}`, например, `\hspace{1.5cm}` [**2003**]. Если этот промежуток также должен сохраняться и в начале ( или конце ) строки, то вместо команды `\hspace` применяется `\hspace*`.

Вертикальные отступы создаются командой `\vskip`, например, `\vskip3.5cm`.



## 2 Управление шрифтом

### 2.1 Команды переключения формы шрифта

Форма шрифта характеризуется семейством (с засечками, без засечек, как у пишущей машинки или моноширинный), насыщенность (средняя, полужирная), начертание (прямое, курсивное, наклонное, capital). Имеются команды, в которых текст является параметром, внутри действия такой команды можно применять другие команды управления формой шрифта. Пример:

собака **Барсик** и кошка **Муся**

Таблица 1: Форма шрифта

ru	en	команда	результат
с засечками	roman	<code>\textrm{Слова}</code>	Слова
без засечек	sans serif	<code>\textsf{Слова}</code>	<b>Слова</b>
пишущая машинка	typewriter	<code>\texttt{Слова}</code>	Слова
средняя насыщенность	medium	<code>\textmd{Слова}</code>	Слова
полужирная	boldface	<code>\textbf{Слова}</code>	<b>Слова</b>
прямое начертание	upright shape	<code>\textup{Слова}</code>	Слова
курсивное	italic	<code>\textit{Слова}</code>	<i>Слова</i>
наклонное	slanted	<code>\textsl{Слова}</code>	<i>Слова</i>
заглавные буквы	capital	<code>\textsc{Слова}</code>	СЛОВА

### 2.2 Размеры шрифта

Размер шрифта отсчитывается от нормального размера. В данном шаблоне нормальный размер, применяемый по умолчанию, 14pt. Из приводимых ниже команд ему соответствует `\normalsize`. Другие команды по определенному алгоритму уменьшают или увеличивают размер. Эти команды могут применяться внутри группы текста, ограниченного фигурными скобками, тогда их действие за пределы скобок не распространяется. Пример :

`{\small Путь}` на `{\large верх}`

Путь на верх.

Таблица 2: Размеры шрифта

<code>\tiny</code>	крошечный
<code>\scriptsize</code>	очень маленький
<code>\footnotesize</code>	довольно маленький
<code>\small</code>	маленький
<code>\normalsize</code>	нормальный
<code>\large</code>	большой
<code>\Large</code>	еще больше
<code>\LARGE</code>	очень большой
<code>\huge</code>	огромный
<code>\Huge</code>	громадный

## 2.3 Расстояние между строками - интерлиньяж

При смене размера шрифта ( непосредственно в том месте, где вставлена соответствующая команда ) определяется только размер символов. Что же касается расстояния между строками, то оно определяется в тот момент, когда очередная строка завершается ( либо по принудительной команде, либо, при верстке целого абзаца, по его окончании ). Возможна такая ситуация, когда вы начали писать абзац крупным шрифтом, а закончили мелким, что первые строки абзаца будут на очень маленьком расстоянии друг от друга. Самый простой способ, чтобы так не происходило, в одном абзаце использовать один размер.

## 3 Рубрикация и оглавление

### 3.1 Команды рубрикации

Для класса `book`, назначенного в этом шаблоне, внутри основной части можно использовать `\chapter` - глава, `\section` - раздел, `\subsection` - подраздел, `\subsubsection` - подподраздел, `\paragraph` - параграф, `\subparagraph` - подпараграф. Эти команды имеют параметр - заголовок соответствующего раздела.

Пример:

```
\section {Рубрикация и оглавление}. Главы, разделы и подразделы авто-  
матически нумеруются и помещаются в оглавление.
```

Если желательно избежать переноса слов в заголовке соответствующего раздела, то можно явно указать перенос на новую строку. При этом можно разрешить перенос слов в оглавлении, если в квадратных скобках привести текст для оглавления. Пример:

```
\chapter[Рубрикация и оглавление]  
{Рубрикация\\ и оглавление}
```

### 3.2 Оглавление

Сформировать оглавление в `Latex` очень просто: достаточно в том месте, где вы хотите увидеть оглавление, вставить команду

```
\tableofcontents
```

### 3.3 Перекрестные ссылки

`Latex` имеет команды, чтобы сослаться на раздел, страницу и другие части текста. Для этого место, на которое нужно сослаться, помечается командой `\label`. В качестве параметра указывается метка, состоящая из латинских букв, цифр, символов двоеточия, подчеркивания и тире. Пример:

```
\section {Рубрикация и оглавление}\label{toc}
```

... Это было рассмотрено в разделе `\ref{toc}` на стр. `\pageref{toc}`  
на печати:

Это было рассмотрено в разделе 3 на стр. 11

## 3.4 Приложения

Если в работе предусмотрены приложения, нужно поставить перед ними команду `\appendix`. Эта команда дается только один раз, чтобы отделить текст всех приложений от основного текста. После команды `\appendix` каждое приложение начинается с команды `\chapter{ }`, но вместо слова «Глава» выводится слово «Приложение», а вместо номера арабскими цифрами выводится номер в виде заглавной латинской буквы.

## 3.5 Ненумерованные заголовки

Команда `\section*{Заголовок}` напечатает заголовок в том же формате, что и обычные заголовки по команде `\section{ }`, но нумероваться он не будет. Такое же воздействие `*` оказывает на другие команды: `\chapter*{ }` и т.д.

Ненумерованные заголовки автоматически не попадают в оглавление, нужно сделать так:

```
\chapter*{ОСНОВНАЯ ЧАСТЬ}
```

```
\addcontentsline{toc}{chapter}{ОСНОВНАЯ ЧАСТЬ}
```

команда `\addcontentsline` поместит заданный текст в оглавление в формате главы. Таким же образом в оглавление помещается заголовок над библиографией.

## 4 Список литературы и ссылки на него

Ниже описано, как подготовить библиографию, чтобы Latex автоматически создал список использованной литературы в соответствии с ГОСТ.

### 4.1 Накопить библиографию

Установите Zotero под Firefox, как описано в приложении ???. Большая буква Z справа вверху на панели Firefox свидетельствует о том, что Zotero установлено. Кликните мышкой на эту букву Z. В нижней половине экрана откроется Zotero.

Создадим новую коллекцию для вашей библиографии. Экран Zotero обычно делится на 3 колонки. В левой колонке вверху подведите правую мышку к заголовку «Моя библиотека». В появившемся меню выберите «Новая подборка» и введите название, например, «bibliogr».

Заполним созданную подборку необходимой библиографией. Откроем в Firefox Scopus, Web of science, РИНЦ или электронный каталог Российской национальной библиотеки.

В нижней половине экрана у вас открыто Zotero. Подборка, где будет собираться библиография, выделена мышкой.

Проведите поиск. Справа от большой буквы Z справа вверху на панели Firefox появится иконка: желтая, если найдено несколько документов, или для одного документа - с маленьким изображением титульного листа. Нажмите на эту иконку. Если потребуется, отметьте мышкой нужные документы и подтвердите. В средней колонке Zotero начнут появляться документы. Выделите мышкой один из них. В правой колонке в закладке «Информация» вы увидите значения всех библиографических полей.

Созданные давно статьи и монографии могут отсутствовать в современных каталогах. Кроме того каталоги иногда содержат ошибки. В обоих случаях необходимо уметь вводить и редактировать библиографическую запись вручную. Чтобы отредактировать запись, выделите в средней колонке документ. В правой колонке в закладке «Информация» появятся поля этой записи. Любое из них можно редактировать. По поводу автора: из выпадающего списка можно выбрать роль - редактор, автор и т.д., имеется два подполя - для фамилии и инициалов, для второго и т.д. автора - нажать на + справа.

## 4.2 Файл с библиографией

Подготовим файл с библиографией в формате biblatex и с расширением .bib.

Откроем Firefox, откроем Zotero в нижней половине экрана, нажав на большую Z, в левой колонке найдем подборку, в которой накоплена библиография для выпускной работы, и подведем к ней правую мышку.

Выберем «Экспортировать подборку».

В выпадающем списке выберем Формат = Biblatex. Нужно убрать, если есть, отметки в строчках Экспорт заметок, Экспорт файлов, Использовать сокращенное название журнала. И жмем ОК. Сохраняем файл с именем из латинских букв в папку, где верстаем выпускную работу.

## 4.3 Как формируется список

В преамбуле вызывается пакет biblatex.

В команде `\addbibresource{bibliogr.bib}` Latex получает информацию, что библиография находится в файле `bibliogr.bib`.

В конце текста, но перед приложениями команда

```
\printbibliography
```

```
[title=Список \использованных источников]
```

печатает список библиографии.

Команда

```
\addcontentsline{toc}{chapter}
```

```
{Список использованных источников}
```

включает список в оглавление.

Команда `\nocite{*}` говорит о том, что нужно выдать в списке использованной литературы не только документы, на которые есть ссылки из текста выпускной работы, но все содержимое файла с библиографическими записями. По окончании отладки текста эту команду лучше убрать.

Все это присутствует в шаблоне и вам остается аккуратно использовать шаблон в верстке выпускной работы. Нужно только в команде

`\addbibresource{bibliogr.bib}` заменить имя файла на используемое вами, когда вы экспортировали библиографию из Zotero.

## 4.4 Ссылки

В тексте можно сослаться на документ из списка использованной литературы с помощью метки библиографической записи. Метки формируются Zotero автоматически, они находятся в первой строке описания библиографической записи. Например, в файле `bibliogr.bib`, прилагаемом к данному шаблону, монография А.В.Столярова снабжена меткой `___2010`. Сослаться на эту монографию можно, употребив в тек-

сте команду `\cite{__2010}`. В квадратных скобках появится номер документа, как он напечатан в списке использованной литературы. Если нужно сослаться на страницу 10, то команда будет такой: `\cite[][с.~10]{__2010}`. Текст из вторых квадратных скобок появится после номера документа. Если нужно поместить какой-то текст до номера документа, то его нужно разместить внутри первых квадратных скобок, иначе эти скобки пустые.

## 4.5 Англоязычные документы

Англоязычные документы в списке использованной литературы представляются библиографической записью на английском языке. В этом случае необходимо стандартные слова «страницы», «том», «редактор» и т.д. представить на английском.

В данном шаблоне Latex по умолчанию считает библиографическую запись составленной по-русски. Если есть записи на английском языке, то нужно отредактировать файл `.bib` с библиографией. В такие записи нужно вставить поле `hyphenation={english}`.

Смотреть и редактировать файлы с расширением `.bib` удобно в бесплатном редакторе `notepad++`. Где его скачать легко найти в [GOOGLE](#).

## 5 Списки, сноски, гиперссылки, дословная печать

### 5.1 Списки

Иногда возникает необходимость перечислить объекты, методы и т.д. Списки можно вкладывать друг в друга.

#### 5.1.1 Простые перечисления

Зарубежные новостные агентства:

```
\begin{itemize}
\item REUTERS
\item CNN
\item BBC
\end{itemize}
```

на печати:

Зарубежные новостные агентства:

- REUTERS
- CNN
- BBC

#### 5.1.2 Перечисления с нумерацией

Стихотворные размеры:

```
\begin{enumerate}
\item Хорей
\item Ямб
\end{enumerate}
```



на печати:

Стихотворные размеры:

1. Хорей

2. Ямб

### 5.1.3 Перечни без пустот

( работает в данном шаблоне )

Постановка научной проблемы:

```
\begin{compactlist}
```

```
\item обнаружение недостаточности знаний, противоречий
```

```
\item осознание потребности
```

```
\item формулирование проблемы
```

```
\end{compactlist}
```

на печати:

Постановка научной проблемы:

- обнаружение недостаточности знаний, противоречий
- осознание потребности
- формулирование проблемы

## 5.2 Сноски

Сноски оформляются с помощью команды `\footnote{<текст>}`. В том месте, где команда встретилась, появляется номер сноски, набранный цифрами уменьшенного размера и сдвинутый вверх. Текст сноски, снабженный тем же номером, размещается в нижней части страницы. От основного текста сноски отделяются горизонтальной чертой [\_\_2010].

В рамках каждой главы сноски имеют сквозную нумерацию; при смене главы (то есть по команде `\chapter`) счетчик сносок сбрасывается, так что первая сноска каждой главы имеет номер 1.

Иногда возникает желание сослаться на одну и ту же сноску несколько раз. Для этого первое вхождение сноски следует оформить как обычно и снабдить меткой, например, так:

```
\label{my_footnote}.
```

А в остальных местах применить команду

```
\footnotemark[\ref{my_footnote}].
```

## 5.3 Гиперссылки

www-адрес, как ссылка: `\url{http://www.rsl.ru/}`  
<http://www.rsl.ru/>

Сайт: `\href{http://urfu.ru/}{УРФУ}`  
Сайт: [УРФУ](http://urfu.ru/).

## 5.4 Воспроизведение «как есть»

В тексте или специальных листингах, например, в программах могут встречаться слова, которые воспринимаются Latex, как команды. Для этих случаев есть инструмент, который воспроизводит текст «как есть».

Начиная с команды `\begin{verbatim}` и до тех пор, пока не встретится строка `\end{verbatim}` Latex все встреченные символы будет выдавать моноширинным шрифтом, причем никакие символы ( кроме комбинации `\end{verbatim}` ) не будут интерпретироваться никаким специальным образом.

Для фрагментов внутри строки применяется команда `\verb`. Выдаваемый текст с обеих сторон ограничивается одним и тем же символом, например, вертикальной чертой, и прижимается вплотную к `\verb` справа без пробела.

## 6 Таблицы

### 6.1 Описание таблицы

Таблица создается окружением `tabular`. Пример:

мужские имена	женские имена
Александр Роман	Анастасия Рената

```
\begin{tabular}{|c|c|}  
\hline  
мужские имена&женские имена\\  
\hline  
Александр&Анастасия\\  
Роман&Рената\\  
\hline  
\end{tabular}
```

В строке `\begin{tabular}` количество букв «с» - это количество столбцов формируемой таблицы.

Буква «с» говорит о выравнивании по центру в соответствующем ей по порядку расположения столбце. Другие возможные буквы вместо с: l - выравнивание влево, r - выравнивание вправо.

Вертикальные черточки по краям и между буквами порождают вертикальные линии на всю высоту таблицы.

`\hline` порождает горизонтальную линию между строками таблицы.

Строки таблицы завершаются командой `\\`. Между ячейками одной строки ставится `&`. Количество ячеек в каждой строке должно соответствовать количеству столбцов, заданному в строке `\begin{tabular}`.

### 6.2 Дополнительные возможности

Пример с ячейкой на несколько столбцов [\_\_2010]:

№	Тип	Автор	Заглавие	Год	Тир.
1	книга	Артур Конан Дойл	Собака Баскервильей	1975	10 000
2	книга	Жюль Верн	Пять недель на воздушном шаре	1981	7000
3	журнал	Вокруг света (№ 5)		1995	5000

```

\begin{tabular}{|r|c|p{0.15\textwidth}|
p{0.25\textwidth}|c|r|}
\hline
№ & Тип & Автор & Заглавие & Год & Тир. \\
\hline
1 & книга & Артур Конан Дойл & Собака Баскервильей & 1975 & 10\,000 \\
\hline
2 & книга & Жюль Верн & Пять недель на воздушном шаре & 1981 & 7000 \\
\hline
3 & журнал & \multicolumn{2}{c|}{Вокруг света (№ 5)} & 1995 & 5000 \\
\hline
\end{tabular}

```

Ячейка на несколько столбцов организуется командой `\multicolumn`.

Первый параметр - количество столбцов, на которые распространяется ячейка.

Второй параметр - буква (с, l или r), задающая выравнивание в этой ячейке, кроме того, может потребоваться обозначить вертикальными черточками вертикальные границы ячейки - одну или обе.

Третий параметр - текст, помещаемый в ячейку.

В этом же примере есть столбцы, в которых текст в ячейках занимает несколько строк. Для этих столбцов вместо буквы, задающей выравнивание, нужно указать размер ширины столбца в формате `p{ширина}`. Примеры:

`p{3.5cm}` - 3,5 сантиметра,

`p{50mm}` - 50 миллиметров,

`p{0.25\textwidth}` - четверть от ширины документа.

## 6.3 Плавающие объекты

Когда в документ требуется вставить рисунок или таблицу, обычно в тексте ставится ссылка ( что-то вроде «см. рис. 2» или «см. табл. 8» ), а сами таблицы или рисунки размещаются вверху или внизу текущей страницы. Это вызвано тем, что жесткая привязанность к тексту приводит к тому, что остаются незаполненные пустые места.

Latex позволяет описывать объекты, которые следует поместить вблизи места, где вставлено их описание, но конкретное размещение можно выбрать, исходя из удобства и эстетичности. Такие объекты называются плавающими [\_\_2010].

Описание плавающей таблицы 3:

```
\begin{table}[htbp]
\caption{Пример}\label{tab}
\begin{tabular}{cc}
A&B\\
C&D\\
\end{tabular}
\end{table}
```

Таблица 3: Пример

A	B
C	D

Окружение

```
\begin{table}[htbp]
...
end{table}
```

сообщает Latex, что внутри него находится описание плавающей таблицы. В квадратных скобках можно дать рекомендации по размещению в порядке убывания желательности:

h - (here) сразу после появления в исходном тексте,

t - (top) вверху страницы,

b - (bottom) внизу страницы,

p - (page) на отдельной странице.

Пожелание можно превратить в требование, добавив восклицательный знак. Latex в этом случае сделает все, что сможет, но требуемое размещение не гарантирует.

Команда `\caption{Пример}\label{tab}` обеспечивает заголовок таблицы, нумерацию и вывод номера. Если `caption` пометить (командой `\label{tab}`), то на таблицу можно сослаться с помощью использованной метки, например, так:

...данные размещены в таблице `\ref{tab}`

...данные размещены в таблице 3.

Можно использовать не плавающие объекты попеременно с плавающими, при этом нумерация будет правильной.

## 6.4 Не плавающие таблицы

Описание не плавающей таблицы 4:

```

{
\captionof{table}{Пример}\label{tabex}
\begin{tabular}{cc}
A&B\\
C&D\\
\end{tabular}
}

```

Обратите внимание, весь комплект одет в фигурные скобки. Если надпись над таблицей и сама таблица оказались на разных страницах, то весь комплект оденьте в окружение

```

\begin{minipage}{\textwidth}
\end{minipage}.

```

Таблица 4: Пример

A	B
C	D

Команда `\captionof{table}{Пример}` обеспечивает заголовок таблицы, нумерацию и вывод номера. Если `captionof` пометить командой

`\label{tabex}`, то на таблицу можно сослаться с помощью использованной метки, например, так:

...данные размещены в таблице `\ref{tabex}`

...данные размещены в таблице 4.

## 7 Рисунки

### 7.1 Описание рисунка

Рисунки из файлов вставляются командой `\includegraphics`.

Примеры.

`\includegraphics[width=0.9\textwidth]{krug.jpeg}` - рисунок займет 90% полосы набора;

`\includegraphics[scale=0.75]{clinical.pdf}` - размеры рисунка умножатся на коэффициент 0.75.

Расширение файла можно опускать.

Допускаются векторные форматы `.pdf` и `.eps`, растровые форматы `.jpg` и `.png`. Векторные форматы предпочтительнее.

Рисунки можно собрать в отдельную папку и указать в команде абсолютный или относительный путь к файлу, например, так:

```
\includegraphics{images/train.eps}
```

### 7.2 Плавающие рисунки

Рисунок становится плавающим с помощью окружения `\begin{figure}`.

Пример описания рисунка ??:

```
\begin{figure}[htbp]
\includegraphics[width=0.7\textwidth]{zclinical.pdf}
\caption{Приборы для медицины}
\label{fig_clinics}
\end{figure}
```

Поведение плавающих объектов и опции `[htbp]` - описаны в разделе 6.3. Подпись под рисунком, нумерация и вывод номера обеспечивается командой `\caption`. Плавающие рисунки могут использоваться попеременно с неплавающими, при этом нумерация будет общая.

На рисунок можно сослаться с помощью метки, использованной в команде `\label`.  
...измерения проведены на приборе ...см. рис. `\ref{fig_clinics}`  
...измерения проведены на приборе ...см. рис. ??

## 7.3 Не плавающие рисунки

Обратите внимание, что весь комплект одет в фигурные скобки. Если рисунок и подпись под ним оказались на разных страницах, то весь комплект оденьте в окружение

```
\begin{minipage}{\textwidth}  
\end{minipage}.
```

Команда `\captionof{figure}` обеспечивает подпись под рисунком, нумерацию и вывод номера на печать.

Команда `\label{fig_krug}` вставляет метку, благодаря чему можно сослаться на рисунок:

...смотрите рисунок `\ref{fig_krug}`

...смотрите рисунок ??



## 8 Формулы

Верстка формул блестяще описана в [\_\_2003] и [\_\_2010]. Если ваша статья содержит формулы, рекомендуем ознакомиться с этими материалами.

Пример описания формулы [8.1](#):

```
\begin{equation}
\label{eq}
\sum_{i=1}^n n^2=\frac{n(n+1)(2n+1)}{6}
\end{equation}
```

$$\sum_{i=1}^n n^2 = \frac{n(n+1)(2n+1)}{6} \quad (8.1)$$

## 9 Теоретическая часть

### 9.1 Задача о многомерном рюкзаке

В данной работе рассматривается задача о многомерном рюкзаке (Multidimensional 0-1 knapsack problem, МКР). Эта задача является модификацией классической задачи о рюкзаке, поставленной в 19 веке Джорджем Мэттьюсоном. (см. [4]) Данный же вариант задачи впервые был предложен Клиффордом Петерсеном в 1967 году. (см. [5])

Постановка задач такова

Пусть существует  $N$  предметов, каждый из которых имеет стоимость  $c_i$  и размеры  $s_{ij}$ , где  $i \in 1, 2, \dots, N$ ,  $j \in 1, 2, \dots, M$ . Пусть также существует рюкзак с ограничениями по вместимости по измерениям  $r_j$ . Требуется максимизировать сумму

$$\sum_{i=1}^N c_i x_i$$

где  $x_i \in \{0, 1\}$  при условии

$$\sum_{i=1}^N s_{ij} x_i < r_j \forall j \in \{1, 2, \dots, M\} \quad (9.1)$$

И стандартная задача, и её модификация являются NP-полными задачами. Вычислительная сложность задачи такого рода при переборном решении для  $N$  предметов -  $O(2^M N^3)$ , что, вкупе с NP-сложностью, делает алгоритмическое решение такой задачи неэффективным для больших  $N$ . Однако такие задачи могут быть решены эвристическими алгоритмами, то есть алгоритмами, для которых их корректность строго не доказана.

### 9.2 Генетические алгоритмы

Генетические алгоритмы являются семейством в множестве эвристических алгоритмов. Впервые такой алгоритм был предложен А. Фразером. (см [6]) Алгоритм является итеративным. Генетический алгоритм моделирует естественные процессы эволюции популяции, а именно - мутацию и скрещивание. Решение задачи с помощью такого алгоритма требует нескольких предварительных этапов:

- **Выбор кодирования генотипа.**  
На этом этапе нужно выбрать способ кодирования генотипа, который будет эффективен для данной задачи. Такой генотип должен однозначно моделировать сущность, рассматриваемую в задаче.
- **Выбор начального приближения.**  
Для запуска итерационного процесса требуется создать начальное множество - пул генотипов. В зависимости от способа задания начального пула скорость поиска оптимального решения может меняться. Размер пула также оказывает влияние на эффективность алгоритма: слишком маленький пул подавляет разнообразие и приводит к попаданию в локальные максимумы, слишком большой - увеличивает число операций на каждой итерации, чем замедляет работу алгоритма.
- **Выбор мутации.**  
На каждой итерации алгоритма заранее определенная часть пула генотипов подвергнется мутациям, то есть определенным образом изменятся их составляющие. Если мутирует слишком малая часть пула, то генетический алгоритм не сможет покинуть локальные максимумы, если же слишком большая - алгоритм потеряет свойство сохранения признаков.
- **Выбор механизма скрещивания (кроссинговера).**  
После мутации происходит создание новых генотипов из частей старых с сохранением признаков родителя. Алгоритм скрещивания позволяет получить из двух родительских генотипов два различных дочерних генотипа.
- **Выбор функции оценки(фитнесс-функции).**  
Такая функция позволяет оценивать генотипы с точки зрения их близости к оптимальному решению и отбирать из них лучшие на каждой итерации.

### 9.2.1 Этапы работы алгоритма

- Создается пул генотипов с использованием заданного алгоритма начального приближения
- Запускается итерационный процесс
  - Случайным образом выбирается часть пула, которая подвергнется мутации
  - Выбранная часть пула генотипов мутируется с использованием заданного алгоритма мутации
  - Мутировавшие генотипы замещают собой исходные в пуле, немутировавшие остаются без изменений
  - Из пула генотипов выбираются пары для скрещивания
  - Производится скрещивание с использованием заданного алгоритма

С использованием заданной функции оценки из результатов скрещивания выбираются лучшие

Если выполнено условие останова - например, достигнут предел числа итераций или известный максимум, то итерационный процесс завершается, в противном случае начинается следующая итерация.

- Результат итерационного процесса отдается пользователю

## 9.2.2 Выбор этапов

Наиболее естественным кодированием отдельного решения задачи о рюкзаке в генотип является бинарная последовательность длины  $N$ , состоящая из нулей и единиц. Каждый  $i$ -й элемент такой последовательности является индикатором вхождения  $i$ -го предмета в текущее решение. Такая модель требует наличия проверки корректности генотипа - соблюдения условия [9.1](#)

Для генерации начального приближения был использован жадный алгоритм. Сначала создается генотип из единиц, соответствующий конфигурации рюкзака, в который положены все предметы. Затем в случайном порядке единицы заменяются на нули, пока полученная конфигурация не будет удовлетворять условию корректности. После этого полученный генотип мутируется с помощью текущей мутации до заполнения пула решений.

В ходе работы было реализовано несколько алгоритмов мутации и скрещивания с целью сравнения их эффективности. Были реализованы следующие алгоритмы мутации:

- Мутация в одной позиции, при которой заменяется значение в одной случайно выбранной точке генотипа.
- Инверсионная мутация, при которой половина генотипа заменяется на противоположные значения.

Были реализованы следующие алгоритмы скрещивания:

- Скрещивание по 1 точке, при котором выбирается произвольная точка в последовательности генотипа, значения до точки берутся от первого генотипа, после - от второго.
- Скрещивание по двум точкам, при котором выбираются две различные произвольные точки, значения внутри интервала и в самих точках берутся из первого генотипа, вне интервала - из второго.
- Побитовое скрещивание, при котором значения на нечетных позициях берутся из первого генотипа, на четных - из второго.

В качестве функции оценки используется стоимость всех предметов, содержащихся в рюкзаке, соответствующем конфигурации.

### 9.2.3 Дополнения алгоритма

В связи со спецификой задачи в алгоритм были внесены дополнения.

Были введены проверки генотипов на корректность после мутации и скрещивания. Если генотип не удовлетворяет условию корректности, то значения начиная с первой позиции начинают зануляться до достижения генотипом корректности.

Были введены дополнительные пулы лучших конфигураций за время работы алгоритма. Такие пулы решают одновременно несколько задач

- Недопущение сильного ухудшения результатов решения вследствие случайных мутаций. При достаточно сильном ухудшении средней стоимости пула текущий пул замещается текущим пулом лучших конфигураций
- Возможность сохранения результатов при перезапуске алгоритма с другим начальным приближением. Такой перезапуск оправдан при получении генотипа - локального максимума, покинуть который с помощью мутации невозможно. Перезапуск алгоритма происходит при продолжительном сохранении лучшего значения в пуле конфигураций неизменным.
- Возможность сравнения решений после окончания работы алгоритма

## 10 Практическая часть

### 10.1 Наборы тестов

Для оценки эффективности работы алгоритма было проведено его тестирование на различных наборах тестов. Первый набор тестов взят из книги Петерсена(см. [5]) и содержит 7 задач. Условия этих задач(см таблицу 5) позволяют проверить эффективность простейшей версии генетического алгоритма без модификаций и оценить её эффективность.

Таблица 5: Параметры первого набора тестов

№ задачи	Размерность	Количество предметов
1	6	10
2	10	10
3	15	10
4	20	10
5	28	10
6	39	5
7	50	5

Второй набор тестов взят из статьи Чу (см [7]) и содержит в себе 30 задач с одинаковыми параметрами: размерность рюкзака равна 5, рассматривается 100 различных предметов. Для каждого набора тестов известно лучшее решение, эффективность алгоритма оценивалась по скорости поиска решения и числу итераций.

Для тестирования использовался компьютер с следующей аппаратно-программной базой:

- Процессор: Intel(R) Core(TM) i5-3230M CPU @ 2.60GHz (4 CPUs), 2.6GHz
- Количество оперативной памяти: 6144MB RAM
- Операционная система: Windows 10 Домашняя для одного языка 64-bit (10.0, Build 15063) (15063.rs2\_release.170317-1834)

Для сборки использовалась программная платформа .NET Framework 4.5.2.

## 10.2 Первый набор тестов

Рассмотрим результаты работы алгоритма без модификаций на первом наборе тестов. Экспериментальным образом были подобраны количество конфигураций в пуле(10) и процент мутирующих особей(20)

### 10.2.1 Мутация в одной позиции

Данная мутация показала себя эффективной с точки зрения решения задачи - для всех возможных вариантов скрещивания во всех тестах найден оптимальный результат(см таблицы 6, 7, 8) При сравнении результатов скрещивания можно заметить, что алгоритм побитового скрещивания в большинстве случаев несколько уступает в скорости другим алгоритмам.

Таблица 6: Одноточечная мутация, одноточечное скрещивание

№ теста	Итерации	Время, мс
1	6	8
2	114340	2694
3	712	18
4	4022	92
5	3049	97
6	400974	13481
7	9343432	396644

Таблица 7: Одноточечная мутация, двуточечное скрещивание

№ теста	Итерации	Время, мс
1	21	9
2	70930	1697
3	901	22
4	2092	67
5	2672	95
6	23261	8181
7	4770456	199461

Таблица 8: Одноточечная мутация, побитовое скрещивание

№ теста	Итерации	Время. мс
1	3	9
2	182633	3301
3	1455	49
4	7869	219
5	2198	74
6	289379	100133
7	348826	139403

### 10.2.2 Инверсионная мутация

Инверсионная мутация в контексте данной задачи оказалась менее эффективной, для тестов 5-7 решения не были найдены за 20 минут для всех вариантов скрещивания (см табл 9, 10, 11). Неэффективность такого варианта мутации может быть обусловлена применением алгоритма коррекции после мутации, что сводит на нет какой-либо положительный эффект от её применения. Также можно отметить существенно более высокую скорость решения в сравнении с алгоритмом одноточечной мутации для теста 2.

Таблица 9: Инверсионная мутация, одноточечное скрещивание

№ теста	Итерации	Время. мс
1	3	6
2	10	2
3	58	2
4	9538	560
5	-	-
6	-	-
7	-	-

Таблица 10: Инверсионная мутация, двуточечное скрещивание

№ теста	Итерации	Время. мс
1	2	10
2	16	1
3	356	19
4	7161	422
5	-	-
6	-	-
7	-	-



Таблица 11: Инверсионная мутация, побитовое скрещивание

№ теста	Итерации	Время. мс
1	10	10
2	23	1
3	1409	52
4	246001	12189
5	-	-
6	-	-
7	-	-

### 10.2.3 Промежуточные выводы

Исходя из приведенных тестов, можно считать подтвержденной принципиальную возможность реализованного алгоритма быть использованным для решения задачи о многомерном рюкзаке. Наилолее эффективной себя показало двуточечное скрещивание.

## 10.3 Второй набор тестов

Для проверки эффективности второго набора тестов были выбраны параметры запуска алгоритма, указанные в таблице 12.

Таблица 12: Параметры запуска второго набора тестов

Вид скрещивания	Двуточечное
Вид мутации	Одноточечная
Размер пула генотипов	10
Процент мутирующих особей	20%
Максимальное число итераций	2000000
Величина отклонения для сброса алгоритма	0.01
Количество итераций до полного перезапуска	200000

Максимальное число итераций было введено с целью ограничить время работы алгоритма. В случае, если лучшее решение не найдено, программа выдает на печать расхождение лучшего из найденных решений с текущим.

Рассмотрим результаты работы программы на втором наборе тестов. Для успешных решений среднее время составило 315 с, среднее число итераций - 652 007 (см. [BigResults1])

Таблица 13: Временная эффективность алгоритма

№ теста	Количество итераций	Время, с
1	-	888
2	197915	44
3	-	915
4	-	812
5	-	860
6	-	970
7	1253381	619
8	83207	39
9	262645	255
10	-	963
11	189635	83
12	-	912
13	-	912
14	-	825
15	-	881
16	942021	484
17	-	864
18	4386	5
19	-	880
20	-	865
21	-	734
22	1735194	752
23	-	779
24	845324	428
25	-	812
26	908217	454
27	594220	281
28	36434	35
29	-	895
30	1423522	625

В 43.3% случаев генетический алгоритм решил задачу и во всех случаях расхождение найденного и действительного максимумов расходится не более чем на 0,0021 с оптимальным результатом в относительных значениях(см [BigResults2]).

Таблица 14: Эффективность в достижении максимума задачи

№ теста	Относительное расхождение с максимумом
1	0,0021
2	0
3	0,0005
4	0.0022
5	0.0018
6	0.0014
7	0
8	0
9	0
10	0,0031
11	0
12	0,0020
13	0,0008
14	0,0001
15	0,0017
16	0
17	0,0006
18	0
19	0,0018
20	0,0005
21	0,0016
22	0
23	0,0008
24	0
25	0,0007
26	0
27	0
28	0
29	0,0006
30	0

+ улучшения

## ЗАКЛЮЧЕНИЕ

В ходе работы были выполнены все поставленные задачи. Была создана программа, реализующая генетический алгоритм с требуемыми для решения задачи модификациями. Было проведено тестирование, показавшее высокую эффективность программы на различных наборах тестов. Для малых наборов предметов (50 предметов и менее) оптимальное решение было достигнуто в 100. Для больших наборов (100 предметов) оптимум был получен в 43,3. При этом среднее время поиска решения составляет 5 минут 25 секунд. Таким образом, алгоритм может эффективно применяться в решении практических проблем.

## Список

### ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Gavish B., Pirkul H.* Allocation of Databases and Processors in a Distributed Computing System // Management of Distributed Data Processing. – 1982. – № 31. – С. 215–231.
2. *Shih W.* A Branch and Bound Method for the Multiconstraint Zero-One Knapsack Problem // Journal of the Operational Research Society. – 1979. – № 30. – С. 369–378.
3. *Gilmore P., Gomory R.* The Theory and Computation of Knapsack Functions // Operations Research. – 1966. – № 14.
4. *Mathews G.* On the partition of numbers // Proceedings of the London Mathematical Society. – 1897. – № 28. – С. 486–490.
5. *C.C.Petersen* Computational experience with variants of the Balas algorithm applied to the selection of R&D projects // Management Science. – 1967. – 13(9). – С. 736–750.
6. *Alex F.* Computer Models in Genetics. – McGraw-Hill, 1970.
7. *P.C.Chu, J.E.Beasley* A genetic algorithm for the multidimensional knapsack problem // Journal of Heuristics. – 1998. – Т. 4. – С. 63–86.

## **ПРИЛОЖЕНИЯ**

## A Исходный код программы

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace GAMultidimKnapsack
{
    class KnapsackConfig:IEquatable<KnapsackConfig>
    {
        private int[] CurrentConfiguration;

        public KnapsackConfig(int elementsAmount)
        {
            CurrentConfiguration = Enumerable.Repeat(-1, elementsAmount).ToArray();
        }

        public KnapsackConfig(int[] initConfig)
        {
            CurrentConfiguration = initConfig;
        }

        public KnapsackConfig(KnapsackConfig conf)//memberwise clone coud replace it or not
        {
            try
            {
                this.CurrentConfiguration = new int[conf.Length()];
                for (int i = 0; i < conf.Length(); i++)
                {
                    this.CurrentConfiguration[i] = conf.valueAt(i);
                }
            }
            catch (NullReferenceException ex)
            {
                Console.WriteLine("Empty configuration");
                return;
            }
        }

        public void setValueToActive(int position)
        {
            CurrentConfiguration[position] = 1;
        }

        public void setValueToPassive(int position)
        {
            CurrentConfiguration[position] = -1;
        }

        public void swapValue(int position)
```

```

    {
        CurrentConfiguration[position] = -CurrentConfiguration[position];
    }

    public bool isValueActive(int position)
    {
        return (CurrentConfiguration[position] > 0);
    }

    public int valueAt(int position)
    {
        return (CurrentConfiguration[position]);
    }

    public int Length()
    {
        try
        {
            return CurrentConfiguration.Length;
        }
        catch (NullReferenceException ex)
        {
            Console.WriteLine("Empty configuration");
            return 0;
        }
    }

    public bool Equals(KnapsackConfig sack)
    {
        if (this.Length() != sack.Length()) return false;
        for (int i = 0; i < this.Length(); i++)
        {
            if (this.valueAt(i) != sack.valueAt(i))
                return false;
        }
        return true;
    }

    public override bool Equals(object obj)
    {
        if (obj is KnapsackConfig)
            return Equals((KnapsackConfig) obj);
        return false;
    }

    public override int GetHashCode()
    {
        return ToString().GetHashCode();
    }

    public override string ToString()
    {
        return String.Join(",", CurrentConfiguration);
    }
}

using System;
using System.Collections.Generic;
using System.Linq;

```



```

using System.Text;

namespace GAMultidimKnapsack
{
    class GeneticalAlgorithm
    {
        private static int itemsAmount, dimensions;
        private static double[,] itemsSet; //amount of items*their dimensions
        private static double[] restrictions;
        private double[] itemsCosts;

        private int configsInPoolAmount;
        private int bestConfigsAmount;
        private KnapsackConfig[] configsPool;
        private KnapsackConfig[] currentBestConfigs; //however, this pool is resetted
            over and over again. We should create a permanent pool for practical use.
            Possibliy
        private KnapsackConfig[] bestConfigsAllTime;
        private double maximalKnapsackCost;

        private Crossover activeCrossover;
        private Mutation activeMutation;
        private static Random rand;
        private double mutationPercentage;

        public GeneticalAlgorithm(int itemsAm, int dim, double[] rest, double[] costs
            , double[,] myItemsSet, int confAm, Crossover myCrs, Mutation myMt, double
            mutationPercentage)
        {
            itemsAmount = itemsAm;
            restrictions = rest;
            dimensions = dim;
            itemsSet = new double[itemsAm, dim];
            rand = new Random();

            itemsSet = myItemsSet;
            itemsCosts = costs;
            configsInPoolAmount = confAm;

            activeCrossover = myCrs;
            activeMutation = myMt;

            bestConfigsAmount = configsInPoolAmount;

            configsPool = new KnapsackConfig[configsInPoolAmount];
            this.mutationPercentage = mutationPercentage;
            maximalKnapsackCost = itemsCosts.Sum();

            StartCycling();
        }

        private void StartCycling()
        {
            try
            {
                configsPool[0] = FirstApproachGenerate();

                int active = 0, passive = 0;
                for (int i = 0; i < itemsAmount; i++)
                {

```

```

        if (configsPool[0].isValueActive(i))
            active++;
        else passive++;
    }
    if (active == itemsAmount || passive == itemsAmount)
        return;
    for (int i = 1; i < configsInPoolAmount; i++)
    {
        configsPool[i] = activeMutation(configsPool[0], rand);
    }
    emptyBestConfigs(ref currentBestConfigs);
    emptyBestConfigs(ref bestConfigsAllTime);

    //int[] emptyConfig = (new int[itemsAmount]).Select(x => 0).ToArray();
    ;
    //bestConfigs = (new KnapsackConfig[bestConfigsAmount]).Select(x =>
        new KnapsackConfig(emptyConfig)).ToArray(); //HACK
}
catch (Exception ex)
{
    Console.WriteLine("Bugs in initialization");
    return;
}
}

void emptyBestConfigs(ref KnapsackConfig[] targetConfig)
{
    int[] emptyConfig = (new int[itemsAmount]).Select(x => 0).ToArray();
    targetConfig = (new KnapsackConfig[bestConfigsAmount]).Select(x => new
        KnapsackConfig(emptyConfig)).ToArray();
}

public void RestartAlgorithm(double flushPercent)
{
    try
    {
        IndexOutOfRangeException ex = new IndexOutOfRangeException(); //TODO:
        write exceptions class;
        if (flushPercent < 0 || flushPercent > 1)
            throw ex;
        else
        {
            //int[] emptyConfig = (new int[itemsAmount]).Select(x => 0).
            ToArray();
            //bestConfigs = (new KnapsackConfig[bestConfigsAmount]).Select(x
            => new KnapsackConfig(emptyConfig)).ToArray(); //HACK
            emptyBestConfigs(ref currentBestConfigs);
            int startFlushpoint = rand.Next(0, itemsAmount), endPoint,
            itemsToFlush = (int)(itemsAmount * flushPercent);
            if (startFlushpoint + itemsToFlush >= itemsAmount)
            {
                endPoint = itemsToFlush - (itemsAmount - startFlushpoint);
                foreach (var conf in configsPool)
                {
                    for (int i = startFlushpoint; i < itemsAmount; i++)
                        conf.SetValueToPassive(i);
                    for (int i = 0; i < endPoint; i++)
                        conf.SetValueToPassive(i);
                }
            }
            else

```

```

        {
            endPoint = startFlushpoint + itemsToFlush;
            foreach (var conf in configsPool)
            {
                for (int i = startFlushpoint; i < endPoint; i++)
                    conf.SetValueToPassive(i);
            }
        }
        //Сбрасываем целиком BestConfigs
        //Выносим flushpercent для текущих конфигураций 0.
        Конфигурации корректны .
    }
}
catch (Exception ex)
{
    Console.WriteLine("Flush_percent_is_in_[0;1]");
}
}
public void MakeIteration()
{
    if (GetKnapsackCost(configsPool[0]) == maximalKnapsackCost) return;
    List<int> positions = new List<int>();
    while (positions.Count < mutationPercentage * configsInPoolAmount)
    {
        positions.Add(rand.Next(configsInPoolAmount));
        positions.Distinct();
    }
    for (int i = 0; i < configsInPoolAmount; i++)
    {
        configsPool[i] = activeMutation(configsPool[i], rand);
    }
    KnapsackConfig[] CrossoverPool = new KnapsackConfig[configsInPoolAmount *
        2 - 2]; //not very well, if i want to customize Crossover ,but works
    for (int j = 0; j < configsInPoolAmount - 1; j++)
    {
        CrossoverPool[j] = activeCrossover(configsPool[j], configsPool[j +
            1], true);
        CrossoverPool[(CrossoverPool.Length - 1) - j] = activeCrossover(
            configsPool[j], configsPool[j + 1], false);
    }
    var tempConfigs = CrossoverPool
        .OrderByDescending(config => GetKnapsackCost(config))
        .Take(Convert.ToInt32(configsInPoolAmount))
        .ToArray();
    configsPool = tempConfigs;

    var tuningCoeff = 0.01;
    updateConfigs(ref currentBestConfigs, tuningCoeff);
    updateConfigs(ref bestConfigsAllTime, tuningCoeff);
}

private void updateConfigs(ref KnapsackConfig[] currentPool, double
    tuningCoeff)
{
    if (currentPool.Length >= configsPool.Length &&
        GetKnapsackCost(currentPool[0]) * (1 - tuningCoeff) > GetKnapsackCost(configsPool
            [0]))
    {
        for (int i = 0; i < configsInPoolAmount; i++)
            configsPool[i] = new KnapsackConfig(currentPool[i]);
    }
}

```

```

        return;
    }
    currentPool = currentPool
        .Concat(configsPool)
        .OrderByDescending(config => GetKnapsackCost(config))
        .Distinct()
        .Take(bestConfigsAmount)
        .ToArray();
}
private KnapsackConfig FirstApproachGenerate()
{
    KnapsackConfig result = new KnapsackConfig(itemsAmount);

    for (var i = 0; i < itemsAmount; i++)
    {
        result.setValueToActive(i);
    }
    Random rand = new Random();
    while (!IsValid(result))
    {
        int positionNumber = rand.Next(itemsAmount);
        while (!result.isValueActive(positionNumber))
        {
            positionNumber = rand.Next(itemsAmount);
        }
        result.setValueToPassive(positionNumber);
    }
    return result;
}

public delegate KnapsackConfig Crossover(KnapsackConfig sack1, KnapsackConfig
    sack2, bool isLeft);

public static KnapsackConfig FixedSinglePointCrossover(KnapsackConfig sack1,
    KnapsackConfig sack2, bool isLeft)
{
    int[] crossItems = new int[itemsAmount];
    if (isLeft)
    {
        for (var i = 0; i < itemsAmount / 2; i++)
            crossItems[i] = sack2.valueAt(i);
        for (var i = itemsAmount / 2; i < itemsAmount; i++)
            crossItems[i] = sack1.valueAt(i);
    }
    else
    {
        for (var i = 0; i < itemsAmount / 2; i++)
            crossItems[i] = sack1.valueAt(i);
        for (var i = itemsAmount / 2; i < itemsAmount; i++)
            crossItems[i] = sack2.valueAt(i);
    }

    KnapsackConfig CrossoverResult = new KnapsackConfig(crossItems);
    if (!IsValid(CrossoverResult))
        CrossoverResult = MakeValid(CrossoverResult);

    return CrossoverResult;
}

```

```

public static KnapsackConfig SinglePointCrossover(KnapsackConfig sack1,
    KnapsackConfig sack2, bool isLeft)
{
    int[] crossItems = new int[itemsAmount];
    int CrossoverPoint = rand.Next(itemsAmount);
    if (isLeft)
    {
        for (var i = 0; i < CrossoverPoint; i++)
            crossItems[i] = sack2.valueAt(i);
        for (var i = CrossoverPoint; i < itemsAmount; i++)
            crossItems[i] = sack1.valueAt(i);
    }
    else
    {
        {
            for (var i = 0; i < CrossoverPoint; i++)
                crossItems[i] = sack1.valueAt(i);
            for (var i = CrossoverPoint; i < itemsAmount; i++)
                crossItems[i] = sack2.valueAt(i);
        }
    }

    KnapsackConfig CrossoverResult = new KnapsackConfig(crossItems);
    if (!IsValid(CrossoverResult))
        CrossoverResult = MakeValid(CrossoverResult);
    return CrossoverResult;
}

public static KnapsackConfig BitByBitCrossover(KnapsackConfig sack1,
    KnapsackConfig sack2, bool isLeft)
{
    int[] crossItems = new int[itemsAmount];
    if (isLeft)
    {
        for (var i = 0; i < itemsAmount; i++)
        {
            if (i % 2 == 0)
                crossItems[i] = sack2.valueAt(i);
            else
                crossItems[i] = sack1.valueAt(i);
        }
    }
    else
    {
        for (var i = 0; i < itemsAmount; i++)
        {
            if (i % 2 == 0)
                crossItems[i] = sack1.valueAt(i);
            else
                crossItems[i] = sack2.valueAt(i);
        }
    }

    KnapsackConfig CrossoverResult = new KnapsackConfig(crossItems);
    if (!IsValid(CrossoverResult))
        CrossoverResult = MakeValid(CrossoverResult);
    return CrossoverResult;
}

```

```

public static KnapsackConfig TwoPointCrossover(KnapsackConfig sack1,
    KnapsackConfig sack2, bool isLeft)
{
    int firstPoint = rand.Next(itemsAmount - 1), secondPoint = rand.Next(
        firstPoint + 1, itemsAmount);
    int[] crossItems = new int[itemsAmount];
    if (isLeft)
    {
        for (var i = 0; i < firstPoint; i++)
            crossItems[i] = sack1.valueAt(i);
        for (var i = firstPoint; i < secondPoint; i++)
            crossItems[i] = sack2.valueAt(i);
        for (var i = secondPoint; i < itemsAmount; i++)
            crossItems[i] = sack1.valueAt(i);
    }
    else
    {
        for (var i = 0; i < firstPoint; i++)
            crossItems[i] = sack2.valueAt(i);
        for (var i = firstPoint; i < secondPoint; i++)
            crossItems[i] = sack1.valueAt(i);
        for (var i = secondPoint; i < itemsAmount; i++)
            crossItems[i] = sack2.valueAt(i);
    }
    KnapsackConfig sack = new KnapsackConfig(crossItems);
    if (!IsValid(sack))
        return (MakeValid(sack));
    return sack;
}

public delegate KnapsackConfig Mutation(KnapsackConfig sack, Random rand);

public static KnapsackConfig SinglePointMutation(KnapsackConfig sack, Random
    rand)
{
    KnapsackConfig mutatedSack = new KnapsackConfig(sack); //copy constructor
    int mutationPosition = rand.Next(itemsAmount);
    var count = 0;
    var iterationsToResque = 100000;
    while (mutatedSack.Equals(sack) && count < iterationsToResque) //TODO -
        not mutate empty sack
    {
        mutatedSack.swapValue(mutationPosition);
        if (!IsValid(mutatedSack)) //somehow unrealistic
        {
            mutatedSack.swapValue(mutationPosition);
            mutationPosition = rand.Next(itemsAmount);
        }
        count++;
    }
    if (count == iterationsToResque)
    {
        return MakeValid(mutatedSack);
    }
    return mutatedSack;
}

public static KnapsackConfig MutateHalf(KnapsackConfig sack, Random rand)
{
    KnapsackConfig mutatedSack = new KnapsackConfig(sack);

```

```

    int mutationPosition = rand.Next(itemsAmount);
    if (rand.Next() % 2 == 0)
    {
        for (var i = 0; i < itemsAmount / 2; i++)
        {
            mutatedSack.swapValue(i);
        }
    }
    else
    {
        for (var i = itemsAmount / 2; i < itemsAmount; i++)
        {
            mutatedSack.swapValue(i);
        }
    }
    if (!IsValid(mutatedSack))
        return (MakeValid(mutatedSack));
    return (mutatedSack);
}

private static bool IsValid(KnapsackConfig config)
{
    double[] summ = new double[dimensions];
    for (var i = 0; i < itemsAmount; i++)
    {
        if (config.isValueActive(i))
        {
            for (var j = 0; j < dimensions; j++)
            {
                summ[j] += itemsSet[i, j];
                if (summ[j] > restrictions[j]) return false;
            }
        }
        //Amount of items is much bigger than number of dimensions, so we can
        do checks on every turn.
    }
    return true;
}

private double GetKnapsackCost(KnapsackConfig sack)
{
    double count = 0;
    for (int i = 0; i < itemsAmount; i++)
        if (sack.isValueActive(i))
            count += itemsCosts[i];

    return count;
}

private static KnapsackConfig MakeValid(KnapsackConfig sack)
{
    for (var i = 0; i < sack.Length() && !IsValid(sack); i++)
    {
        sack.setValueToPassive(i);
    }
    return sack;
}

//methods for current active pool
public double GetNormalizedMaximalKnapsackCost()
{

```

```

        return GetAbsoluteMaximalKnapsackCost() / maximalKnapsackCost;
    }

    public double GetNormaizedAveragePoolCost()
    {
        return GetAbsoluteAverageKnapsackCost() / maximalKnapsackCost;
    }

    public double GetAbsoluteMaximalKnapsackCost()
    {
        return GetKnapsackCost(configsPool[0]);
    }

    public double GetAbsoluteAverageKnapsackCost()
    {
        if (GetKnapsackCost(configsPool[0]) == maximalKnapsackCost) return
            GetKnapsackCost(configsPool[0]);
        double averagePoolCost = 0;
        foreach (var config in configsPool)
        {
            averagePoolCost += GetKnapsackCost(config);
        }
        averagePoolCost /= configsInPoolAmount;
        return averagePoolCost;
    }

    public List<double> GetBestConfigsCosts()
    {
        return currentBestConfigs.Select(x => GetKnapsackCost(x)).ToList();
    }

    //methods for best all-time configs
    public double GetAbsoluteMaximalCostAllTime()
    {
        return GetKnapsackCost(bestConfigsAllTime[0]); //does it work is uncertain
    }
}

using GAMultidimKnapsack;
using System;
using System.Collections.Concurrent;
using System.Collections.Generic;
using System.Diagnostics;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Threading;
using System.Threading.Tasks;

namespace GAMultidimKnapsack
{
    public partial class SetPartition
    {
        static Random rand = new Random();
        static ConcurrentQueue<double> averageValuations = new ConcurrentQueue<double>();
        static ConcurrentQueue<double> maxValuations = new ConcurrentQueue<double>();
        static ConcurrentQueue<double> ages = new ConcurrentQueue<double>();
    }
}

```



```

static List<string> algorithmWithRestart(int itemsAmount, int dimensions,
    double maxCost, double[] restrictions, double[] costs, double[,] itemsSet)
{
    int ConfigsAmount = 10, restartTime = 2*100000, currentMaxValueLiveLength
        = 0;
    double PrevCost = 0;
    double mutationPercent = 0.20; //FROM 0 TO 1
    GeneticalAlgorithm ga = new GeneticalAlgorithm(itemsAmount, dimensions,
        restrictions, costs, itemsSet, ConfigsAmount, GeneticalAlgorithm.
        TwoPointCrossover, GeneticalAlgorithm.SinglePointMutation,
        mutationPercent);
    int iterationNumber = 0, endIteration = 2*1000000;
    List<double> resetPoints = new List<double>();
    var workTime = new Stopwatch();
    workTime.Start();

    // string logFileName = "plotwithreset.txt";
    // List<string> values = new List<string>();
    while (ga.GetAbsoluteMaximalCostAllTime() != maxCost && iterationNumber <
        endIteration)
    {
        var watch = new Stopwatch();
        watch.Start();
        ga.MakeIteration();
        iterationNumber++;
        double tmp = ga.GetBestConfigsCosts()[0];
        // values.Add(tmp.ToString());
        if (tmp != PrevCost)
        {
            PrevCost = tmp;
            currentMaxValueLiveLength = 0;
        }
        else
        {
            currentMaxValueLiveLength++;
        }
        /*
        if (iterationNumber % 10000 == 0) Отрисовка//
        {
            Console.Write(iterationNumber + " "); // delta with avg is " + (
                maxCost - ga.GetAbsoluteAverageKnapsackCost()) + "\n delta
                with max is " + (maxCost - ga.GetAbsoluteMaximalKnapsackCost()
                ));
            var bestCosts = ga.GetBestConfigsCosts();
            Console.WriteLine("Top 3 of the best configs pool are {0}, {1},
                {2}, {3}, {4}",
                (maxCost - bestCosts[0]),
                (maxCost - bestCosts[1]),
                (maxCost - bestCosts[2]),
                (maxCost - bestCosts[3]),
                (maxCost - bestCosts[4]));
        }
        */
        if (currentMaxValueLiveLength == restartTime)
        {
            var restartPercent = 0.4;
            resetPoints.Add(maxCost - ga.GetBestConfigsCosts()[0]);
            ga.RestartAlgorithm(restartPercent);
            PrevCost = 0;
            currentMaxValueLiveLength = 0;
        }
    }
}

```

```

        // Console.WriteLine("Restart");

    }
    watch.Stop();
}
workTime.Stop();
Console.WriteLine(workTime.Elapsed.TotalSeconds.ToString());
//Can use resetPoints for Something.
//File.WriteAllLines(logFileName, values);
if (ga.GetAbsoluteMaximalCostAllTime() == maxCost) //problem solved
    return transformResults(iterationNumber, workTime);

    else return transformResults(maxCost - ga.GetAbsoluteMaximalCostAllTime()
        , workTime);
}

static List<string> transformResults(int finishingIteration, Stopwatch
workTime)
{
    List<string> results = new List<string>();
    results.Add("Succeeded_in_" + finishingIteration + "_iterations_" +
        workTime.Elapsed.Seconds.ToString() + "_seconds");
    //string tmpString = "";
    //foreach (var x in resetPoints)
    //    tmpString += x.ToString() + ",";
    //results.Add(tmpString);
    //results.Add(resetPoints.Count.ToString());
    return results;
}

static List<string> transformResults(double difference, Stopwatch workTime)
{
    List<string> results = new List<string>();
    results.Add("Unsucceeded!_It_took_" + workTime.Elapsed.Seconds.ToString()
        + "_seconds_and_the_difference_between_best_of_found_configurations_
        and_known_maximum_is_" + difference.ToString());
    return results;
}

static void WriteResutls(int experimentNumber, List<string> results, string
filename)
{
    var fs = new FileStream(filename, FileMode.Append);
    using (StreamWriter sw = new StreamWriter(fs))
    {
        sw.WriteLine("Experiment_" + experimentNumber.ToString());
        foreach (var result in results)
            sw.WriteLine(result);
        sw.WriteLine();
    }
}

static void ProcessTestSet(string inputFileData, string inputFileResults) //
WORK WITH IT!
{
    using (StreamReader dataReader = new StreamReader(inputFileData))
    {
        string[] resultsArray = File.ReadAllLines(inputFileResults);
        var resultsStringNumber = 12;
        int experimentsAmount = Convert.ToInt32(dataReader.ReadLine());
    }
}

```

```

for (int experimentNumber = 0; experimentNumber < experimentsAmount;
    experimentNumber++, resultsStringNumber++)
{
    string[] initializationSequence;
    string firstString = dataReader.ReadLine();
    if (firstString.Trim() == "")
        initializationSequence = dataReader.ReadLine().Split("\u000A".
            ToCharArray(), StringSplitOptions.RemoveEmptyEntries);
    else initializationSequence = firstString.Split("\u000A".ToCharArray(),
        StringSplitOptions.RemoveEmptyEntries); ;
    int itemsAmount = Convert.ToInt32(initializationSequence[0]),
    dimensions = Convert.ToInt32(initializationSequence[1]);
    double maxCost = Convert.ToDouble(resultsArray[
        resultsStringNumber].Substring(25)); //Convert.ToDouble(temp);
    List<double> tempCosts = new List<double>();
    while (tempCosts.Count() != itemsAmount)
        tempCosts.AddRange(dataReader
            .ReadLine()
            .Split("\u000A".ToCharArray(), StringSplitOptions.
                RemoveEmptyEntries)
            .Select(x => Convert.ToDouble(x))
            .ToList());
    double[] costs = tempCosts.ToArray();
    double[,] itemsSet = new double[itemsAmount, dimensions];
    for (int i = 0; i < dimensions; i++)
    {
        int itemsReaden = 0;
        while (itemsReaden != itemsAmount)
        {
            double[] currentString = dataReader.ReadLine()
                .Split("\u000A".ToCharArray(), StringSplitOptions.
                    RemoveEmptyEntries).
                Select(x => Convert.ToDouble(x)).
                ToArray();
            for (int j = itemsReaden, k = 0; j < currentString.Count
                () + itemsReaden; j++, k++)
                itemsSet[j, i] = currentString[k];
            itemsReaden += currentString.Count();
        }
    }
    List<double> tempRestrictions = new List<double>();
    while (tempRestrictions.Count() != dimensions)
        tempRestrictions.AddRange(dataReader
            .ReadLine()
            .Split("\u000A".ToCharArray(), StringSplitOptions.
                RemoveEmptyEntries)
            .Select(x => Convert.ToDouble(x))
            .ToList());
    double[] restrictions = tempRestrictions.ToArray();
    //some silly work with reading from file.
    //написать перегрузку выбора алгоритма
    List<string> resultsList = algorithmWithRestart(itemsAmount,
        dimensions, maxCost, restrictions, costs, itemsSet);
    WriteResutls(experimentNumber, resultsList, "results.txt");

    Thread.Sleep(3000);
    maxValuations.Enqueue(0);
    averageValuations.Enqueue(0);
}
}

```

```

    }

    static void Main()
    {
        //new Thread(TestAlgorithm) { IsBackground = true }.Start();
        //new Thread(() => ProcessTestSet(@"C:\Users\black_000\Source\Repos\
        GeneticKnapsack\GAMultidimKnapsack\3.txt", @"C:\Users\black_000\Source
        \Repos\GeneticKnapsack\GAMultidimKnapsack\_res.txt")) { IsBackground =
        true }.Start();
        ProcessTestSet(@"C:\Users\black_000\Source\Repos\GeneticKnapsack\
        GAMultidimKnapsack\3.txt", @"C:\Users\black_000\Source\Repos\
        GeneticKnapsack\GAMultidimKnapsack\_res.txt");
    }
}

using GAMultidimKnapsack;
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading;
using System.Threading.Tasks;

namespace GAMultidimKnapsack
{
    public partial class SetPartition
    {
        static void Algorithm(int itemsAmount, int dimensions, double maxCost, double
            [] restrictions, double[] costs, double[,] itemsSet)//version w/o restarts
        {
            int ConfigsAmount = 10;
            double mutationPercent = 0.20;//FROM 0 TO 1
            GeneticalAlgorithm ga = new GeneticalAlgorithm(itemsAmount, dimensions,
                restrictions, costs, itemsSet, ConfigsAmount, GeneticalAlgorithm.
                TwoPointCrossover, GeneticalAlgorithm.SinglePointMutation,
                mutationPercent);
            int iterationNumber = 0;

            while (ga.GetAbsoluteMaximalKnapsackCost() != maxCost)
            {
                //var watch = new Stopwatch();
                //watch.Start();
                ga.MakeIteration();
                iterationNumber++;
                if (iterationNumber % 10000 == 0)
                {
                    Console.WriteLine(iterationNumber + ")_delta_with_avg_is_" + (
                        maxCost - ga.GetAbsoluteAverageKnapsackCost()) + "\n_delta_
                        with_max_is_" + (maxCost - ga.GetAbsoluteMaximalKnapsackCost()
                    ));
                    var bestCosts = ga.GetBestConfigsCosts();
                    Console.WriteLine("Top_3_of_the_best_configs_pool_are_{0},{1},{
                        2}",
                        (maxCost - bestCosts[0]),
                        (maxCost - bestCosts[1]),

```

```

        (maxCost - bestCosts[2]));
    }
    // watch.Stop();
}
Console.WriteLine("Finished in {0}", iterationNumber);
Console.ReadKey();
}

static void TestAlgorithm()//First proof of concept
{
    int itemsAmount = 500, dimensions = 6;
    double[] restrictions = new double[] { 100, 600, 1200, 2400, 500, 2000 },
        costs = new double[itemsAmount];
    for (int i = 0; i < itemsAmount; i++)
        costs[i] = rand.NextDouble() * 30;
    double[,] itemsSet = new double[itemsAmount, dimensions];
    for (int i = 0; i < itemsAmount; i++)
        for (int j = 0; j < dimensions; j++)
            itemsSet[i, j] = rand.NextDouble() * 50;
    int ConfigsAmount = 6;
    GeneticalAlgorithm ga = new GeneticalAlgorithm(itemsAmount, dimensions,
        restrictions, costs, itemsSet, ConfigsAmount, GeneticalAlgorithm.
        FixedSinglePointCrossover, GeneticalAlgorithm.SinglePointMutation,
        0.75);

    int iterationNumber = 0;
    while (true)
    {
        var watch = new Stopwatch();
        watch.Start();

        while (watch.ElapsedMilliseconds < 200)
        {
            ga.MakeIteration();
            iterationNumber++;
            averageValuations.Enqueue(ga.GetNormaizedAveragePoolCost());
            maxValuations.Enqueue(ga.GetNormalizedMaximalKnapsackCost());
        }
        watch.Stop();
    }
}

static void ProcessTestSet(string file) //worked with first files. proven its
    efficiency.TODO - check, if that true
{
    using (StreamReader sr = new StreamReader(file))
    {
        int experimentsAmount = Convert.ToInt32(sr.ReadLine());

        for (int experiment = 0; experiment < experimentsAmount; experiment
            ++)
        {
            string[] initializationSequence;
            string firstString = sr.ReadLine();
            if (firstString.Trim() == "")
                initializationSequence = sr.ReadLine().Split("_".ToCharArray
                    (), StringSplitOptions.RemoveEmptyEntries);
            else initializationSequence = firstString.Split("_".ToCharArray()
                , StringSplitOptions.RemoveEmptyEntries); ;
            int itemsAmount = Convert.ToInt32(initializationSequence[0]),
                dimensions = Convert.ToInt32(initializationSequence[1]);

```

```

        double maxCost = Convert.ToDouble(initializationSequence[2]);

        List<double> tempCosts = new List<double>();
        while (tempCosts.Count() != itemsAmount)
            tempCosts.AddRange(sr
                .ReadLine()
                .Split("␣".ToCharArray(), StringSplitOptions.
                    RemoveEmptyEntries)
                .Select(x => Convert.ToDouble(x))
                .ToList());
        double[] costs = tempCosts.ToArray();

        double[,] itemsSet = new double[itemsAmount, dimensions];
        for (int i = 0; i < dimensions; i++)
        {
            int itemsReaden = 0;
            while (itemsReaden != itemsAmount)
            {
                double[] currentString = sr.ReadLine()
                    .Split("␣".ToCharArray(), StringSplitOptions.
                        RemoveEmptyEntries)
                    .Select(x => Convert.ToDouble(x))
                    .ToArray();
                for (int j = itemsReaden, k = 0; j < currentString.Count
                    () + itemsReaden; j++, k++)
                    itemsSet[j, i] = currentString[k];
                itemsReaden += currentString.Count();
            }
        }
        List<double> tempRestrictions = new List<double>();
        while (tempRestrictions.Count() != dimensions)
            tempRestrictions.AddRange(sr
                .ReadLine()
                .Split("␣".ToCharArray(), StringSplitOptions.
                    RemoveEmptyEntries)
                .Select(x => Convert.ToDouble(x))
                .ToList());
        double[] restrictions = tempRestrictions.ToArray();
        Algorithm(itemsAmount, dimensions, maxCost, restrictions, costs,
            itemsSet);
        Thread.Sleep(3000);
        maxValuations.Enqueue(0);
        averageValuations.Enqueue(0);
    }
}

static List<string> multiThreadAlgorithms(int itemsAmount, int dimensions,
    double maxCost, double[] restrictions, double[] costs, double[,] itemsSet)
    //launches multiple algorithms with different start approximations. Does
    not work
{
    int ConfigsAmount = 10, algorithmsNumber = 3;
    double mutationPercent = 0.20;

    GeneticalAlgorithm[] gas = new GeneticalAlgorithm[algorithmsNumber];
    for (int i = 0; i < algorithmsNumber; i++)//пока предположим,
        что первые приближения различны
        gas[i] = new GeneticalAlgorithm(itemsAmount, dimensions, restrictions
            , costs, itemsSet, ConfigsAmount, GeneticalAlgorithm.

```

```

                TwoPointCrossover, GeneticalAlgorithm.SinglePointMutation,
                mutationPercent);
    int iterationNumber = 1;
    var controlWatch = new Stopwatch();
    controlWatch.Start();
    while (!gas.Select(x => x.GetAbsoluteMaximalKnapsackCost()).ToArray().
        Contains(maxCost))
    {
        Parallel.ForEach(gas, ga =>
            //foreach(GeneticalAlgorithm ga in gas)
            {
                ga.MakeIteration();
            });

        iterationNumber++;
        if (iterationNumber%10==0)
            Console.WriteLine(iterationNumber);
    }

    controlWatch.Stop();
    List<string> t=new List<string>();
    t.Add(iterationNumber.ToString());
    return t;
}

// transformResults
//     using (StreamWriter file1 = new StreamWriter(@"C:\Users\black_000\
// Documents\visual studio 2015\Projects\ConsoleKnapsack\ConsoleKnapsack\out.
// txt", true))
//         file1.WriteLine(iterationNumber + " iterations, " + controlWatch.
// ElapsedMilliseconds + " ms");
// }

}

using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.IO;
using System.Linq;
using System.Threading;
using static System.Net.WebRequestMethods;

namespace GAMultidimKnapsack
{
    //this whole file processes 1st file with fixed settings.
    class Program
    {
        private static List<long>[] averageTime=new List<long>[7];
        private static List<long>[] averageIterations = new List<long>[7];

        static void Algorithm(int itemsAmount, int dimensions, double maxCost, double
            [] restrictions, double[] costs, double[,] itemsSet, int testNumber)
        {
            int ConfigsAmount = 8;
            double mutationPercent = 0.75;
            GeneticalAlgorithm ga = new GeneticalAlgorithm(itemsAmount, dimensions,
                restrictions, costs, itemsSet, ConfigsAmount, GeneticalAlgorithm.
                BitByBitCrossover, GeneticalAlgorithm.MutateHalf, mutationPercent);
            int iterationNumber = 1;

```

```

var controlWatch = new Stopwatch();
controlWatch.Start();
while (ga.GetAbsoluteMaximalKnapsackCost() != maxCost)
{
    ga.MakeIteration();
    iterationNumber++;
}
controlWatch.Stop();
using (StreamWriter file1 = new StreamWriter(@"C:\Users\black_000\
Documents\visual_studio_2015\Projects\ConsoleKnapsack\ConsoleKnapsack\
out.txt", true))
    file1.WriteLine(iterationNumber + " iterations, " + controlWatch.
        ElapsedMilliseconds + " ms");
//averageTime[testNumber].Add(controlWatch.ElapsedMilliseconds);
//averageIterations[testNumber].Add(iterationNumber);
}

```

```

static void ProcessTestSet(string file)
{
    using (StreamReader sr = new StreamReader(file))
    {
        int experimentsAmount = Convert.ToInt32(sr.ReadLine());

        for (int experiment = 0; experiment < experimentsAmount; experiment
            ++)
        {
            string[] initializationSequence;
            string firstString = sr.ReadLine();
            if (firstString.Trim() == "")
                initializationSequence = sr.ReadLine().Split("_".ToCharArray
                    (), StringSplitOptions.RemoveEmptyEntries);
            else initializationSequence = firstString.Split("_".ToCharArray()
                , StringSplitOptions.RemoveEmptyEntries); ;
            int itemsAmount = Convert.ToInt32(initializationSequence[0]),
                dimensions = Convert.ToInt32(initializationSequence[1]);
            double maxCost = Convert.ToDouble(initializationSequence[2]);

            List<double> tempCosts = new List<double>();
            while (tempCosts.Count() != itemsAmount)
                tempCosts.AddRange(sr
                    .ReadLine()
                    .Split("_".ToCharArray(), StringSplitOptions.
                        RemoveEmptyEntries)
                    .Select(x => Convert.ToDouble(x))
                    .ToList());
            double[] costs = tempCosts.ToArray();

            double[,] itemsSet = new double[itemsAmount, dimensions];
            for (int i = 0; i < dimensions; i++)
            {
                int itemsReaden = 0;
                while (itemsReaden != itemsAmount)
                {
                    double[] currentString = sr.ReadLine()
                        .Split("_".ToCharArray(), StringSplitOptions.
                            RemoveEmptyEntries)
                        .Select(x => Convert.ToDouble(x))
                        .ToArray();

```



```

        for (int j = itemsReaden, k = 0; j < currentString.Count
            () + itemsReaden; j++, k++)
            itemsSet[j, i] = currentString[k];
        itemsReaden += currentString.Count();
    }
}
List<double> tempRestrictions = new List<double>();
while (tempRestrictions.Count() != dimensions)
    tempRestrictions.AddRange(sr
        .ReadLine()
        .Split("\n".ToCharArray(), StringSplitOptions.
            RemoveEmptyEntries)
        .Select(x => Convert.ToDouble(x))
        .ToList());
double[] restrictions = tempRestrictions.ToArray();
//using (StreamWriter file1 = new StreamWriter(@"C:\Users\
    black_000\Documents\visual studio 2015\Projects\
    ConsoleKnapsack\ConsoleKnapsack\out.txt", true))
//{
//    file1.Write(experiment + 1 + " ");
//}
Algorithm(itemsAmount, dimensions, maxCost, restrictions, costs,
    itemsSet, experiment);

// Thread.Sleep(3000);
}
}

static void Main2(string[] args)
{
    ProcessTestSet(@"C:\Users\black_000\Source\Repos\GeneticKnapsack\
        GAMultidimKnapsack\1.txt");

    //for (var i = 0; i < 7; i++)
    //{
    //    resultsTime[i] = averageTime[i].Sum() / testsAmount;
    //    resultsIterations[i] = averageIterations[i].Sum() / testsAmount;
    //}

    //using (StreamWriter file = new StreamWriter(@"C:\Users\black_000\
        Documents\visual studio 2015\Projects\ConsoleKnapsack\ConsoleKnapsack\
        out.txt", true))
    //{
    //    file.WriteLine("Iterations:");
    //    for (var i = 0; i < resultsTime.Length; i++)
    //        file.WriteLine(resultsTime[i]);
    //    file.WriteLine("Time:");
    //    for (var i = 0; i < resultsTime.Length; i++)
    //        file.WriteLine(resultsIterations[i]);
    //    file.Close();
    //}
}
}

```