Департамент образования города Москвы

Государственное бюджетное общеобразовательное учреждение

города Москвы

«Гимназия № 1505»

**ДИПЛОМНОЕ ИССЛЕДОВАНИЕ**

на тему:

[**Генератор**](http://research.gym1505.ru/node/14548) **учебного расписания**

Выполнил:

Ткаченко Никита Андреевич

Руководитель:

Наумов Алексей Леонидович

Москва

2019/2020 уч. г.

**Оглавление**

1. Введение
2. Глава 1
3. Глава 2
4. Заключение
5. Список литературы

**Введение**

Данные диплом посвящен проблеме генерации оптимального расписания. Задача по нахождению оптимального решения, или дисциплина "Исследование операций" существует уже давно. Эта дисциплина используется в различных сферах – от составления плана боевых действий и организации производства до такой, казалось бы, простой задачи, как составления обычных расписаний. Как ни странно, нередко составить качественное расписание для нескольких классов, которые подходят под заданные условия бывает трудоемко и неэффективно. В данной работе я попробую применить генетический алгоритм к задаче составления идеального расписания, а так же попрактиковаться в применении Объектно-Ориентированного подхода Программирования.

**Актуальность** данного диплома объясняется тем, что во многих школах и других учебных заведениях расписания составляют вручную. Это неэффективный и трудоемкий процесс, который в идеале можно будет упростить с помощью моего продукта.

Данная исследовательская работа не решает задачу составления идеального расписания, но изучает и предлагает одно из решений проблеме подбора и нахождения оптимального расписания.

**Цель диплома** – Разобраться в работе генетического алгоритма и написать прототип программы, которая использует генетический алгоритм для построения расписания.

**Задачи:**

1. Понять принцип работы генетического алгоритма.
2. Выбрать язык программирования, изучить минимальный необходимый материал для создания диплома.
3. Получить практический навык использования ООП для разработки программы.
4. Создать рабочий прототип.

**Критерии эффективности продукта:**

1. Продукт должен являться программой, построенной по принципу ООП
2. Продукт должен являться рабочей программой, на вход которой поступает набор параметров, а на выходе – расписание.

**Глава 1.**

**Генетический алгоритм** - эвристический алгоритм поиска (ответ может быть не идеально точным, но достаточно точным для поставленной задачи), который решает задачи с использованием механизма, похожего на естественный отбор в природе.

**Основные понятия:**

* Днк - набор параметров, который описывает, как выглядит и ведет себя объект (в данном случае - расписание)
* Ген - параметр, который описывает один признак объекта (к примеру, один урок)
* Популяция - набор объектов, каждый со своим днк.
* Приспособленность (fitness) - значение, которое определяет, насколько хорошо данное днк подходит к поставленной задаче (насколько близок объект к идеальному)
* Скрещивание - процесс, во время которого из двух различных объектов создается 1 "ребенок".
* Мутация - процесс, благодаря которому в Днк может случайно измениться один из параметров в лучшую или худшую сторону.
* Селекция - отбор лучших объектов в новую популяцию.

**Главная структура алгоритма выглядит так:**

* Задать идеал, к которому стремится алгоритм.
* Создать начальную популяцию днк.
* Цикл:
  1. Скрещивание.
  2. Мутация.
  3. Вычисление fitness’а для каждого объекта.
  4. Селекция.

Цикл продолжается до тех пор пока не будет получен достаточно близкий к идеалу объект, либо количество популяций достигнет установленного ограничения.

**Существующие методы реализации:**   
Дисциплина "Исследование операций" существует уже давно. Эта наука тесно связана с системным анализом, математическим программированием, методами искусственного интеллекта и другими направлениями, включающие в себя эвристические методы. Проблемами упорядочения и расписаний занимается отдел дискретной математики – Теория Расписаний, и хотя там используются различные подходы и сложный математический аппарат, как и во всей дисциплине ИП, лично я не нашел ни одного решения поставленной мною задачи с помощью генетического алгоритма.

**Глава 2.**

Продуктом данного диплома является программа на языке Python, построенная по принципу ООП, которая, используя генетический алгоритм, по заданным параметрам *(количество уроков, набор уроков, дни недели, ..)* создает приближенное к идеальному расписание. Расписания сохраняются в формате CSV и удовлетворяют поставленным условиям *(к примеру, недопустимо 3 одинаковых урока подряд, или недопустимо одни и те же уроки у разных классов в одно и то же время)* и параметрам.

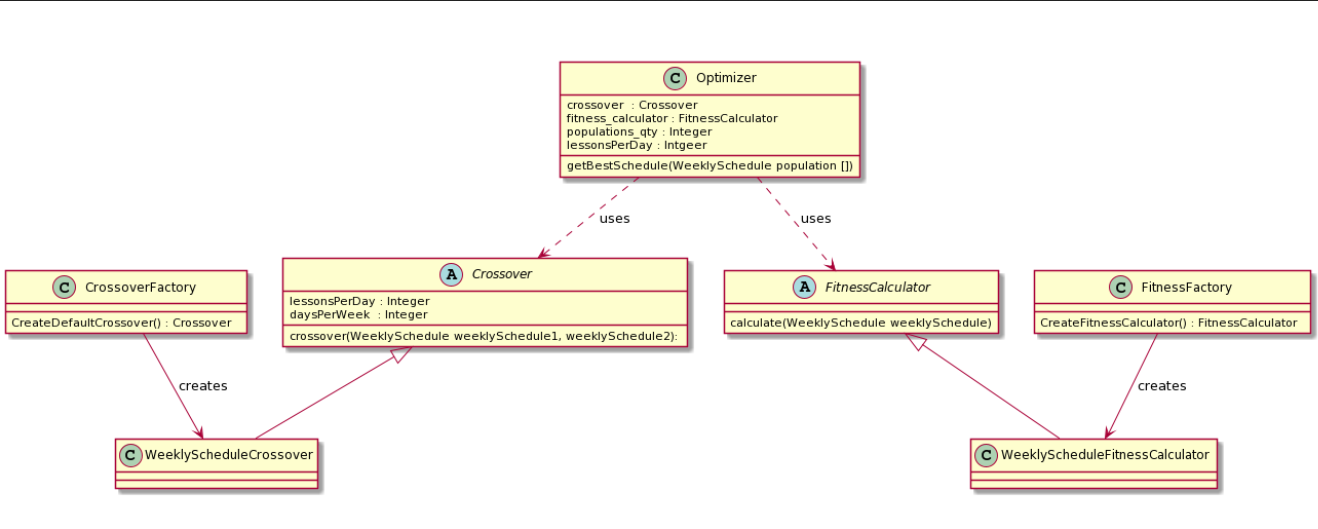
**Основные определения:**

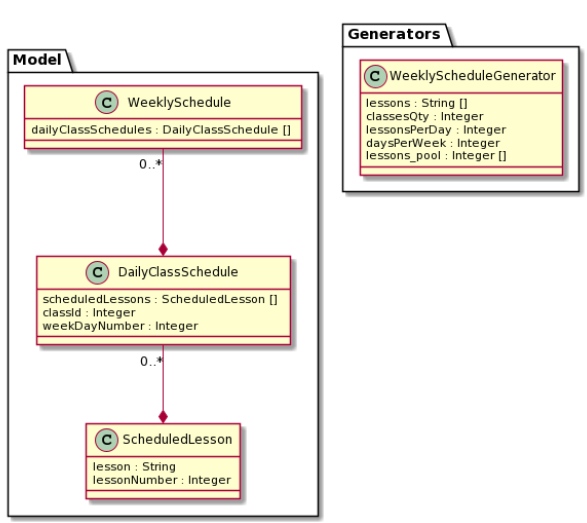
* **ООП – Объектно-Ориентированное программирование – совокупность методов, основанные на представлении программы в виде набора объектов, каждый из которых является экземпляром определенного класса, а классы наследуются друг от друга.**
* Расписание - план школьных уроков за определенный промежуток времени (день, неделя, месяц, ...)
* Урок - промежуток времени, во время которого ученики изучают 1 определенный предмет (математика, русский)
* Генератор - [Фабрика](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B1%D1%80%D0%B8%D1%87%D0%BD%D1%8B%D0%B9_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4_(%D1%88%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD_%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)) класса.

**Список файлов и классов:**

* main - основной файл программы. Здесь программа получает параметры из конфигурационного файла, создает все необходимые объекты и вызывает нужные функции для работы генератора.
* model - файл со всеми моделями
  1. WeeklySchedule - расписание на неделю.
  2. DailyClassSchedule - расписание на один день для одного класса.
  3. ScheduledLesson - запланированный урок.
* generators - файл в котором хранятся все генераторы.
  1. WeeklyScheduleGenerator - Генератор расписания на неделю.
* fitness\_calculator - файл с классами по вычислению fitness'а.
  1. WeeklyScheduleFitnessCalculator - класс вычисления fitness'а для одного расписания.
* crossover - файл скрещивания расписанийю
  1. WeeklyScheduleCrossover - класс по скрещиванию одной популяции.
* serializer - файл по сериализации и выводу расписания.
  1. Serializer - класс, который выводит расписание в текстовый файл.
  2. Csv\_serializer - класс, который выводит расписание в файл формата .csv (для просмотра с помощью электронных таблиц)
* optimizer - файл, в котором производится поиск оптимального расписания.
  1. GenericOptimizer - главный класс программы. В нем производятся все нужные операции для нахождения наиболее схожее с идеальным расписание.
* Общие классы:
  1. Factory - Фабрика определенного класса.
  2. ConfigurationClass - класс, в который передаются параметры для успешной работы другого класса.

UML – диаграмма программы





Весь код программы расположен на сайте Github (<https://github.com/tka4nik/Schedules_Generator>). Github – крупнейший веб-сервис для хостинга IT-проектов. Для установки и использования программы на странице проекта в git’е расположена инструкция. Настройка параметров производится в конфигурационном файле config.yaml, а выходные файлы расположены в папке outputs и представляют из себя файлы в формате .csv. Файлы такого формата легко открыть и в простом текстовом редакторе, и в редакторе электронных таблиц типа Excel.

**Заключение**

Во время работы над дипломом мне удалось выполнить все задачи и достичь поставленной цели. Однако продукт не идеален, и требует доработок: необходимо отладить алгоритм, сделать более удобный User Interface и предоставить пользователю больше информации о процессе работы программы. Знания и опыт программирования и применения ООП непременно пригодятся мне в последующих проектах и в решении проблем.

**Источники:**

1. Основная информация по генетическому алгоритму - <https://web.archive.org/web/20100404012527/http://www.geneticprogramming.us/>
2. Книга по английским терминам программирования (ISBN-13: 978-1719439558)

<https://www.amazon.com/Computer-Programming-Beginners-Fundamentals-Concepts/dp/1719439559/ref=tmm_pap_swatch_0?_encoding=UTF8&qid=&sr>=

1. Пример генетического алгоритма - <https://natureofcode.com/book/chapter-9-the-evolution-of-code/>
2. Книга по JavaScript (ISBN-13: 9780596527747) <https://ebooks-it.org/0596527748-ebook.htm>
3. Книга по Python (ISBN-13: 978-1-449-38267-4) https://www.ozon.ru/context/detail/id/7330472/