

ОЛИМПИАДА ШКОЛЬНИКОВ «ШАГ В БУДУЩЕЕ»

НАУЧНО-ОБРАЗОВАТЕЛЬНОЕ СОРЕВНОВАНИЕ «ШАГ В БУДУЩЕЕ,
МОСКВА»

273

регистрационный номер

Информатика и системы управления

Программное обеспечение ЭВМ и информационные технологии

Исследовательская работа – проект
«Система распределения студентов
между научными руководителями»

Автор:

Княжев Алексей Викторович
ГБОУ МО СП ФМЛ, 11 класс

Москва - 2020

Аннотация

Во время получения студентами высшего образования необходимо многократно выполнять курсовые работы по различным дисциплинам. Для этого необходимо определиться с выбором не только темы, но и научного руководителя. Данный вопрос легко решается, когда обучающийся может самостоятельно определиться. Наиболее часто встречаются ситуации, когда сделать выбор затруднительно. В этом случае целесообразно использовать автоматизированную систему распределения студентов и преподавателей.

Реализованный проект оказывает помощь в формировании распределения студентов между преподавателями с целью написания курсовых или проектных работ. При этом учитываются пожелания не только преподавателей, но и студентов.

Содержание

Введение	4
1 Сбор данных	5
1.1 Предпочтения преподавателей	5
1.2 Предпочтения студентов	5
2 Организация данных	7
2.1 Средство для хранения данных	7
2.2 Структура опроса	7
2.3 Данные для обработки	8
3 Алгоритм распределения	10
3.1 Обработка входных данных	10
3.2 Работа алгоритма	10
3.3 Асимптотика	11
4 Интерфейс	12
4.1 Выбор	12
4.2 Реализация Telegram-бота	13
4.2.1 Структура опроса	13
4.2.2 Обработка ответа на вопрос	13
4.2.3 Взаимодействие с пользователем	13
5 Техническая информация	15
Заключение	16
Приложение	17
Список литературы	18

Введение

В жизни каждого студента наступает момент, когда ему необходимо выполнить курсовую работу. Для успешного выполнения этой работы, ему необходима помощь опытного преподавателя — научного руководителя. Не у всех поиски наставника оканчиваются успехом. Эта проблема натолкнула меня на создание данной распределительной системы.

На данный момент в ВУЗах разделение студентов по научным руководителям осуществляется следующим образом:

- Преподаватель отбирает определенных студентов для проектной деятельности.
- Студент самостоятельно определяется с выбором научного руководителя.
- Куратор или иное ответственное лицо (ОЛ) распределяет студентов между преподавателями.

Система, разработанная в рамках выполнения данного проекта, способна оказать необходимую помощь куратору (или иному ОЛ) в решении вопроса распределения студентов между преподавателями, учитывая их предпочтения.

Распределение происходит на основе пожеланий не только студентов, но и преподавателей. Система производит сбор необходимой информации на основе специальных опросов, реализованных в формате чат-бота на платформе Telegram.

Целью данной работы является:

- Создание универсального чат-бота для опроса студентов с целью выявления их предпочтений по вопросу выбора научного руководителя.
- Обработка полученной в результате опроса информации.
- Получение конечного результата в виде списка распределения.

1 Сбор данных

1.1 Предпочтения преподавателей

При формировании списка тем курсовых или проектных работ каждый из возможных преподавателей по данной дисциплине указывает свое предпочтение по каждой из тем. Для этого используется шкала от 0 до 5, где

- * 0 — это отказ преподавателя от работы по данной теме.
- * 1 — нежелательная работа по данной теме.
- * 2 — возможная, но нежелательная работа по данной теме.
- * 3 — скорее возможная, чем нежелательная работа.
- * 4 — возможная работа по данной теме.
- * 5 — наиболее предпочтительная работа по данной теме.

Далее списки тем передаются куратору для последующей обработки.

1.2 Предпочтения студентов

Для определения предпочтений студентов при выборе научного руководителя используется чат-бот на платформе Telegram, в котором каждый из студентов должен пройти определенный опрос.

Необходимым условием для прохождения опроса является обязательная регистрация в системе Telegram с реальными личными данными (Фамилия, имя). Опрос формируется куратором и содержит перечень вопросов, необходимых для определения предпочтений студентов.

В системе опросов используются следующие принципы:

- * Формат ответа на вопрос — да/нет.
- * Создание дополнительных вопросов, ответ на которые мог бы отсеять сразу несколько заведомо неподходящих вариантов. Такой вопрос затрагивает не отдельную тему, а целый раздел данной дисциплины.
- * Вопросы с множественным выбором: варианты ответа — перечень тем одного раздела данной дисциплины.

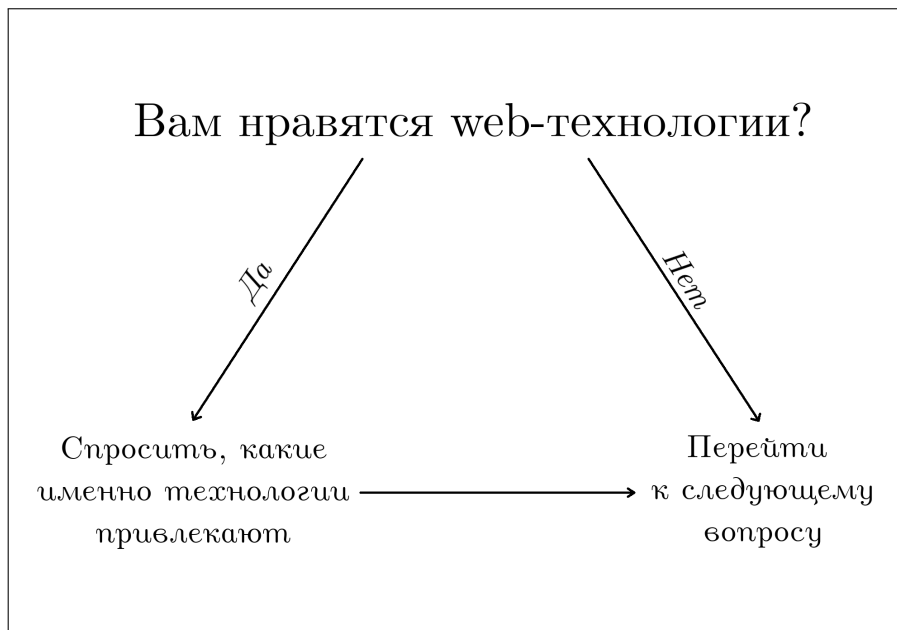


Рис. 1. Пример опроса

2 Организация данных

2.1 Средство для хранения данных

В качестве средства для хранения данных был выбран текстовый формат обмена данными JSON, основанный на Javascript.

Преимущества данного формата:

- + Легкость переноса — JSON данные представляют собой обычные текстовые файлы, для работы с которыми не нужно устанавливать какое-либо дополнительное программное обеспечение.
- + Простота работы и наличие предустановленных библиотек во многих языках программирования.
- + В случае возникновения потребности в использовании реальной базы данных, программу легко адаптировать для работы с базой данных MongoDB из-за схожести форматов JSON и BSON.

2.2 Структура опроса

Поскольку программа должна быть масштабируемой, описание структуры опроса должно быть вынесено вне исходного кода, чтобы её изменение не требовало recompilation.

Структура опроса описана в файле JSON и представляет собой список объектов — вопросов. Каждый такой объект содержит:

- * Текст вопроса.
- * Уникальный идентификатор вопроса (ID).
- * Тип вопроса:
 - Простой — вариант ответа в формате да/нет.
 - Вопрос со множественным выбором вариантов ответа.
- * Варианты ответа — список тем (для вопросов со множественным выбором).

Навигация от вопроса к вопросу осуществляется с помощью ID.

```
{
  "id": 12,
  "type": "multiple",
  "text": "Какие ОС вы хотите рассмотреть?",
  "options": [
    {
      "name": "windows",
      "display": "Windows"
    },
    {
      "name": "linux",
      "display": "GNU/Linux"
    }
  ]
}
```

Рис. 2. Пример описания структуры опроса

2.3 Данные для обработки

Данные о предпочтениях студентов и преподавателей хранятся в двух файлах формата JSON: `students.json` и `teachers.json` (Рис. 3).

```
"students": [
  {
    "name": "Иван Иванов",
    "props": {
      "c": 1,
      "asm": 0
    }
  }
]
```

Рис. 3. Пример описания структуры опроса

Информация о каждом студенте и преподавателе имеет одинаковую структуру и содержит:

- * **name** — фамилия, имя (для студентов); фамилия, имя, отчество (для преподавателей).
- * **props** — данные о предпочтениях.

Информация о предпочтениях представляет собой список тем и соответствующие каждой теме коэффициенты.

Ответы студентов на вопросы преобразуются для хранения в файле следующим образом:

- * Ответу «да» соответствует число 1.
- * Ответу «нет» соответствует число 0.

В соответствии с п. 2.1, предпочтения преподавателей, для удобства обработки, преобразуются следующим образом:

- * Критерию 0 соответствует число 0.0.
- * Критерию 1 соответствует число 0.2.
- * Критерию 2 соответствует число 0.4.
- * Критерию 3 соответствует число 0.6.
- * Критерию 4 соответствует число 0.8.
- * Критерию 5 соответствует число 1.0.

3 Алгоритм распределения

3.1 Обработка входных данных

Перед непосредственной работой алгоритма для удобства проводится предпросчет — рассчитывается специальный коэффициент, равный сумме произведений соответствующих предпочтений для каждого студента и преподавателя. Таким образом:

$$K_{teacher,student} = \sum tp_i \cdot sp_i,$$

где tp_i — i -ое предпочтение преподавателя;

sp_i — i -ое предпочтение студента.

На основе этих вычислений составляется таблица, в которой каждой паре студент-преподаватель соответствует коэффициент K_{ts} .

Затем происходит группировка по полю «Преподаватель». Каждая группа «преподаватель-студенты» заносится в отдельный элемент массива и сортируется по убыванию K_{ts} .

3.2 Работа алгоритма

После подготовки входных данных начинает работу алгоритм распределения.

На первом этапе определяется максимальное количество студентов, которое может быть закреплено за одним преподавателем после работы алгоритма. Обозначим это значение как N_{max} :

$$N_{max} = \left\lfloor \frac{N_{students}}{N_{teachers}} + 1 \right\rfloor$$

где $N_{students}$ — количество студентов;

$N_{teachers}$ — количество преподавателей.

После расчета N_{max} производится распределение, которое работает следующим образом:

- 1) Для каждого преподавателя обходим первые N_{max} элементов (Т.е. первую группу студентов, наиболее подходящих данному преподавателю).

- 2) Если какой-либо элемент из п.1) также встречается в первых N_{max} элементах других преподавателей, то алгоритм удаляет все вхождения этого элемента с меньшим K_{ts} у других преподавателей.
- 3) Алгоритм повторяется до тех пор, пока у каждого преподавателя количество студентов в списке не станет меньше либо равно N_{max} .

Таким образом, алгоритм распределения группирует студентов между преподавателями, учитывая их предпочтения.

3.3 Асимптотика

Стоит отметить, что алгоритм не является полным перебором. Мной установлено, что для любого распределения достаточно $N_{students} \cdot N_{teachers}^2$ повторений операций, описанных выше. Таким образом, асимптотика алгоритма будет составлять

$$\begin{aligned}
 N_{teachers} \cdot N_{students} \cdot N_{teachers} \cdot N_{max} \cdot N_{teachers} \cdot N_{max} &= \\
 &= N_{teachers}^3 \cdot N_{students} \cdot \frac{N_{students}^2}{N_{teachers}^2} = \\
 &= O(N_{students}^3 \cdot N_{teachers})
 \end{aligned}$$

Такая сложность вполне приемлема в контексте данной задачи, так как в данном случае от программы не требуется очень высокая скорость или обработка больших потоков данных.

4 Интерфейс

4.1 Выбор

Существует большое количество способов для создания интерфейса.

Рассмотрим возможные следующие для создания интерфейса:

- * Web-приложение.
- * Мобильное приложение.
- * Чат-бот в мессенджере.

Сравнительная характеристика представленных форматов указана в таблице (Табл. 1):

Характеристика	Web	Mobile	Bot
Кроссплатформенность	да	с ограничениями	да
Занимает память	нет	да	нет
Высокая скорость	да	да	да
Удобно на смартфонах	нет	да	да
Рассылка уведомлений	с ограничениями	на мобильных	да

Табл. 1. Сравнительная характеристика

На основании представленных данных наиболее оптимальным является использование чат-бота. Ввиду высокой популярности и развитой инфраструктуры чат-ботов было принято решение использовать платформу Telegram.

В качестве основного языка программирования был выбран язык Go. Выбор основа на:

- * Высокой скорости работы.
- * Простоте исходного кода.
- * Наличии необходимых библиотек.
- * Личном опыте использования.

4.2 Реализация Telegram-бота

4.2.1 Структура опроса

Как было описано в п. 2.2, опрос хранится в формате JSON, и каждый вопрос представляет собой отдельный объект. Для реализации этого в программе была создана структура, содержащая информацию об отдельном вопросе (Рис. 4).

```
type Quiz struct {  
    ID      int      'json:"id"'  
    Type    string   'json:"type"'  
    Text    string   'json:"text"'  
    Options []Option 'json:"options"'  
    Children []Quiz  'json:"children"'  
}
```

Рис. 4. Структура опроса в исходном коде программы

4.2.2 Обработка ответа на вопрос

Наиболее эффективной является такая структура опроса, в которой вопросы с ответами вида да/нет являются дополнительными, то есть помогают отсеять часть заведомо неподходящих тем. Таким образом, обработка ответов на такие вопросы представляет собой просто переход к следующему вопросу.

Обработка же вопросов со множественным выбором ответов сводится к занесению выбранных вариантов в список предпочтений студента и переходу к следующему вопросу.

4.2.3 Взаимодействие с пользователем

В чат-боте реализована работа двух основных команд:

- * `/start` — начало тестирования.
- * `/solve` — работа алгоритма распределения с выводом результата.

Тестирование происходит посредством встроенных callback-кнопок¹. Текст вопроса представлен в виде сообщения, а варианты ответа к нему — в виде callback-кнопок.

При нажатии такой кнопки вызывается специальный обработчик, который получает ID вопроса и выбранный вариант ответа, сохраняя их в оперативной памяти. На основе этих данных осуществляется переход к следующему вопросу. Данная процедура продолжается до достижения последнего вопроса. По окончании цикла происходит автоматическое сохранение полученных в результате опроса данных в файл `students.json`.

Чат-бот имеет функционал формирования списков групп студентов, закрепленных за конкретным преподавателем, и вывода этих списков на экран.

В приложении представлены примеры сообщений чат-бота.

¹CallBack-кнопки — кнопки, прикрепляемые к сообщениям, и позволяющие динамически обновлять сообщение или встроенные кнопки (не засоряя при этом ленту), а так же отображать уведомление вверху чат-бота или модальном окне

5 Техническая информация

Чат-бот был реализован с помощью языка программирования Go. Для взаимодействия с Telegram Bot API была использована популярная библиотека `go-telegram-bot-api` (github.com/go-telegram-bot-api/telegram-bot-api).

Кроме этой библиотеки были использованы встроенные в язык Go библиотеки, такие как:

- * `encoding/json` — работа с JSON.
- * `fmt` — создание строк (для вывода таблицы с результатами распределения).
- * `io/ioutil` — работа с файлами.
- * `log` — вывод служебной информации.
- * `strings` — работа со строками.
- * `sort` — сортировка.
- * `errors` — обработка ошибок.

Заключение

Материалом для работы в рамках данного проекта послужила необходимость автоматизации процесса распределения студентов между научными руководителями при написании курсовых и проектных работ.

В реализованном проекте разработана система, предусматривающая учитывать:

- * Предпочтения преподавателей к определенным темам работ.
- * Предпочтения студентов при выборе научного руководителя и тем проектов.

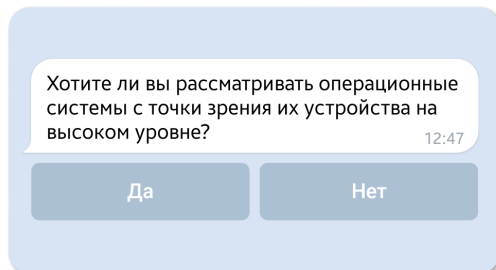
В рамках проекта созданы:

- * Система опроса студентов через чат-бот на платформе Telegram.
- * Система хранения исходных данных.
- * Алгоритм распределения студентов между преподавателями.

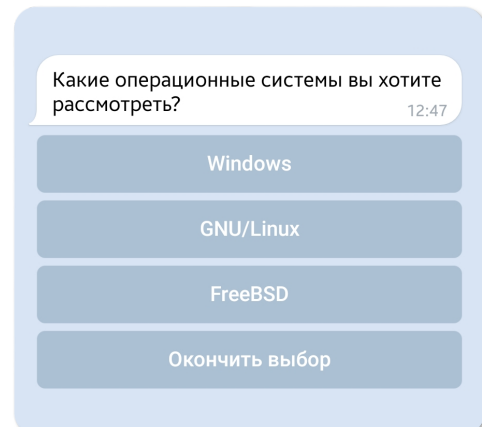
Результатом выполнения проекта является формирование списков групп студентов, закрепленных за конкретным преподавателем.

Поскольку проблема выбора научных руководителей актуальна в системе высшего образования, то предложенный мной метод распределения может быть рекомендован для использования в российских ВУЗах.

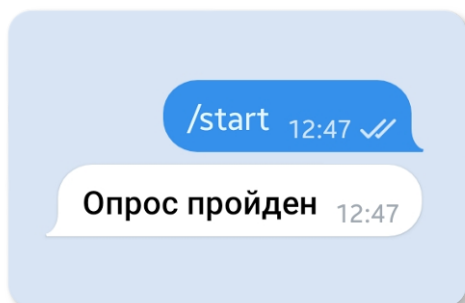
Приложение



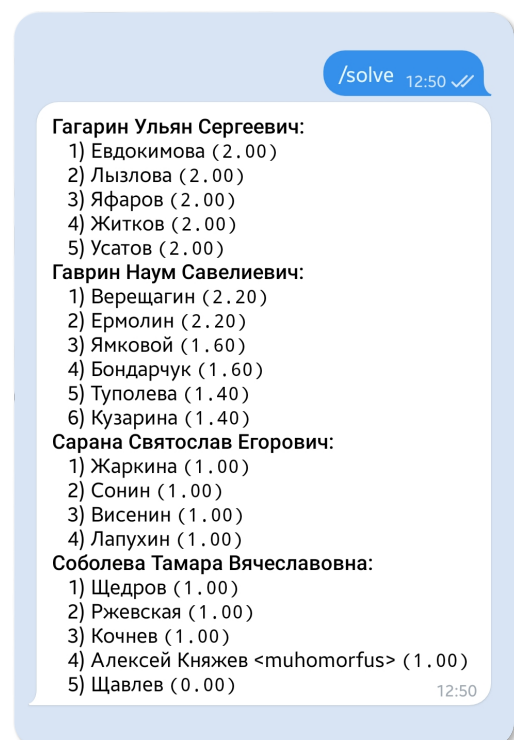
Вопрос с ответом вида да/нет



Вопрос с множественным
выбором ответа



Опрос пройден



Результаты распределения

Список литературы

- [1] Статья «Встроенные кнопки в Telegram Bot API — pyTelegramBotAPI» (*habr.com/ru/post/335886*)
- [2] Книга «The Go programing language» (Alan A. A. Donovan, Brian W. Kernighan)
- [3] Официальная документация языка программирования Go (*pkg.go.dev*)
- [4] Репозиторий библиотеки `go-telegram-bot-api` на GitHub (*github.com/go-telegram-bot-api/telegram-bot-api*)
- [5] Документация Telegram Bot API (*core.telegram.org/bots/api*)
- [6] Википедия — свободная энциклопедия (*ru.wikipedia.org*)