МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ ДНР

ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Институт КНТ

Факультет ФИСП

Кафедра ПИ им. Л.П. Фельдмана

Лабораторная работа №4

Тема: «Оценка качества разработанного исходного текста программы на основе лексического анализа»

Курс: Качество программного обеспечения и тестирование

Выполнил

ст. гр. ПИ-18Б

Моргунов А.Г.

Проверил

Федяев О. И.

Ищенко А. П.

Донецк – 2021

# Цель работы

Получить практические навыки по анализу программного кода при помощи метрик, а также получение практической рекомендации по улучшению кода на основе полученных метрик.

# Листинг кода

#include "Bot.hpp"

#include <algorithm>

namespace bot {

void Bot::RegistryNextStep(

ChatID chat\_id, std::function<void(const TgBot::Message::Ptr&)> handler) {

next\_steps\_.emplace\_back(chat\_id, handler);

}

auto Bot::FindNextStep(ChatID chat\_id) noexcept

-> std::list<std::pair<ChatID, Step>>::const\_iterator {

return std::find\_if(next\_steps\_.cbegin(), next\_steps\_.cend(),

[chat\_id](const std::pair<ChatID, Step>& value) {

return value.first == chat\_id;

});

}

bool Bot::IsFindNextStep(

std::list<std::pair<ChatID, Step>>::const\_iterator it) const noexcept {

return it != next\_steps\_.cend();

}

bool Bot::TryNextStep(const TgBot::Message::Ptr& message) {

auto it = FindNextStep(message->chat->id);

bool is\_find = IsFindNextStep(it);

if (is\_find) {

it->second(message);

next\_steps\_.erase(it);

}

return is\_find;

}

}

void SearcherBot::InitUserCommands() {

user\_commands\_ = {{"➖ Удалить ключевую фразу для подписки",

[this](const MessagePtr& message) {

on\_remove\_key\_phrase\_for\_subscribe(message);

}},

{"➕ Добавить ключевую фразу для подписки",

[this](const MessagePtr& message) {

on\_append\_key\_phrase\_for\_subscribe(message);

}},

{"➖ Удалить ключевую фразу для канала",

[this](const MessagePtr& message) {

on\_remove\_key\_phrase\_for\_channel(message);

}},

{"➕ Добавить ключевую фразу для канала",

[this](const MessagePtr& message) {

on\_append\_key\_phrase\_for\_channel(message);

}},

{"📗 Просмотреть подписки",

[this](const MessagePtr& message) {

on\_show\_subscribes(message);

}},

{"➖ Удалить подписку",

[this](const MessagePtr& message) {

on\_delete\_subscribe(message);

}},

{"➕ Добавить подписку",

[this](const MessagePtr& message) {

on\_insert\_subscribe(message);

}},

{"🔙 Выбор функции",

[this](const MessagePtr& message) {

SendMessage(message->chat->id, "Выберите действие.",

ui::UserStartMarkup());

}},

{"📜 Подписки",

[this](const MessagePtr& message) {

SendMessage(message->chat->id, "Выберите действие.",

ui::UserSubscribeMarkup());

}},

{"🔍 Найти",

[this](const MessagePtr& message) {

SendMessage(message->chat->id, "Введите запрос");

bot\_.RegistryNextStep(message->chat->id,

[this](const MessagePtr& message) {

on\_search\_messages(message);

});

}},

{"📒 Список каналов",

[this](const MessagePtr& message) {

on\_channels\_for\_search\_show(message);

}},

{"🔕 Выйти из канала",

[this](const MessagePtr& message) {

on\_leave\_from\_channel(message);

}},

{"📣 Добавить публичный канал",

[this](const MessagePtr& message) {

on\_insert\_channel\_for\_search(message);

}},

{"🔎 Поиск",

[this](const MessagePtr& message) {

SendMessage(message->chat->id, "Выберите действие.",

ui::UserSearchingMarkup());

}},

{"🔙 На главную", [this](const MessagePtr& message) {

db::User::ROLE role =

database\_.GetRole(message->from->id);

SendMessage(message->chat->id, "Вы на главной странице.",

GetStartMarkup(role));

}}};

}

# Входные выходные параметры

|  |  |
| --- | --- |
| Входные параметры | Выходные параметры |
| ➖ Удалить ключевую фразу для подписки | Ключевая фраза |
| ➕ Добавить ключевую фразу для подписки | номер текущей страницы |
| ➖ Удалить ключевую фразу для канала | всего страниц |
| ➕ Добавить ключевую фразу для канала | ссылка на исходное сообщение в канале |
| 📗 Просмотреть подписки | «СООБЩЕНИЕ» |
| ➖ Удалить подписку | «1» |
| ➕ Добавить подписку | «2» |
| 🔙 Выбор функции | «3» |
| 📜 Подписки | «4» |
| 🔍 Найти | «5» |
| 📒 Список каналов | «<<» |
| 🔕 Выйти из канала | «>>» |
| 📣 Добавить публичный канал | «Сообщение» |
| 🔎 Поиск | «1)» |
| 🔙 На главную | Имя канала |
| Ключевая фраза | «Список чатов» |
| ID канала получателя | ID чата |
| ID канала отправителя | Название чата |
| ID канала из списка каналов | «Ключевые фразы» |
| ID пользователя | «Прослушивает:» |
|  | «- 1.1» |
|  | «Id» |
|  | ID пользователя |
|  | Пользователи |
|  | Имя пользователя |
|  | «Администратор» |
|  | Имя администратора |
|  | Дата окончания аренды |
|  | Дата подачи заявки |

Входными параметрами являются команды, а также параметры к ним. Выходные параметры зависят от выбранной команды, и генерируются в зависимости от данных, хранящихся на сервере.

КРАТКОЕ ОПИСАНИЕ МОДУЛЕЙ СИСТЕМЫ

В процессе анализа требований к проекту были выявлены следующие сущности:

* Телеграм;
* Телеграмм-бот;
* База данных;
* Медиатор.

## 2.1.1 Телеграм

Телеграм – модуль реализован

## 2.1.2 Телеграмм-бот

Телеграмм-бот выступает пользовательским интерфейсом, то есть позволяет пользователю взаимодействовать с системой. Эта сущность интерпретирует команды, которые вводит пользователь и выполняет связанные с этой командой действия. Расширяет функциональность базового API бота путем предоставления возможности ведения цепочки сообщений для диалога с пользователем.

## 2.1.3 База данных

База данных – реализована на стороне сервера

## 2.1.4 Медиатор

Медиатор является связующим звеном системы. Он принимает сообщения, которые приходят из сущности телеграмма, берет из базы данных связи, которые относятся к сообществу, из которого пришло сообщение, фильтрует его, по ключевым словам, определенным для каждой связи, и рассылает ко всем адресатам, которые подходят по связям и ключевым словам. Телеграмм-бот является интерфейсом к данной сущности и позволяет менять конфигурацию связей в базе данных, регистрировать новых пользователей и администраторов, продлевать аренду. Преобразует список найденных новостей в сообщение с постраничным просмотром всех выданных результатов.

# Измеримые характеристики программы (словарь программы)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Оператор | Количество вхождений fi1 |  | Операнд | Количество вхождений  fi2 |
| include | 2 |  | Bot.hpp | 1 |
| namespase | 1 |  | <algorithm> | 1 |
| void | 3 |  | bot | 5 |
| if | 2 |  | chat\_id | 5 |
| this | 16 |  | handler | 2 |
| < | 5 |  | next\_steps\_ | 5 |
| > | 22 |  | value | 2 |
| = | 5 |  | it | 10 |
| - | 17 |  | is\_find | 3 |
| -> | 17 |  | message | 61 |
| , | 44 |  | user\_commands\_ | 1 |
| " " | 40 |  | ➖ Удалить ключевую фразу для подписки | 1 |
| . | 11 |  | ➕ Добавить ключевую фразу для подписки | 1 |
| bool | 3 |  | ➖ Удалить ключевую фразу для канала | 1 |
| MessagePtr& | 16 |  | ➕ Добавить ключевую фразу для канала | 1 |
| auto | 2 |  | 📗 Просмотреть подписки | 1 |
| return | 4 |  | ➖ Удалить подписку | 1 |
| ui::UserStartMarkup() | 1 |  | ➕ Добавить подписку | 1 |
| std::list | 2 |  | 🔙 Выбор функции | 1 |
| ChatID | 5 |  | 📜 Подписки | 1 |
| ; | 28 |  | 🔍 Найти | 1 |
| FindNextStep() | 2 |  | 📒 Список каналов | 1 |
| std::pair | 3 |  | 🔕 Выйти из канала | 1 |
| IsFindNextStep() | 2 |  | 📣 Добавить публичный канал | 1 |
| TryNextStep() | 1 |  | 🔎 Поиск | 1 |
| SearcherBot::InitUserCommands() | 1 |  | 🔙 На главную | 1 |
| RegistryNextStep() | 2 |  |  |  |
| { } | 40 |  |  |  |
| [ ] | 17 |  |  |  |
| std::function | 1 |  |  |  |
| :: | 23 |  |  |  |
| Step | 15 |  |  |  |
| noexcept | 2 |  |  |  |
| db::User::ROLE | 1 |  |  |  |
| cbegin() | 1 |  |  |  |
| emplace\_back() | 1 |  |  |  |
| cbegin() | 1 |  |  |  |
| cend() | 2 |  |  |  |
| erase() | 1 |  |  |  |
| RegistryNextStep | 2 |  |  |  |
| != | 1 |  |  |  |
| == | 1 |  |  |  |
| second | 1 |  |  |  |
| const | 22 |  |  |  |
| on\_remove\_key\_phrase\_for\_subscribe() | 1 |  |  |  |
| on\_append\_key\_phrase\_for\_subscribe() | 1 |  |  |  |
| on\_remove\_key\_phrase\_for\_channel() | 1 |  |  |  |
| on\_append\_key\_phrase\_for\_channel() | 1 |  |  |  |
| on\_show\_subscribes() | 1 |  |  |  |
| on\_delete\_subscribe() | 1 |  |  |  |
| on\_insert\_subscribe() | 1 |  |  |  |
| SendMessage() | 2 |  |  |  |
| UserStartMarkup() | 1 |  |  |  |
| UserSubscribeMarkup() | 1 |  |  |  |
| ui | 3 |  |  |  |
| on\_search\_messages() | 1 |  |  |  |
| on\_channels\_for\_search\_show() | 1 |  |  |  |
| on\_leave\_from\_channel() | 1 |  |  |  |
| on\_insert\_channel\_for\_search() | 1 |  |  |  |
| UserSearchingMarkup() | 1 |  |  |  |
| GetStartMarkup() | 1 |  |  |  |
| const\_iterator | 2 |  |  |  |

# Расчет метрик Холстеда

|  |  |  |  |
| --- | --- | --- | --- |
| Описание | Метрика | Формула | Значение |
| Число (уникальных) операторов | n1 |  | 62 |
| Число (уникальных) операндов | n2 |  | 26 |
| Общее число всех операторов | N1 | sum(f1) | 411 |
| Общее число всех операндов | N2 | Sum(f2) | 111 |
| Общее число входных и выходных параметров | n'2 |  | 49 |
| Словарь программы | n | n1+n2 | 88 |
| Длина реализации | N | N1+N2 | 522 |
| Длина программы | N' | n1\*log2(n1)+n2\*log2(n2) | 491,3716039 |
| Объём программы (в битах) | V | N\*log2(n) | 3371,823305 |
| Потенциальный объём программы | V' | (2+n’2)\*log2(2+n’2) | 87,08607898 |
| Уровень реализации программы | L | V’/V | 0,025827593 |
| Уровень языка | Lambda | L\*V’ | 2,249223778 |
| Работа по программированию | E | V/L | 130551,2033 |

# Оценка качества программного кода

Уровень программы приемлемый (L = 0,025827593).

Возможности языка программирования раскрыт (Lambda = 2,249223778)

Длина реализации приближается к теоретическому (N’ – N = 30,62839608).

# Рекомендации по улучшению программного кода

Уменьшить количество кода. Изменить количество переменных. Увеличить количество входных и выходных параметров.