**Задание 2. Чат-бот.**

Создайте программу с графическим интерфейсом пользователя. Реализуйте логику в отдельном модуле.

Вместо этого задания можно создать текстовый квест с сопоставимыми функциями.

**Требования**

* При запуске появляется окно для задания имени пользователя, потом основное окно.
* Ответ ботом на несколько реплик заданного шаблона («Привет, Бот!» и т.п.)
* Ответ на простые команды (вопросы). Например: «Который час?», вопросы о статистике по обмену сообщениями и т.п.
* Ответ на команды с параметрами: Например: «умножь 12 на 157»
* Бот должен хранить историю сообщений, включая время отправки и автора.
* Записывать историю в файл при завершении программы. Загружать из файла при запуске программы.
* "Отправка" сообщения по нажатию на Enter (Ctrl + Enter)
* *Дополнительно реализовать один или несколько пунктов (макс. Оценка на экзамене 3, если не выполнено)*:
  + Получение актуальной информации из интернета (погода, курсы валют, последние новости и т. п.); допустима задержка в отклике на действия пользователя во время выполнения запроса.
  + Запуск отдельных программ, работа с операционной системой и файлами.
  + Сохранение информации о собеседнике. Собеседник предполагается неизменным
  + Опционально: показ изображений (в том числе загрузка из интернета, например APOD))
* Требования к GUI:
  + шрифт и цветовая палитра (опционально: использование фоновых изображений) отличные от задаваемых по умолчанию.
  + иконка приложения
  + отправка сообщения по горячей клавише

**Рекомендации**

1. Используйте библиотеки для обработки естественного языка, регулярных выражений.
2. Помните, что класс (или классы) отвечающие за обработку сообщений должны быть независимы от интерфейса программы. Их, например, можно использовать для реализации онлайн бота.
3. Старайтесь сделать эти классы гибкими. В них должно просто добавлять новые методы обработки сообщений или реакции на сообщения.
4. Используйте абстрактные классы для описания программного интерфейса.

**См. также**

* Расстояние Левенштейна
* Word2vec, готовые словари word2vec (<https://rusvectores.org/ru/models/>)

**Вопросы**

1. Изобразите диаграмму классов для приложения.
2. Имеется ли интерфейс определяющий способы взаимодействия с классом обрабатывающим сообщения пользователя?
3. Что такое бизнес-логика?
4. Опишите шаблон проектирования «Модель – представление».
5. Что такое SOLID? Опишите каждый принцип.
6. Ваша лабораторная соблюдает принципы SOLID?
7. Что такое регулярное выражение?

**Ссылки**

**Примеры**

1. github.com/VetrovSV/OOP/tree/master/2021-fall/chat\_bot – шаблон для программы на C++
2. многооконные приложения в Qt:
   * github.com/VetrovSV/OOP/blob/master/Qt\_lec4.pdf
   * github.com/VetrovSV/OOP/tree/master/examples/Qt/multy-window
3. github.com/VetrovSV/OOP/blob/master/examples/cpp\_map/main.cpp – тип данных словарь (С++)
4. stackoverflow.com/questions/46943134/how-do-i-write-a-qt-http-get- request
5. <https://mkyong.com/java/java-how-to-download-web-page-from-internet> – Интернет-запросы в Java **регулярные выражения**
6. Фридл Дж. Регулярные выражения, 3-е издание. – Пер. с англ. – СПб.: СимволПлюс, 2008. – 608 с., ил.
7. regex101.com – сайт для проверки регулярных выражений
8. <https://www.rexegg.com/regex-quickstart.html> -- шпаргалка (eng) по регулярным выражениям
9. [https://regexper.com/](https://regexper.com/#%5B%D0%90-%D0%AF%5D%5B%D0%B0-%D1%8F%5D%7B1%2C10%7D%5Cs%3F%5B%D0%90-%D0%AF%5D%5C.%5B%D0%90-%D0%AF%5D%5C.) -- визуализация регулярных выражений
10. Сайт выдающий прогноз погоды в текстовом (plain text) формате: <https://github.com/chubin/wttr.in#usage>
11. [https://reqbin.com](https://reqbin.com/) тестирование web-API
12. <https://github.com/public-apis/public-apis> - Каталог сайтов с открытым API