МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

**«Сибирский государственный университет науки и технологий имени академика М.Ф. Решетнева»**

**(СибГУ)**

ИНСТИТУТ (ФАКУЛЬТЕТ) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

НАПРАВЛЕНИЕ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ПРОФИЛЬ ПОДГОТОВКИ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ**

РАЗРАБОТКА СИСТЕМЫ ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ В ЕЖЕДНЕВНОМ БЫТУ

Студент Жудрак Антон Петрович

Руководитель Коморовский Витольд Станиславович

Рецензент (Ф.И.О.)

Нормоконтроль (Ф.И.О.)

Допускается к защите

Зав. Кафедрой (Ф.И.О.)

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_ г.

Красноярск 20\_\_\_ г.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

**«Сибирский государственныйуниверситет науки и технологий имени академика М. Ф. Решетнёва»**

**(СибГУ)**

УТВЕРЖДАЮ

Зав. кафедрой ИУС

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_\_г.

**ЗАДАНИЕ**

**на магистерскую диссертацию**

Студенту\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_группы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

направления\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

магистерской программы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Тема диссертации\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Утверждена приказом по университету №\_\_\_\_\_\_\_от «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_\_\_г.

Срок сдачи студентом первого варианта диссертации «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_20\_\_\_\_г.

Срок сдачи студентом законченной диссертации «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_\_г.

Исходные данные к диссертации\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Содержание диссертации (перечень вопросов, подлежащих разработке)  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Перечень графического материала с указанием обязательных чертежей

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нормоконтроль\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания «\_\_\_»\_\_\_\_\_\_ 20\_\_г. \_\_\_\_\_\_\_\_\_\_ /Коморовский Витольд Станиславович/

**АННОТАЦИЯ**

ЖУДРАК АНТОН ПЕТРОВИЧ

РАЗРАБОТКА СИСТЕМЫ ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЙ В ЕЖЕДНЕВНОМ БЫТУ

НАПРАВЛЕНИЕ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

МАГИСТЕРСКАЯ ПРОГРАММА \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

УЧЕБНАЯ ГРУППА \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

ГОД ЗАЩИТЫ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ПРИМЕР.** В магистерской диссертации исследован вопрос и далее по работе.

**THE SUMMARY**

DEVELOPMENT OF THE SYSTEM OF SUPPORT OF DECISION-MAKING DURING EVERYDAY LIFE.

**EXAMPLE.** In master’s thesis investigated the issue of strategic planning and etc.

Содержание

[Введение 5](#_Toc482797672)

[1 Системы поддержки принятия решений 7](#_Toc482797673)

[1.1 История развития «систем поддержки принятия решений» 7](#_Toc482797674)

[1.2 Содержательная сущность поддержки принятия решений 11](#_Toc482797675)

[1.3 Преимущества МБ ДОБАВИТЬ ХЗ ПОКА 20](#_Toc482797676)

[2 Выбор среды для разработки 20](#_Toc482797677)

[2.1Причины использования Python 21](#_Toc482797678)

[2.1.1 Качество программного обеспечения 21](#_Toc482797679)

[2.1.2 Высокая скорость разработки 21](#_Toc482797680)

[2.1.3 Переносимость программ 22](#_Toc482797681)

[2.1.4 Библиотеки поддержки 22](#_Toc482797682)

[2.1.5 Интеграция компонентов 22](#_Toc482797683)

[2.2 История создания 23](#_Toc482797684)

[2.2.1 Влияние других языков 24](#_Toc482797685)

[2.3 Функциональные возможности 24](#_Toc482797686)

[2.4 Прикладное применение Python 26](#_Toc482797687)

[2.5 Преимущество Python перед другими языками высокого уровня 27](#_Toc482797688)

[2.6 Реализации Python 28](#_Toc482797689)

[2.7 Недостатки 29](#_Toc482797690)

[3 Телеграм(мессенджер) 31](#_Toc482797691)

[3.1 История создания телеграмма 31](#_Toc482797692)

[3.2Технология 33](#_Toc482797693)

[3.3 Основные возможности приложения Telegram 34](#_Toc482797694)

[3.4 Преимущества и недостатки использования Telegram 35](#_Toc482797695)

[3.5 Секретный чат 35](#_Toc482797696)

[3.5.1Создание секретного чата 36](#_Toc482797697)

[Источники 40](#_Toc482797698)

Введение

Стандартизация во все большей и большей степени определяет деятельность человека в профессиональной сфере. Это касается и информатики. В частности, в связи с развитием систем поддержки принятия решений (Decision Support Systems – DSS). Почему приходится обращать внимание на стандартизацию? В первую очередь потому, что всех интересует коммерческое применение прикладных решений. Продукт (товар, услуга) становится массовым только после унификации требований к нему, пусть даже с учетом пожеланий «узких» групп потребителей

**Актуальность исследования.**

В соответствии с изменениями в законодательстве в области образования отечественная высшая школа переходит на новую – «уровневую» - систему высшего профессионального образования (ВПО) и новые – «федеральные» - государственные образовательные стандарты (ФГОС). Новая система (структура) ВПО включает в себя два вида основных образовательных программ (ООП): для подготовки бакалавров (первый уровень) и для подготовки магистров (второй уровень).

При переходе на новую систему в рабочих программах появились несоответствия с учебными планами, вследствие чего рабочие программы теряют свою значимость и актуальность. Данная работа позволяет выявить такие несоответствия и предложить возможные решения данных ошибок.

**Цель исследования.** Анализ рабочих программ и учебных планов, выявление возможных ошибок с помощью иерархических структур и таблиц сравнительного анализа, проектирование и обучение программного средства, которое могло бы помочь избежать несогласованностей рабочих программ и учебных планов и посодействовать в их решении.

**Объект исследования.** Объектом исследования являются рабочие программы и учебные планы направлений подготовки «230100», «230400» и «231000»

**Предмет исследования** – Междисциплинарные связи.

**Методы исследования** – анализ, обобщение, систематизация, сравнение, сопоставление.

**Научная новизна исследования:** При переходе на новые стандарты возникла необходимость в разработке новых учебных планов, а, следовательно, и рабочих программ. Как правило, рабочие программы разрабатываются после утверждения учебных планов, что приводит к несоответствию места дисциплины в основной образовательной программе и учебном плане. Новизна данного исследования состоит в применении методов ранжирования для выявления несоответствия места дисциплины в структуре основной образовательной программы.

Не окончательный вариант, необходима доработка.

1 Системы поддержки принятия решений

1.1 История развития «систем поддержки принятия решений»

Система поддержки принятия решений (СППР) (англ. Decision Support System, DSS) — компьютерная автоматизированная система, целью которой является помощь людям, принимающим решение в сложных условиях для полного и объективного анализа предметной деятельности.

СППР возникли в результате слияния управленческих информационных систем и систем управления базами данных. Для анализа и выработки предложений в СППР используются разные методы. Это могут быть: информационный поиск, интеллектуальный анализ данных, поиск знаний в базах данных, рассуждение на основе прецедентов, имитационное моделирование, генетические алгоритмы, нейронные сети и другие. Некоторые из этих методов были разработаны в рамках искусственного интеллекта.

Если в основе работы СППР лежат методы искусственного интеллекта, то говорят об интеллектуальной СППР, или ИСППР. Близкие к СППР классы систем — это экспертные системы и автоматизированные системы управления. Современные СППР представляют собой системы, максимально приспособленные к решению задач повседневной управленческой деятельности, являются инструментом, призванным оказать помощь лицам, принимающим решения (ЛПР). С помощью СППР может производится выбор решений некоторых неструктурированных и слабоструктурированных задач, в том числе и многокритериальных. СППР, как правило, являются результатом мультидисциплинарного исследования, включающего теории баз данных, искусственного интеллекта, интерактивных компьютерных систем, методов имитационного моделирования.

Ранние определения СППР (в начале 70-х годов прошлого века) отражали следующие три момента:

(1) возможность оперировать с неструктурированными или слабоструктурированными задачами, в отличие от задач, с которыми имеет дело исследование операций;

(2) интерактивные автоматизированные (то есть реализованные на базе компьютера) системы;

(3) разделение данных и моделей.

СППР – совокупность процедур по обработке данных и суждений, помогающих руководителю в принятии решений, основанная на использовании моделей.

СППР – это интерактивные автоматизированные системы, помогающие лицу, принимающему решения, использовать данные и модели для решения слабоструктуризированных проблем или СППР – это система, которая обеспечивает пользователям доступ к данным и/или моделям, так что они могут принимать лучшие решения.

В настоящее время нет общепринятого определения СППР, поскольку конструкция СППР существенно зависит от вида задач, для решения которых она разрабатывается, от доступных данных, информации и знаний, а также от пользователей системы. Можно привести, тем не менее, некоторые элементы и характеристики, общепризнанные, как части СППР:

СППР – в большинстве случаев – это интерактивная автоматизированная система, которая помогает пользователю (ЛПР) использовать данные и модели для идентификации и решения задач и принятия решений. Система должна обладать возможностью работать с интерактивными запросами с достаточно простым для изучения языком запросов.

СППР обладает следующими четырьмя основными характеристиками:

1) СППР использует и данные, и модели;

2) СППР предназначены для помощи менеджерам в принятии решений для слабоструктурированных и неструктурированных задач;

3) Они поддерживают, а не заменяют, выработку решений менеджерами; 4) Цель СППР – улучшение эффективности решений.

До середины 60-х годов прошлого века создание больших информационных систем (ИС) было чрезвычайно дорогостоящим, поэтому первые ИС менеджмента (так называемые Management Information Systems – MIS) были созданы в эти годы лишь в достаточно больших компаниях. MIS предназначались для подготовки периодических структурированных отчетов для менеджеров. В конце 60-х годов появляется новый тип ИС — модель-ориентированные СППР (Model-oriented Decision Support Systems – DSS) или системы управленческих решений (Management Decision Systems – MDS). По мнению одних из первооткрывателей СППР Keen P. G. W., Scott Morton M. S., концепция поддержки решений была развита на основе «теоретических исследований в области принятия решений… и технических работ по созданию интерактивных компьютерных систем».

В 1971 г. – опубликована книга Scott Morton‘а, в которой, по-видимому, впервые были описаны результаты внедрения СППР, основанной на использовании математических моделей.

1974 г. – в работе [13] дано определение ИС менеджмента – MIS (Management Information System): «MIS – это интегрированная человеко-машинная система обеспечения информацией, поддерживающая функции операций, менеджмента и принятия решений в организации. Системы используют компьютерную технику и программное обеспечение, модели управления и принятия решений, а также базу данных».

1975 г. – J.D.C.Little в работе [1] предложил критерии проектирования СППР в менеджменте. 1978 г. – опубликован учебник по СППР [12], в котором исчерпывающе описаны аспекты создания СППР: анализ, проектирование, внедрение, оценка и разработка.

1980 г. – в [14] даны основы классификации СППР.

1981 г. – Bonczek, Holsapple и Whinston в книге [6] создали теоретические основы проектирования СППР. Они выделили четыре необходимых компонента, присущих всем СППР:

1) Языковая система (Language System – LS) – СППР может принимать все сообщения;

2) Система презентаций (Presentation System (PS)) (СППР может выдавать свои сообщения);

3) Система знаний (Knowledge System –KS) – все знания СППР сохраняет;

4) Система обработки задач (Problem-Processing System (PPS)) – программный «механизм», который пытается распознать и решать задачу во время работы СППР.

1981 г. – в книге [15] R.Sprague и E.Carlson описали, каким образом на практике можно построить СППР. Тогда же была разработана информационная система руководителя (Executive Information System (EIS)) – компьютерная система, предназначенная для обеспечения текущей адекватной информации для поддержки принятия управленческих решений менеджером. Начиная с 1990-х, разрабатывают так называемые Data Warehouses – хранилища данных.

В 1993 г Е. Коддом (E.F. Codd) для СППР специального вида был предложен термин OLAP (Online Analytical Processing)- оперативный анализ данных, онлайновая аналитическая обработка данных для поддержки принятия важных решений. Исходные данные для анализа представлены в виде многомерного куба, по которому можно получать нужные разрезы — отчёты. Выполнение операций над данными осуществляется OLAP-машиной. По способу хранения данных различают MOLAP, ROLAP и HOLAP. По месту размещения OLAP-машины различаются OLAP-клиенты и OLAP-серверы. OLAP-клиент производит построение многомерного куба и вычисления на клиентском ПК, а OLAP-сервер получает запрос, вычисляет и хранит агрегатные данные на сервере, выдавая только результаты.

В начале нового тысячелетия была создана СППР на основе Web. 27 октября 2005 года в Москве на Международной конференции «Информационные и телемедицинские технологии в охране здоровья» (ITTHC 2005), А. Пастухов (Россия) представил СППР нового класса — PSTM (Personal Information Systems of Top Managers). Основным отличием PSTM от существующих СППР является построение системы для конкретного лица, принимающее решение, с предварительной логико-аналитической обработкой информации в автоматическом режиме и выводом информации на один экран.

1.2 Содержательная сущность поддержки принятия решений

Деятельность деловых людей связана с необходимостью ежедневно принимать решения различной сложности. Примерами могут служить: выбор направления развития фирмы, выбор варианта автоматизации деятельности компании, определение варианта размещения филиала, типа выпускаемого или закупаемого товара, выбор помещения под офис, типа оборудования, выбор кредитора, соисполнителя работы, назначение на вакантную должность одного из многих кандидатов и т. д. Обоснованность и профессиональный уровень принимаемых решений определяет, в конце концов, эффективность деятельности фирмы. Необходимость учета при принятии управленческих решений большого количества политических, экономических, социальных, юридических и моральных факторов значительно усложняет задачу выбора правильного варианта решения. В первую очередь, это связано с необходимостью сбора необходимой для принятия решения информации. В этом отношении существенную помощь руководителю оказывают современные информационные системы. Однако обладание необходимой информацией - необходимое, но недостаточное условие для принятия правильного решения.

Действительно, проведем такую аналогию. Представьте себе, что вы обладаете всеми анализами, рентгенограммами, кардиограммами, томограммами и так далее, по которым можно поставить правильный диагноз и принять правильное решение о методах и средствах лечения. Однако для этого нужно обладать знаниями врача, причем, как правило, в сложных случаях привлекать врачей нескольких специальностей. При принятии действительно сложных решений необходимо привлекать экспертов - специалистов в различных областях знаний, ибо, как говаривал Козьма Прутков, «нельзя объять необъятное». Не может один, даже самый талантливый, человек быть действительно большим профессионалом во многих областях знаний. Однако, чтобы эффективно использовать знания экспертов, нужно, во-первых, знать, какие эксперты вам необходимы, во-вторых, какие вопросы ставить перед ними, и, наконец, как использовать их знания для принятия решения. Ошибкой будет поручать эксперту принимать решение. Опираясь на знания экспертов, принимает решение руководитель, поставивший задачу. Допустим далее, что как-то вам удалось собрать нужных экспертов и извлечь их нужные для обоснования решения знания. Однако эксперты показали, что существует великое множество взаимосвязанных факторов, влияющих на эффективность решения. Каждый из экспертов знает, как связаны факторы, относящиеся к его компетенции, но и только. А для принятия решения нужно учесть взаимосвязь всех факторов, относящихся к компетенции разных экспертов. Таких связей могут быть десятки, а для сложных проблем (например, определения стратегического направления развития фирмы) и более. В то же время руководитель, принимающий решение, - обычный человек с присущими всем людям психофизиологическими ограничениями, заключающимися, в частности, в том, что человек может оперировать одновременно не более, чем рядом объектов. Где же пути разрешения имеющегося противоречия? Оно может быть разрешено только на основе применения математических методов, воплощенных в современных информационных системах поддержки принятия решений (СППР). Опыт человека, многих людей может быть аккумулирован в форме накопленного тысячелетиями опыта логики, накопленного опыта ближайших предшествующих лет в форме приложений конкретного вида моделей к однотипным ситуациям. Наконец, опыт отражается в виде сопоставляемых с эталонами данных ближайших предшествующих периодов для конкретного предприятия, в конкретных реальных условиях, и такой опыт может быть востребован наряду с указанным методическим опытом в виде моделей и методов обработки имеющихся данных. Эти системы начинают все шире применяться государственными организациями и крупными корпорациями (U.S. Navy, NASA, IBM, General Motors, Xerox, 3M, Rockwell International, Reiter Consulting Group International и др.) Примеры задач, решаемых с привлечением СППР:

* обоснование направлений развития систем высшего образования США на период 1985-2000 годы;
* выбор методов завоевания рынка бытовой техники; оценка привлекательности в ближайшие 10 лет регионов США для трудоустройства людей, окончивших колледж;
* распределение средств между мероприятиями, направленными на уменьшение бандитизма;
* оценка перспективности видов альтернативного горючего для автомобилей;
* распределение средств между проектами социальной программы гуманитарной направленности;
* отбор научно-технических проектов в рамках конкурса; выбор перспективных направлений информатизации страны;
* выбор ERP-системы для корпорации и пр.

В последнее время СППР начинают применяться и в интересах малого и среднего бизнеса (например, выбор варианта размещения торговых точек, выбор кандидатуры на замещение вакантной должности, выбор варианта информатизации и т. д.) Главной задачей, которую приходится разрешать при принятии решения, является выбор альтернативы, наилучшей для достижения некоторой цели, или ранжирование множества возможных альтернатив по степени их влияния на достижение этой цели. В настоящее время разработано большое количество методов оказания помощи лицу, принимающему решение (ЛПР), при осуществлении этой задачи. Рассмотрим основные из них. Прежде всего, необходимо решить нетривиальную задачу выбора множества критериев оценки альтернатив. Для этого может быть использован, в частности, подход, предусматривающий декомпозицию главной цели до того уровня детализации, когда для нижнего уровня иерархии целей можно сформулировать критерии, позволяющие адекватно описать степень достижения целей при принятии той или иной альтернативы. Применительно к задаче выбора варианта информатизации в качестве главной цели фирмы может быть, например, повышение рентабельности фирмы, а критериями оценки вариантов могут выступать, например, затраты на информатизацию, способность поддерживать решения, возможность адаптации к другим видам деятельности фирмы, возможность защиты информации, время реакции на запрос, надежность оборудования и пр. Наборы значений этих критериев используются для описания исходов альтернативных вариантов решений (в дальнейшем, «альтернатив») с последующим формированием функции полезности, отражающей предпочтения ЛПР и позволяющей каждому набору значений оценок альтернатив по выбранным критериям поставить в соответствие некоторое число, характеризующее эффективность альтернативы. Построение функции полезности на основании формального описания предпочтений предполагает, в частности, способность ЛПР выполнить замещения, то есть в простейшем случае выбрать в качестве лучшей одну из двух альтернатив, причем одна из них превосходит вторую по некоторому критерию А, но уступает по критерию В при равных оценках по остальным. Для этого ЛПР должен полностью владеть проблемой. Применительно к задаче выбора варианта информатизации фирмы ЛПР должен, в частности, указать, какой из двух вариантов предпочтительней для повышения рентабельности фирмы: тот, который обеспечивает возможность поддержки принятия решений, но уступает по времени реакции, или вариант, обеспечивающий меньшее время реакции на запрос, но не содержащий средств поддержки принятия решений при равных оценках по остальным критериям. Как видно, ЛПР совсем не просто сделать выбор! Для действительно сложных комплексных проблем допущение о способности ЛПР осуществлять подобные замещения весьма сильно, ЛПР не может обладать всеми знаниями, необходимыми для квалифицированного решения этой задачи. Поэтому для решения таких сложных проблем следует привлекать многих специалистов в разных областях знаний, что при использовании такого подхода весьма затруднительно. Упрощение построения функции полезности достигается при использовании метода аналитических иерархических процессов (АНР). Сущность подхода заключается в следующем. ЛПР осуществляет вначале попарное сравнение значимости выбранных критериев. Затем этот же метод используется для попарного сравнения альтернатив относительно каждого выбранного критерия. На основе этого СППР рассчитывает коэффициенты значимости критериев, коэффициенты значимости альтернатив относительно каждого критерия, что позволяет рассчитать для каждой альтернативы значения линейной функции полезности. Развитием этого подхода является метод аналитических сетевых процессов (ANP), который позволяет учесть взаимосвязи между критериями. Однако использование каждого из трех названных подходов связано с необходимостью ЛПР решать сложную задачу выбора набора критериев, адекватно описывающих влияние альтернатив на достижение главной цели. Особенно сложным становится эта задача при принятии решений относительно формирования комплексных целевых программ. Опыт разработки сложных СППР показывает, что для ЛПР понятие «главная цель» программы более близко, чем понятие «критерий». Вследствие этого при разработке сложных СППР удобнее пользоваться понятиями «цель», «подцель», «надцель» и рассматривать процесс выполнения программы как достижение взаимосвязанных подцелей. Для оценки эффективности проектов в этом случае более приемлемым представляется подход, который можно назвать многоцелевым оцениванием альтернатив. Сущность его заключается в следующем. Каждый из проектов (альтернатива) оценивается единым показателем эффективности - степенью влияния его выполнения на достижение главной цели программы. Применительно к задаче выбора варианта информатизации фирмы главной целью может быть «повышение рентабельности». Непосредственная оценка влияния выполнения проектов на достижение главной цели программы является трудной задачей. Поэтому используется прием, основанный на построении иерархии целей, то есть на декомпозиции главной цели программы на подцели, где на нижнем уровне иерархии находятся цели проектов. Применительно к выбора варианта информатизации фирмы в качестве непосредственных подцелей главной цели могут быть конкретные цели такие, как: ускорение оборачиваемости средств, увеличение контролируемой части рынка, снижение цены продукции и т. д. Заметим, что между подцелями могут существовать взаимосвязи, например, подцель «снижение цены продукции» оказывает положительное влияние на достижение подцели «увеличение контролируемой части рынка». Применительно к рассматриваемому примеру в качестве проектов выступают различные варианты информатизации. В общем случае иерархия целей содержит как стимулирующие, так и угнетающие прямые и обратные связи между целями различных типов, что позволяет отразить многообразие взаимовлияний целей более полно, чем при использовании критериев. После построения иерархии эксперты количественно оценивают влияние подцелей, в том числе и проектов, на достижение непосредственных надцелей, используя методы парных сравнений либо непосредственного оценивания. На основании этой информации, а также структуры иерархии подцелей рассчитываются показатели относительной эффективности проектов. Такая технология построения иерархии целей позволяет действительно квалифицированно структурировать проблему, наметить промежуточные цели и превратить ее в базу знаний о проблеме, аккумулирующей знания экспертов именно в областях их действительной («проверенной временем») компетенции. Очевидно, напрашивается вопрос: «А где же здесь ЛПР? Не доверился ли он полностью экспертам и не передал ли он им право принимать решение?» Нет, это не так. ЛПР решает стратегические вопросы: формулирует главную цель программы, выражая тем самым свой замысел. Кроме того, он, как правило, определяет подцели главной цели, решая оперативные задачи. Если к тому же он - высокий профессионал в какой-нибудь узкой области (например, финансист), то может выступать в роли эксперта и при декомпозиции некоторой цели более низкого порядка, относящейся к области своей компетенции. Может возникнуть еще один вопрос: «Нельзя ли, осуществив декомпозицию главной цели на ее непосредственные подцели, разделить сложную задачу на более простые, для которых полученные подцели являются главными?» Как правило, нельзя, так как при этом не будут учитываться многие реально существующие связи между целями. В качестве примера того, к чему может это привести, вспомним, как в недалеком прошлом организовывалась работа некоторых отраслей, например, железнодорожного транспорта. Показателем эффективности этой отрасли были тонно-километры. При этом часто эти тонно-километры набирались за счет абсолютно бесполезных и даже вредных с точки зрения эффективности всего хозяйства встречных перевозок. Или производство товаров, не находящих спроса? Но отрасли планы относительно своих показателей эффективности перевыполняли! Применительно к решению задачи информатизации фирмы независимое от главной цели рассмотрение такой проблемы, как «повышение качества делопроизводства», может привести к выводу о необходимости замены, например, PC/AT 486, используемой только для написания текстов, на Pentium 200, что несмотря на очень незначительное повышение качества делопроизводства приведет к снижению степени достижения цели «повышение рентабельности фирмы» вследствие неоправданных в данном случае затрат на приобретение этих более совершенных ПЭВМ. Какова же сложность иерархий целей? Ни одному человеку не под силу учесть большое (разрастающееся) количество связей при принятии решений. Однако современные ПК справляются с такого рода задачами. Построение в полном объеме иерархии достаточно трудоемко и дорого. Однако – такой труд оправдывается соответствующим возникновением база знаний для решения не одной управленческой задачи. Она может и должна использоваться для решения многих задач. Для этого предусматриваются средства «подвязки» вариантов решений (альтернатив) различных задач к иерархии целей. Можно, например, иерархию целей, построенную для выбора варианта информатизации, использовать для определения наилучшего в отношении той же главной цели варианта организации рекламной кампании, выбора типа продукции и т. д. Здесь изменится лишь содержание проектов и их связи с подцелями. Рассмотренные методы поддержки решений основаны на определении коэффициентов влияния экспертами или ЛПР. Однако разработаны методы поддержки принятия решений, адаптирующиеся к предыдущему опыту ЛПР. Необходимость такого подхода возникает при периодическом решении однотипных задач. Например: руководителю, нужно выбрать одного из нескольких кандидатов и назначить его на вакантную должность, причем на подобных должностях уже работает несколько сотрудников. Система предложит проранжировать этих сотрудников по степени успешности работы. Как правило, это не составит труда, хотя в большинстве случаев будет затруднительно четко сформулировать критерий, по которому провели ранжирование. Далее система предложит сформулировать частные критерии (профессионально важные качества), которые, по мнению руководителя, определяют успешность работы на данной должности (например, добросовестность, обязательность, эрудиция, трудолюбие, коммуникабельность и т. д.) и проранжировать уже работающих на данной должности сотрудников по каждому из этих частных критериев. На основании этого система рассчитывает коэффициенты важности этих частных критериев, которые несут информацию о вашем опыте в принятии решения по этому кадровому вопросу. Далее следует проранжировать кандидатов на должность по каждому из частных критериев, и система, используя эти оценки и ранее рассчитанные коэффициенты значимости частных критериев, определит рейтинг каждого из кандидатов и предложит самого достойного кандидата. Нетрудно видеть, что эта же система поможет выбрать тип сетевого оборудования, исполнителя работ по информатизации фирмы и т. д. Можно отметить, что такая система является личным инструментом, поэтому в качестве частных критериев можно использовать и конфиденциальные.

Исходя из изложенных принципов поддержки решений, представляется целесообразным определить три класса СППР в зависимости от сложности решаемых задач и областей применения.

СППР первого класса, обладающие наибольшими функциональными возможностями, предназначены для применения в органах государственного управления высшего уровня (администрация президента, министерства) и органах управления больших компаний (совет директоров корпорации) при планировании крупных комплексных целевых программ для обоснования решений относительно включения в программу различных политических, социальных или экономических мероприятий и распределения между ними ресурсов на основе оценки их влияния на достижение основной цели программы. СППР этого класса являются системами коллективного пользования, базы знаний которых формируются многими экспертами - специалистами в различных областях знаний.

СППР второго класса являются системами индивидуального пользования, базы знаний которых формируются непосредственным пользователем. Они предназначены для использования государственными служащими среднего ранга, а также руководителями малых и средних фирм для решения оперативных задач управления.

СППР третьего класса являются системами индивидуального пользования, адаптирующимися к опыту пользователя. Они предназначены для решения часто встречающихся прикладных задач системного анализа и управления (например, выбор субъекта кредитования, выбор исполнителя работы, назначение на должность и пр.). Такие системы обеспечивают получение решения текущей задачи на основе информации о результатах практического использования решений этой же задачи, принятых в прошлом. Кроме того, системы этого класса предназначены для использования в торговых предприятиях, торгующих дорогими товарами длительного пользования, в качестве средства «интеллектуальной рекламы», позволяющего покупателю выбрать товар на основе своего опыта применения товаров аналогичного назначения. В компьютерном мире происходят удивительные события: люди одного поколения стали свидетелями стремительного прогресса в области вычислительной техники и программного обеспечения - от «мастодонтов» типа «Урал» и программирования вычислительных задач в машинных кодах до современных ПЭВМ и конструирования сложных информационных систем в объектно-ориентированной среде. Такой же прогресс наблюдается и в прикладном программном обеспечении. По-видимому, в недалеком будущем обычными пользователями будут повседневно широко применяться системы, являющиеся интеллектуальными помощниками, в частности, системы поддержки принятия решений. Такие системы уже реально существуют. Надо поверить в их возможности и преодолеть психологический барьер (как, отдать «ящику» святое право принимать решения?!). Это право всегда остается за человеком, просто система учтет ваши желания, обобщит мнения знающих людей и предложит оценки вариантов. (Ведь даже диктаторы всегда имели советников!) Тот, кто раньше начнет применять эти системы в повседневной деятельности (так же широко, как, например, Word), получит преимущество перед своими конкурентами.

1.3 Преимущества МБ ДОБАВИТЬ ХЗ ПОКА

2 Выбор среды для разработки

В связи с наблюдаемым в настоящее время стремительным развитием персональной вычислительной техники, происходит постепенное изменение требований, предъявляемых к языкам программирования. Все большую роль начинают играть интерпретируемые языки, поскольку возрастающая мощь персональных компьютеров начинает обеспечивать достаточную скорость выполнения интерпретируемых программ. А единственным существенным преимуществом компилируемых языков программирования является создаваемый ими высокоскоростной код. Когда скорость выполнения программы не является критичной величиной, наиболее правильным выбором будет интерпретируемый язык, как более простой и гибкий инструмент программирования.

В связи с этим, определенный интерес представляет рассмотрение сравнительно нового языка программирования Python, который был создан его автором Гвидо ван Россумом в начале 90-х годов.

Отличительные характеристики языка:

* очень низкий порог вхождения, уже после одного дня изучения можно начать писать простые программы;
* минималистичный язык, с небольшим количеством конструкций;
* краткий код;
* прекрасно подходит для создания программ-обёрток, поддерживается импорт Си-библиотек;
* существует большое количество реализаций: CPython (основная реализация); Jython (реализация для JVM); IronPython (CLR); PyPy;
* очень хорошая поддержка математических вычислений (библиотеки NumPy, SciPy);
* используется для обработки естественных языков (NLTK);
* большое количество развитых web-фреймворков (Django, TurboGear, CherryPy, Flask).

Язык можно рекомендовать всем, кто только начинает программировать, как первый язык программирования в жизни.

2.1Причины использования Python

2.1.1 Качество программного обеспечения

Для многих основное преимущество языка Python заключается в удобочитаемости, ясности и более высоком качестве, отличающими его от других инструментов в мире языков программирования. Программный код на языке Python читается легче, а значит, многократное его использование и обслуживание выполняется гораздо проще, чем использование программного кода на других языках сценариев. Единообразие оформления программного кода на языке Python облегчает его понимание даже для тех, кто не участвовал в его создании. Кроме того, Python поддерживает самые современные механизмы многократного использования программного кода, каким является объектно-ориентированное программирование (ООП).

2.1.2 Высокая скорость разработки

По сравнению с компилирующими или строго типизированными языками, такими как C, C++ и Java, Python во много раз повышает производительность труда разработчика. Объем программного кода на языке Python обычно составляет треть или даже пятую часть эквивалентного программного кода на языке C++ или Java. Это означает меньший объем ввода с клавиатуры, меньшее количество времени на отладку и меньший объем трудозатрат на сопровождение. Кроме того, программы на языке Python запускаются сразу же, минуя длительные этапы компиляции и связывания, необходимые в некоторых других языках программирования, что еще больше увеличивает производительность труда программиста.

2.1.3 Переносимость программ

Большая часть программ на языке Python выполняется без изменений на всех основных платформах. Перенос программного кода из операционной системы Linux в Windows обычно заключается в простом копировании файлов программ с одной машины на другую. Более того, Python предоставляет массу возможностей по созданию переносимых графических интерфейсов, программ доступа к базам данных, веб-приложений и многих других типов программ. Даже интерфейсы операционных систем, включая способ запуска программ и обработку каталогов, в языке Python реализованы переносимым способом.

2.1.4 Библиотеки поддержки

В составе Python поставляется большое число собранных и переносимых функциональных возможностей, известных как *стандартная библиотека*. Эта библиотека предоставляет массу возможностей, востребованных в прикладных программах, начиная от поиска текста по шаблону и заканчивая сетевыми функциями. Кроме того, Python допускает расширение как за счёт ваших собственных библиотек, так и за счет библиотек, созданных сторонними разработчиками. Из числа сторонних разработок можно назвать инструменты создания веб-сайтов, программирование математических вычислений, доступ к последовательному порту, разработку игровых программ и многое другое. Например, расширение NumPy позиционируется как свободный и более мощный эквивалент системы программирования математических вычислений Mathlab.

2.1.5 Интеграция компонентов

Сценарии Python легко могут взаимодействовать с другими частями приложения благодаря различным механизмам интеграции. Эта интеграция позволяет использовать Python для настройки и расширения функциональных возможностей программных продуктов. На сегодняшний день программный код на языке Python имеет возможность вызывать функции из библиотек на языке C/C++, сам вызываться из программ, написанных на языке C/C++, интегрироваться с программными компонентами на языке Java, взаимодействовать с такими платформами, как COM и .NET, и производить обмен данными через последовательный порт или по сети с помощью таких протоколов, как SOAP, XML-RPC и CORBA.

2.2 История создания

Разработка языка Python была начата в конце 1980-х годов сотрудником голландского института CWI Гвидо ван Россумом. Для распределённой ОС Amoeba требовался расширяемый скриптовый язык, и Гвидо начал писать Python на досуге, позаимствовав некоторые наработки для языка ABC (Гвидо участвовал в разработке этого языка, ориентированного на обучение программированию). В феврале 1991 года Гвидо опубликовал исходный текст в ньюсгруппе alt.sources. С самого начала Python проектировался как объектно-ориентированный язык.

Название языка произошло вовсе не от вида пресмыкающихся. Автор назвал язык в честь популярного британского комедийного телешоу 1970-х "Летающий цирк Монти Пайтона". Впрочем, всё равно название языка чаще ассоциируют именно со змеёй, нежели с передачей — пиктограммы файлов в KDE или в Microsoft Windows и даже эмблема на сайте python.org (до выхода версии 2.5) изображают змеиные головы.

Наличие дружелюбного, отзывчивого сообщества пользователей считается наряду с дизайнерской интуицией Гвидо одним из факторов успеха Python. Развитие языка происходит согласно чётко регламентированному процессу создания, обсуждения, отбора и реализации документов PEP (англ. *Python Enhancement Proposal*) — предложений по развитию Python.

3 декабря 2008 года, после длительного тестирования, вышла первая версия Python 3000 (или Python 3.0, также используется сокращение Py3k). В Python 3000 устранены многие недостатки архитектуры с максимально возможным (но не полным) сохранением совместимости со старыми версиями Python. На сегодня поддерживаются обе ветви развития (Python 3.x и 2.x)

2.2.1 Влияние других языков

Появившись сравнительно поздно, Python создавался под влиянием множества языков программирования:

* ABC — отступы для группировки операторов, высокоуровневые структуры данных (map) (Python фактически создавался как попытка исправить ошибки, допущенные при проектировании ABC);
* Modula-3 — пакеты, модули, использование else совместно с try и except, именованные аргументы функций (на это также повлиял Common Lisp);
* С, C++ — некоторые синтаксические конструкции (как пишет сам Гвидо ван Россум — он использовал наиболее непротиворечивые конструкции из С, чтобы не вызвать неприязнь у С-программистов к Python);
* Smalltalk — объектно-ориентированное программирование;
* Lisp — отдельные черты функционального программирования (lambda, map, reduce, filter и другие);
* Fortran — срезы массивов, комплексная арифметика;
* Miranda — списочные выражения;
* Java — модули logging, unittest, threading (часть возможностей оригинального модуля не реализована), xml.sax стандартной библиотеки, совместное использование finally и except при обработке исключений, использование @ для декораторов;
* Icon — генераторы.

Большая часть других возможностей Python (например, байт-компиляция исходного кода) также была реализована ранее в других языках

2.3 Функциональные возможности

С точки зрения функциональных возможностей Python можно назвать гибридом. Его инструментальные средства укладываются в диапазон между традиционными языками сценариев (такими как Tcl, Scheme и Perl) и языками разработки программных систем (такими как C, C++ и Java). Python обеспечивает простоту и непринужденность языка сценариев и мощь, которую обычно можно найти в компилирующих языках. Превышая возможности других языков сценариев, такая комбинация делает Python удобным средством разработки крупномасштабных проектов. Ниже приводится список основных возможностей, которые есть в арсенале Python:

*Динамическая типизация*

Python сам следит за типами объектов, используемых в программе, благодаря чему не требуется писать длинные и сложные объявления в программном коде. В действительности, в языке Python вообще отсутствуют понятие типа и необходимость объявления переменных. Так как программный код на языке Python не стеснен рамками типов данных, он автоматически может обрабатывать целый диапазон объектов.

*Автоматическое управление памятью*

Python автоматически распределяет память под объекты и освобождает ее ("сборка мусора"), когда объекты становятся ненужными. Большинство объектов могут увеличивать и уменьшать занимаемый объем памяти по мере необходимости.

*Модульное программирование*

Для создания крупных систем Python предоставляет такие возможности, как модули, классы и исключения. Они позволяют разбить систему на составляющие, применять ООП для создания программного кода многократного пользования и элегантно обрабатывать возникающие события и ошибки.

*Встроенные типы объектов*

Python предоставляет наиболее типичные структуры данных, такие как списки, словари и строки, в виде особенностей, присущих самому языку программирования. Эти типы отличаются высокой гибкостью и удобством. Например, встроенные объекты могут расширяться и сжиматься по мере необходимости, могут комбинироваться друг с другом для представления данных со сложной структурой.

*Встроенные инструменты*

Для работы со всеми этими типами объектов в составе Python имеются мощные и стандартные средства, включая такие операции, как конкатенация (объединение коллекций), получение срезов (извлечение части коллекции), сортировка, отображение и многое другое.

*Библиотеки утилит*

Для выполнения более узких задач в состав Python также входит большая коллекция библиотечных инструментов, которые поддерживают практически все, что только может потребоваться, – от поиска с использованием регулярных выражений до работы в сети. Библиотечные инструменты языка Python – это то место, где выполняется большая часть операций.

*Утилиты сторонних разработчиков*

Python – это открытый программный продукт и поэтому разработчики могут создавать свои предварительно скомпилированные инструменты поддержки задач, решить которые внутренними средствами невозможно.

2.4 Прикладное применение Python

Python используется не только отдельными пользователями, он также применяется компаниями для создания продуктов, приносящих настоящую прибыль. Например:

− Компания Google широко использует Python в своей поисковой системе и оплачивает труд создателя Python.

− Служба коллективного использования видеоматериалов YouTube в значительной степени реализована на языке Python.

− Популярная программа BitTorrent для обмена файлами в пиринговых сетях (peer-to-peer) написана на языке Python.

− Популярный веб-фреймворк App Engine от компании Google использует Python в качестве прикладного языка программирования.

− Такие компании, как EVE Online и Massively Multiplayer Online Game (MMOG), широко используют Python в своих разработках.

− Мощная система трехмерного моделирования и создания мультипликации Maya поддерживает интерфейс для управления из сценариев на языке Python.

− Такие компании, как Intel, Cisco, Hewlett-Packard, Seagate, Qualcomm и IBM, используют Python для тестирования аппаратного обеспечения.

Такие компании, как Industrial Light & Magic, Pixar и другие, используют Python в производстве анимационных фильмов.

− NASA, Los Alamos, Fermilab, JPL и другие используют Python для научных вычислений.

− iRobot использует Python в разработке коммерческих роботизированных устройств.

− ESRI использует Python в качестве инструмента настройки своих популярных геоинформационных программных продуктов под нужды конечного пользователя.

− NSA использует Python для шифрования и анализа разведданных.

− В реализации почтового сервера IronProt используется более 1 миллиона строк программного кода на языке Python.

− Проект "ноутбук каждому ребенку" (One Laptop Per Child, OLPC) строит свой пользовательский интерфейс и модель функционирования на языке Python.

Универсальная природа языка обеспечивает возможность его применения в самых разных областях. Фактически с определенной долей уверенности можно утверждать, что Python так или иначе используется практически каждой достаточно крупной организацией, занимающейся разработкой программного обеспечения, – как для решения краткосрочных тактических задач, так и для разработки долгосрочных стратегических проектов.

2.5 Преимущество Python перед другими языками высокого уровня

* Имеет более широкие возможности, чем Tcl. Язык Python поддерживает "программирование в целом", что делает его применимым для разработки крупных систем.
* Имеет более четкий синтаксис и более простую архитектуру, чем Perl, что делает программный код более удобочитаемым, простым в сопровождении и снижает вероятность появления ошибок.
* Проще и удобнее, чем Java. Python – это язык сценариев, а Java унаследовала сложный синтаксис от таких языков программирования, как C++.
* Проще и удобнее, чем C++, но нередко он не может конкурировать с C++, поскольку, будучи языком сценариев, Python предназначен для решения другого круга задач.
* Более мощный и более переносимый, чем Visual Basic. Открытая природа Python также означает, что нет какой-то отдельной компании, которая его контролирует.
* Более удобочитаемый и более универсальный, чем PHP. Иногда Python используется для создания веб-сайтов, но он способен решать гораздо более широкий круг задач, от управления роботами до создания анимационных фильмов.
* Более зрелый и имеет более ясный синтаксис, чем Ruby. В отличие от Ruby и Java, объектно-ориентированный стиль программирования является необязательным в Python – он не вынуждает использовать ООП в проектах, где этот стиль неприменим.

Обладает динамическими особенностями таких языков, как SmallTalk и Lisp, но имеет более простой и традиционный синтаксис, доступный как для разработчиков, так и для конечных пользователей настраиваемых систем.

2.6 Реализации Python

CPython является основной, но не единственной реализацией языка программирования Python. Существуют также следующие реализации:

* Jython — реализация Python, использующая JVM в качестве среды исполнения. Позволяет прозрачно использовать Java-библиотеки.
* PyS60 — реализация языка для смартфонов фирмы Nokia на платформе Series 60.
* IronPython — Python для .NET Framework и Mono. Компилирует Python программы в MSIL, таким образом предоставляя полную интеграцию с .NET-системой.
* Stackless — также написанная на Си реализация Python. Это не полноценная реализация, а патчи к CPython. Предоставляет расширенные возможности многопоточного программирования и значительно большую глубину рекурсии.
* Python for .NET — ещё одна реализация Python для .NET. В отличие от IronPython эта реализация не компилирует Python код в MSIL, а только предоставляет интерпретатор, написанный на C#. Позволяет использовать .NET-сборки из Python кода.
* PyPy — реализация Python, написанная на Python. Позволяет легко проверять новые возможности. В PyPy кроме стандартного CPython включены возможности Stackless, Psyco. В проект интегрированы возможности анализа Python кода и трансляция в другие языки и байткоды виртуальных машин.
* python-safethread — версия CPython без GIL, что позволяет одновременно исполнять Python потоки на всех доступных процессорах. Внесены также некоторые другие изменения.
* Unladen Swallow — начатый Google проект по разработке высокоэффективного, максимально совместимого с CPython JIT-компилятора, а на базе LLVM. Согласно планам по развитию Python, планировалось перенести исходный код Unladen Swallow в CPython в версии 3.3. Но PEP-3146 был отменён в связи с отсутствием интереса к Unladen Swallow со стороны Google, основного спонсора разработки.
* tinypy — минималистическая версия Python. Часть возможностей CPython не реализована.

Micro Python — эффективная реализация Python 3 для встроенных систем с малым объёмом оперативной памяти.

2.7 Недостатки

**Низкое быстродействие**

Классический Python, как и многие другие интерпретируемые языки, не применяющие, например, JIT-компиляторы, имеют общий недостаток — сравнительно невысокую скорость выполнения программ. Сохранение байт-кода позволяет интерпретатору не тратить лишнее время на перекомпиляцию кода модулей при каждом запуске, в отличие, например, от языка Perl. Кроме того, существует специальная JIT-библиотека psyco, позволяющая ускорить в выполнение программ (однако приводящая к увеличению потребления оперативной памяти). Эффективность psyco сильно зависит от архитектуры программы.

Существуют реализации языка Python, вводящие высокопроизводительные виртуальные машины (ВМ) в качестве бэк-энда компилятора. Примерами таких реализаций может служить PyPy, базирующийся на LLVM; более ранней инициативой является проект Parrot. Ожидается, что использование ВМ типа LLVM приведёт к тем же результатам, что и использование аналогичных подходов для реализаций языка Java, где низкая вычислительная производительность в основном преодолена.

Множество программ/библиотек для интеграции с другими языками программирования предоставляют возможность использовать другой язык для написания критических участков.

**Невозможность модификации встроенных классов**

По сравнению с Ruby и некоторыми другими языками, в Python отсутствует возможность модифицировать встроенные классы, такие, как int, str, float, list и другие, что, однако, позволяет Python потреблять меньше оперативной памяти и быстрее работать. Ещё одной причиной введения такого ограничения является необходимость согласования с модулями расширения. Многие модули (в целях оптимизации быстродействия) преобразуют Python-объекты элементарных типов к соответствующим Си-типам вместо манипуляций с ними посредством Си-API. Также это избавляет от многих потенциальных ошибок при неконтролируемом динамическом переопределении встроенных типов

3 Телеграм(мессенджер)

Мессенджер – (IM – Instant Messenger) – это программа мобильное приложение или веб сервис для мгновенного обмена сообщениями. Чаще всего под мессенджером понимают программу, в которой вы пишите сообщения и где вы их читаете. Однако, за каждой такой программой входит понятие «мессенджер».

Telegram – бесплатный кроссплатформенный мессенджер для смартфонов и других устройств, позволяющий обмениваться текстовыми сообщениями и медиафайлами различных форматов. Используются проприетарная сервераня часть с закрытым кодом, работающая на мощностях нескольких компаниях США и Германии, финансируемых Павлом Дуровым в объёме порядка 12 млн долларов США ежегодно, и несколько клиентов с открытым кодом, в том числе под GNU GPL. Количество активных пользователей сервиса на февраль 2016 года составляло более 100 млн человек, а количество ежедневно пересылаемых сообщений достигло 10 миллиардов на август 2015.

3.1 История создания телеграмма

Проект создан Павлом Дуровым, основателем социальной сети «ВКонтакте». В интервью The New York Times Павел рассказал, что первоначальная идея приложения пришла ему ещё в 2011-м, когда к его двери приходили спецназовцы. Когда последние всё-таки ушли, Дуров сразу же написал своему брату Николаю. Тогда же он и осознал, что у него нет безопасного способа коммуникации с братом. Сервис построен на технологии шифрования переписки MTProto, разработанной братом Павла — Николаем. Сам «Телеграм» изначально был экспериментом принадлежащей Павлу компании Digital Fortress с целью протестировать MTProto на больших нагрузках.

14 августа 2013 года представлен первый клиент Telegram для устройств на платформе iOS.

22 августа 2013 года один из участников конкурса Durov’s Android Challenge написал и выложил в открытый доступ первое приложение для операционной системы Android, совместимое с Telegram, (использует тот же протокол MTProto).

В октябре у проекта открылся веб-сайт и была представлена официальная версия Telegram под Android с открытым исходным кодом (GPL2). Предыдущая версия программы доступна под названием «Unofficial Telegram S».

7 ноября 2013 года появились сторонние клиенты сервиса для Windows и OS X с ограниченным функционалом. Также был разработан концепт веб-версии клиента.

В ноябре у программы насчитывалось, по данным «TJournal», около 1 миллиона установок.

В январе 2014 года вышла неофициальная веб-версия Webogram от бывшего разработчика ВКонтакте Игоря Жукова.

21 июля 2014 года в App Store появилось приложение Telegram HD для iPhone и iPad, которое загрузила компания Telegram Messenger LLP.

Новое приложение получило специальную версию для Apple iPad, улучшило поддержку видео и фотографий высокого разрешения, добавило возможность пересылки анимированных изображений в формате gif. На официальном сайте мессенджера в качестве клиента для iOS указано именно это приложение.

15 октября 2014 года в Telegram была добавлена поддержка псевдонимов, по которым возможно связываться с пользователями, даже не зная их телефонного номера, а также запущен веб-клиент.

2 января 2015 года в Telegram была добавлена поддержка стикеров. Изначально в приложении 14 стикеров, но любой пользователь может модифицировать их или добавить свои собственные. В отличие от многих приложений, в Telegram стикеры полностью бесплатные.

В феврале 2016 года один из создателей Telegram Павел Дуров заявил, что мессенджером пользуются уже более 100 миллионов человек, при этом сервис доставляет около 15 миллиардов сообщений ежедневно. Еще в сентябре 2015 года Telegram передавал 12 миллиардов посланий в день.

В апреле 2016 года стало известно, что в мае 2015 года корпорация Google рассматривала возможность покупки мессенджера за более чем 1 млрд долларов США.

В мае 2016 года появилась возможность редактирования отправленных сообщений. Внести изменения возможно в течение двух суток с момента отправки. В этом случае в сообщении появится специальная метка.

22 ноября 2016 года разработчиками был запущен проект Telegraph — блог-платформа, бесплатный издательский инструмент, который позволяет создавать публикации, обзоры, вставлять фотографии и все виды embedded-кода. Telegraph — гибрид блог-платформы, мессенджера и платишера (подобного Medium), с концепцией анонимных имиджборд. Проект Павла Дурова, основателя социальной сети «ВКонтакте»

3 января 2017 года один из разработчиков добавил возможность удаления своих отправленных сообщений. После удаления отправителем сообщения, собеседник не сможет увидеть удаленное сообщение.

В марте 2017 года В. Д. Соловей со ссылкой на анонимный источник сообщил, что российские спецслужбы получили доступ к сообщениям пользователей и их архиву за три года. Павел Дуров назвал это заявление уткой.

15 мая 2017 года стало известно о том, что декстопная версия Telegram получила возможность совершать звонки.

16 мая 2017 года администрация Telegram заявила, что не будет предоставлять информацию российским госорганам.

3.2Технология

Для мессенджера был создан протокол MTProto, предполагающий использование нескольких протоколов шифрования. При авторизации и аутентификации используются алгоритмы RSA-2048, DH-2048 для шифрования, при передаче сообщений протокола в сеть они шифруются AES с ключом, известным клиенту и серверу. Также применяются криптографические хеш-алгоритмы SHA-1 и MD5.

Безопасность от перехвата пересылаемых сообщений со стороны сервера Telegram обеспечивается лишь в режиме «секретных» чатов (Secret Chats), доступном с 8 октября 2013 года. Этот режим реализует шифрование, при котором лишь отправитель и получатель обладают общим ключом (end-to-end шифрование), с применением алгоритма AES-256 в режиме IGE (англ. Infinite Garble Extension) для пересылаемых сообщений. В отличие от обычного режима, сообщения в секретных чатах не расшифровываются сервером, история переписки сохраняется лишь на тех двух устройствах, на которых был создан чат.

При обмене файлами можно как отправить файлы с устройства, так и искать медиаконтент в интернете, в том случае, если используется мобильная версия для iOS или Android. Размер передаваемых файлов ограничен 1,5 Гб. Программа использует систему докачки файлов после обрыва связи.

Имеется возможность организовывать мультичаты до 200 участников, начиная с ноября 2015 года, супергруппы до 1000 участников, с 14 марта 2016 — супергруппы до 5000 участников.

3.3 Основные возможности приложения Telegram

* Программа поддерживает большую часть распространенных операционных систем, такие как Windows, Linux, Mac OS;
* доступна для мобильных устройств, работающих на Android, Windows Phone и iOS;
* автоматическая синхронизация программы между всеми устройствами пользователя;
* применение специальной системы шифрования для передачи и хранения личной информации;
* самостоятельная очистка всей истории переписки по истечению выбранного промежутка времени;
* возможность сохранять мультимедийные объекты в «облаке»;
* открытый доступ к API для создания собственных интегрируемых модулей.

Начало работы с месседжером Телеграмм подразумевает регистрацию мобильного аккаунта в приложении. Единственное обязательное условие для авторизации – указание номера мобильного телефона.

3.4 Преимущества и недостатки использования Telegram

В пользу использования этой программы можно отметить следующие характеристики:

* бесплатное пользование приложением на неограниченный срок;
* высококачественное шифрование информации;
* мгновенная доставка пересланных данных;
* отсутствие рекламных блоков и микро-транзакций;
* возможность самостоятельной настройки и регулирования приложения «под себя».

Скромные, но тем не менее, недостатки:

* отсутствие интерфейса с переводом на русский язык;
* исходный код серверной части закрыт для пользователей.

Несмотря на это, Телеграмм остается одним из самых удобных, а главное безопасных способов передачи информации в сети. Ведь разработчики приложения доказали, что обеспечить конфиденциальность в интернете возможно.

3.5 Секретный чат

Мессенджер Telegram обладает некоторыми отличительными положительными чертами, относящимися к безопасности пользования данной программой. Одной из основных черт считается секретный чат в Telegram.

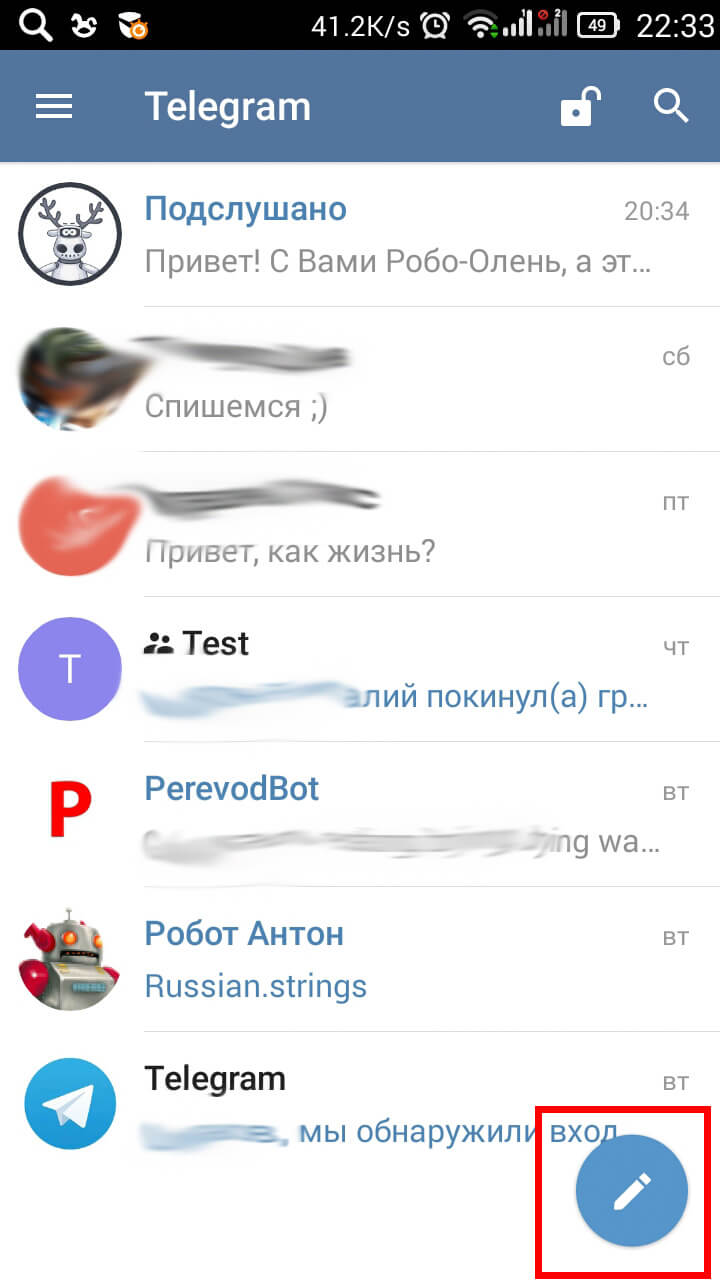
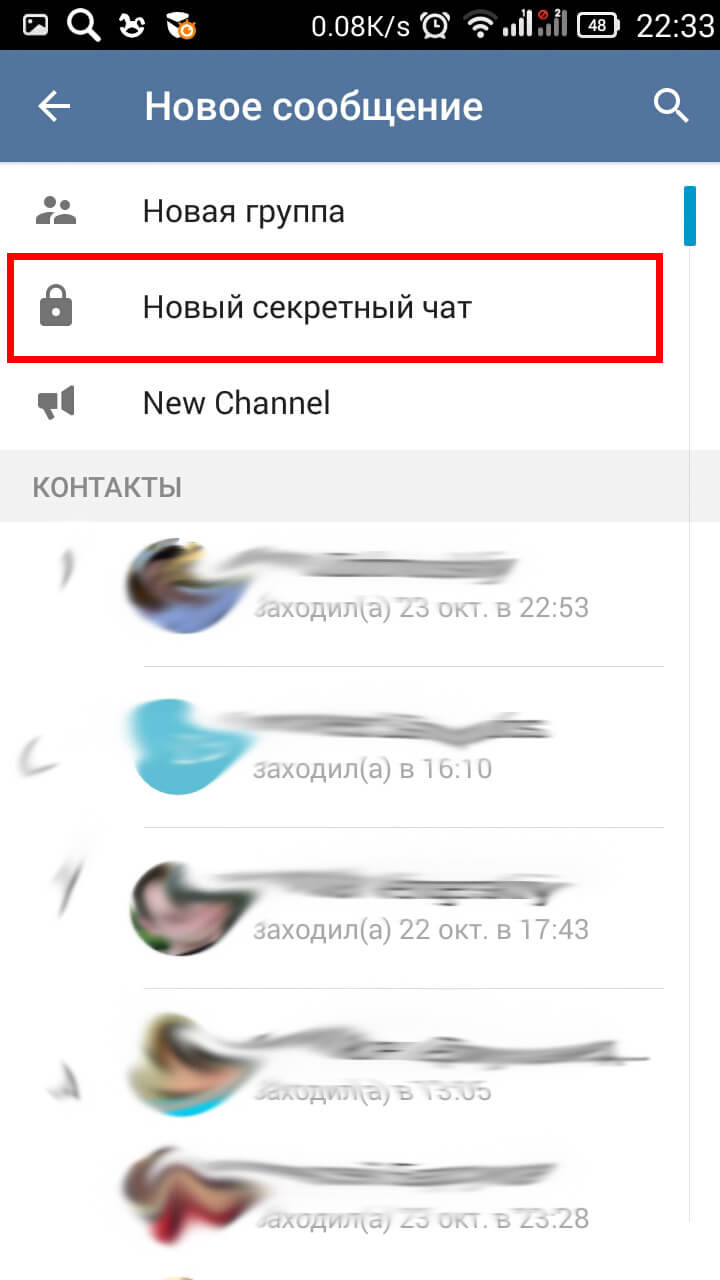
Если подробнее ознакомиться с этой функцией, то можно увидеть ряд особенностей её применения. Переписка не подвергается сохранению на сервере и моментально удаляется сразу после отправления сообщения. Остальные пользователи, естественно, не могут прочитать её. А для повышения мощности защиты реализуется конечное шифрование сообщения. Таким образом, отправленное сообщение остается доступным лишь для его адресата и адресанта. Применяемые системой алгоритмы гарантируют высокий уровень безопасности, а расшифровать чужую личную переписку не могут даже сами разработчики мессенджера.

Для каждого сообщения или отдельной переписки можно подобрать и установить отдельный таймер самоликвидации, который составляет минимум две секунды и максимум семь дней. Как только срок завершается, сообщение самостоятельно удаляется.

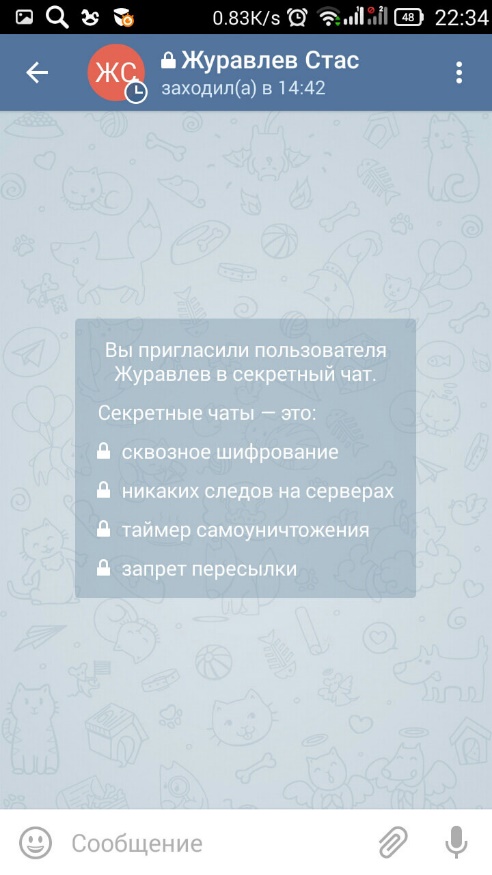
Получатель читает новое сообщение и через несколько секунд оно попросту исчезает. При проявлении смекалки, конечно, можно успеть сделать снимок экрана или скриншот. Но в таком случае программа зафиксирует скриншот и перешлет отправителю уведомление об этой операции.

3.5.1Создание секретного чата

Для этого понадобиться:

* при помощи нажатия кнопки справа внизу перейти в меню[](http://telegram-online.ru/wp-content/uploads/2015/10/sekretnyj-chat-v-telegram-1.jpg)
* затем выбрав вкладку **«Новый секретный чат»[](http://telegram-online.ru/wp-content/uploads/2015/10/sekretnyj-chat-v-telegram-2.jpg)**
* после чего, в списке контактов необходимо выбрать контакт, с которым Вы хотели бы начать секретный чат.

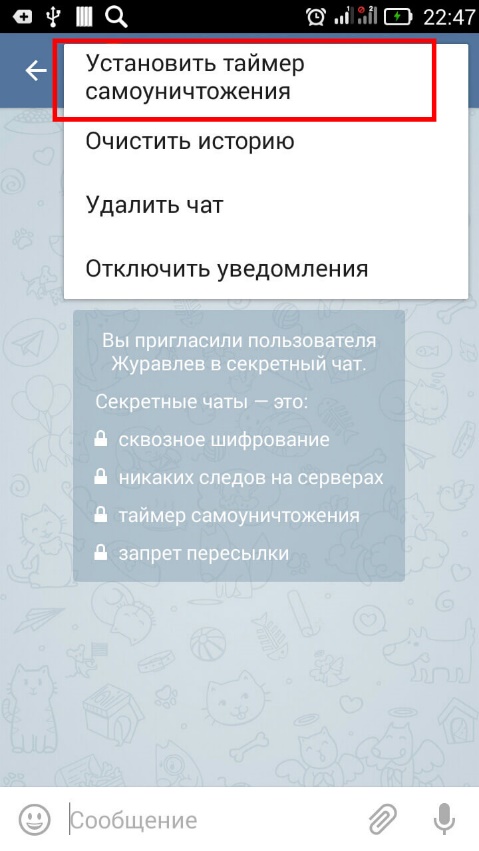
Вот и все, секретный чат создан:

[](http://telegram-online.ru/wp-content/uploads/2015/10/sekretnyj-chat-v-telegram-3.jpg)

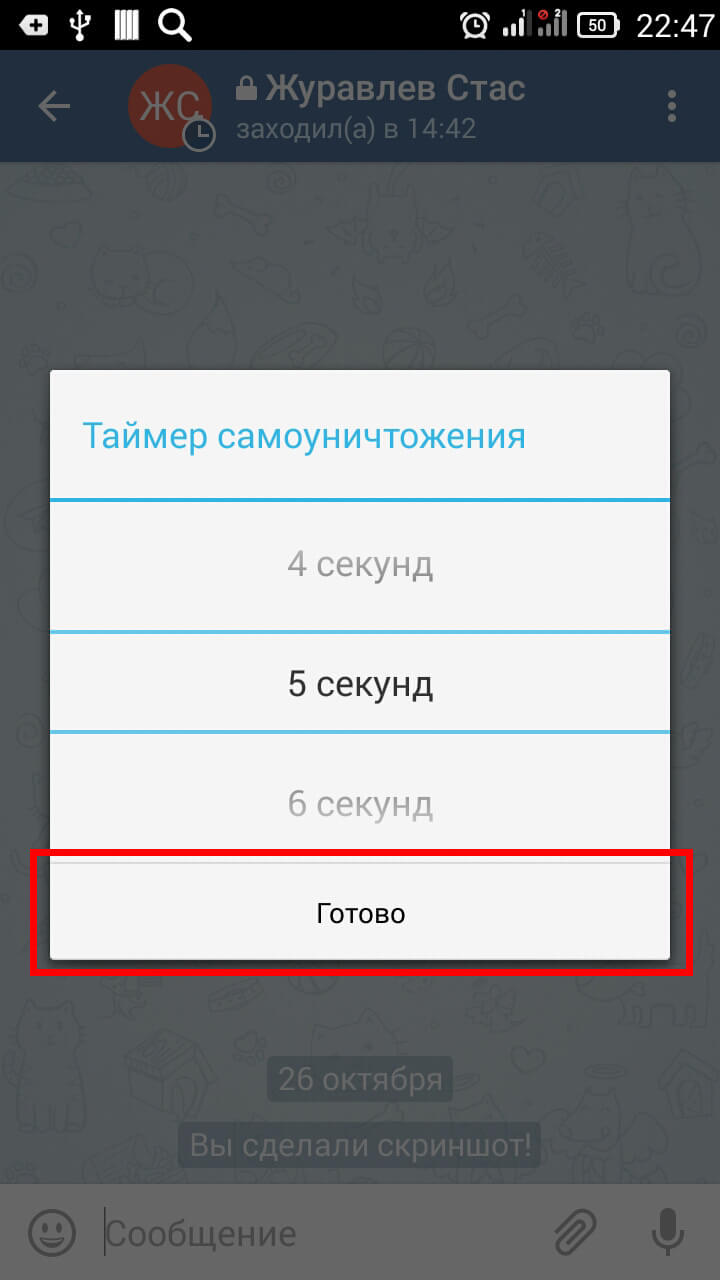
**Как установить таймер самоуничтожения сообщений**

Что бы установить таймер самоликвидации сообщений в секретном чате Телеграмма, необходимо:

* в самом чате, зайдите в меню, которое расположено в правом верхнем углу в виде трех точек, и выберите пункт **«Установить таймер самоуничтожения»**

**[](http://telegram-online.ru/wp-content/uploads/2015/10/sekretnyj-chat-v-telegram-4.jpg)**

* с помощью прокрутки установите необходимое время и нажмите кнопку **«Готово»**

**[](http://telegram-online.ru/wp-content/uploads/2015/10/sekretnyj-chat-v-telegram-5.jpg)**

Существуют некоторые особенности создания таких чатов в Telegram:

• секретный чат в Телеграмм привязывается к определенному устройству, то есть, когда переписка начата на смартфоне, то только на нем можно будет продолжить её; • при выходе из аккаунта секретный чат не сохраняется; • с одним другом можно создать сразу несколько чатов.

ТУТ ДОЛЖНА БЫТЬ КАРТИНКА ТЕЛЕГРАММА ЛИБО РЕКЛАМКА)

<https://texterra.ru/blog/pochemu-stoit-poprobovat-telegram-obzor-messendzhera.html> про телеграм топчик просто глянуть как будет сказка может быть,

<http://telegram-online.ru/sekretnyj-chat-v-telegram.html> ссылка про секретный чатик телеграммма

Источники

<http://www.tadviser.ru/index.php/%D0%A1%D1%82%D0%B0%D1%82%D1%8C%D1%8F:%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BF%D0%BE%D0%B4%D0%B4%D0%B5%D1%80%D0%B6%D0%BA%D0%B8_%D0%BF%D1%80%D0%B8%D0%BD%D1%8F%D1%82%D0%B8%D1%8F_%D1%80%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D0%B9#.D0.9F.D1.80.D0.B5.D0.B8.D0.BC.D1.83.D1.89.D0.B5.D1.81.D1.82.D0.B2.D0.B0>

<http://bourabai.ru/tpoi/dss.htm#1>

<http://elar.urfu.ru/bitstream/10995/1676/5/1335843_schoolbook.pdf>

<http://pro-spo.ru/erp/1816-dss>

<https://www.youtube.com/watch?v=dDwTKNsrv54>

<http://mirznanii.com/a/111739/sozdanie-sistem-podderzhki-prinyatiya-resheniy>

<http://www.treko.ru/show_article_763>

<https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%B0%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0_%D0%B8%D0%B5%D1%80%D0%B0%D1%80%D1%85%D0%B8%D0%B9>