**Теория по-гуриновски**

1. Принципы дискретизации и квантования для оцифровки мультимедийной информации.

Мультимедийная информация – графическая и звуковая, воспринимается органами чувств человека только в аналоговом виде и только в определенных пределах по графике, звуку и скорости воспроизведения чередующихся кадров в видео или анимации.

Это обстоятельство лежит в основе принципов оцифровки мультимедийной информации и записи ее в графические, аудио и видео-файлы.

Оцифровка мультимедийной информации подразумевает запись в цифровой форме в файлы соответствующего формата для хранения и последующего воспроизведения на компьютере аналоговой информации.

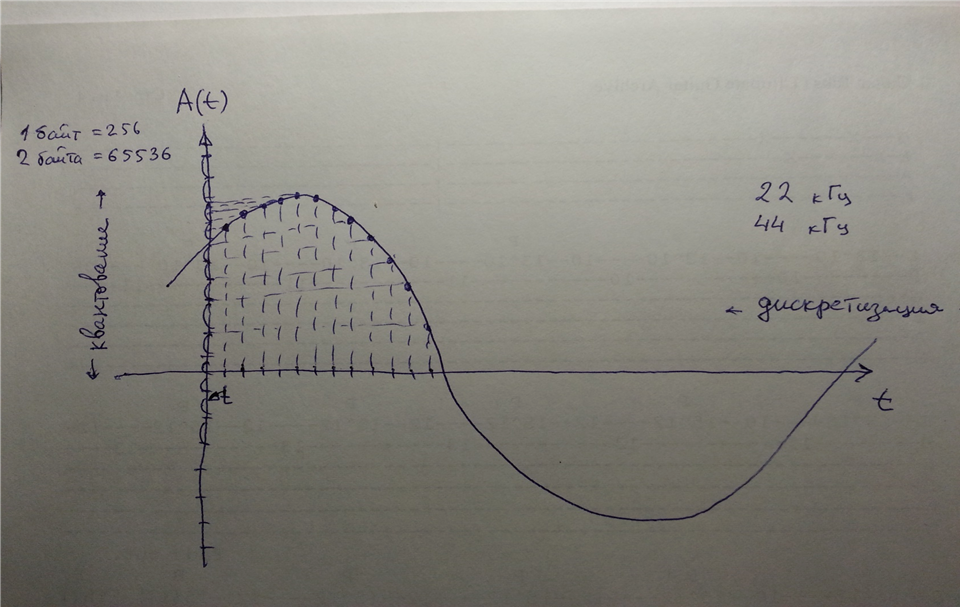
частота дискретизации звука должна быть не менее 40 кГц, поскольку максимальная частота звука для восприятия человеком составляет примерно 20 кГц.

При оцифровке звука (Digital sampling)при выполнении этих этапов звуковой платой компьютера, как правило, для них назначаются параметры в следующих пределах:

* **Дискретизация** (Sampling rate) – от 11 кГц до 44 кГц
* **Квантование** (Sampling size) – от 8 бит (речь) до 16 бит (музыка)

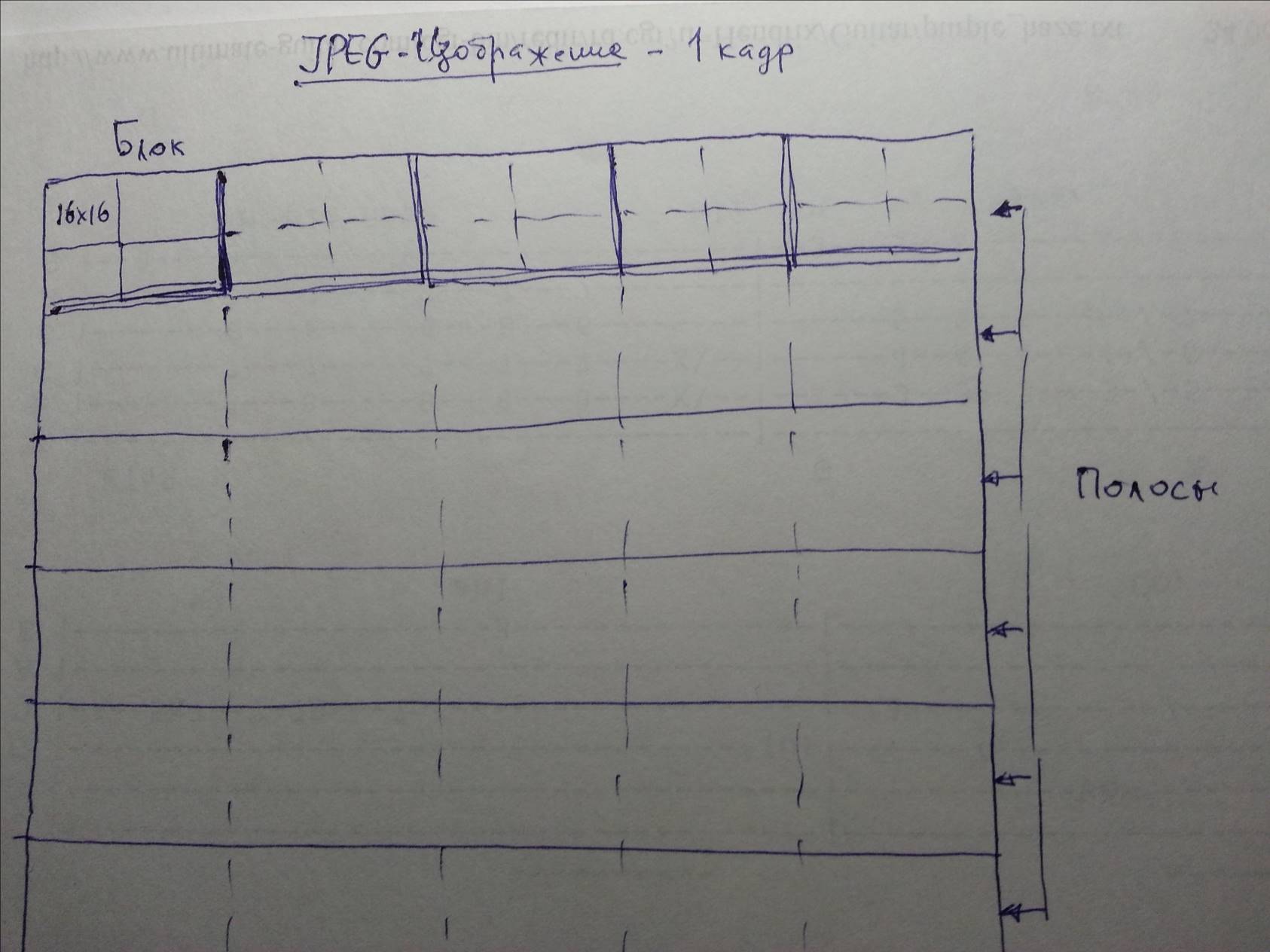
Типы звуковых форматов: **WAV** (несжатый), **MP3** (сжатый по определенному алгоритму) и др.

**Оцифровка звуковой волны**

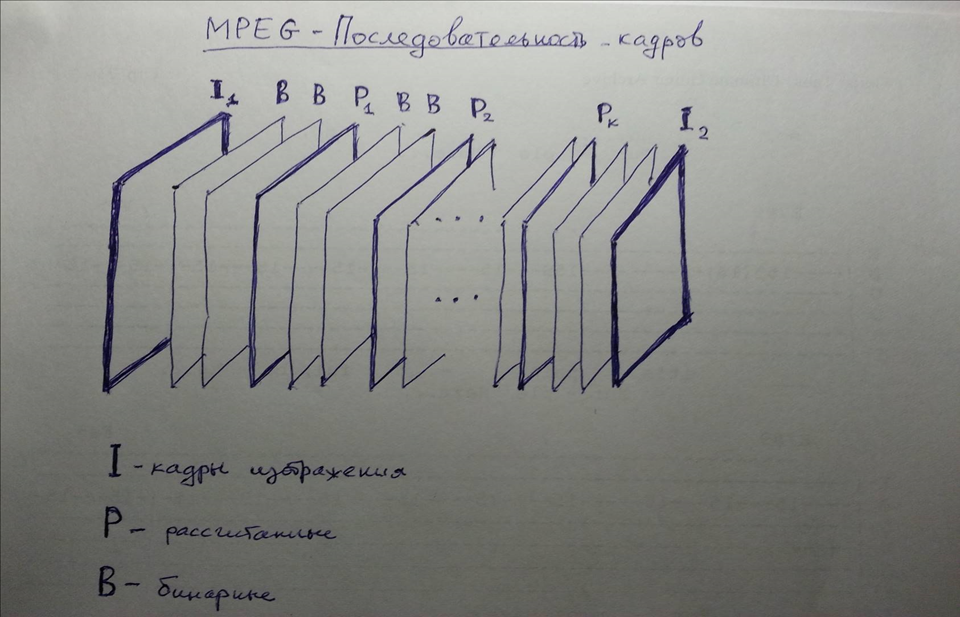


2. Общая структура форматов сжатия мультимедийной информации: JPEG для графики и MPEG для видео.

**JPEG-формат**



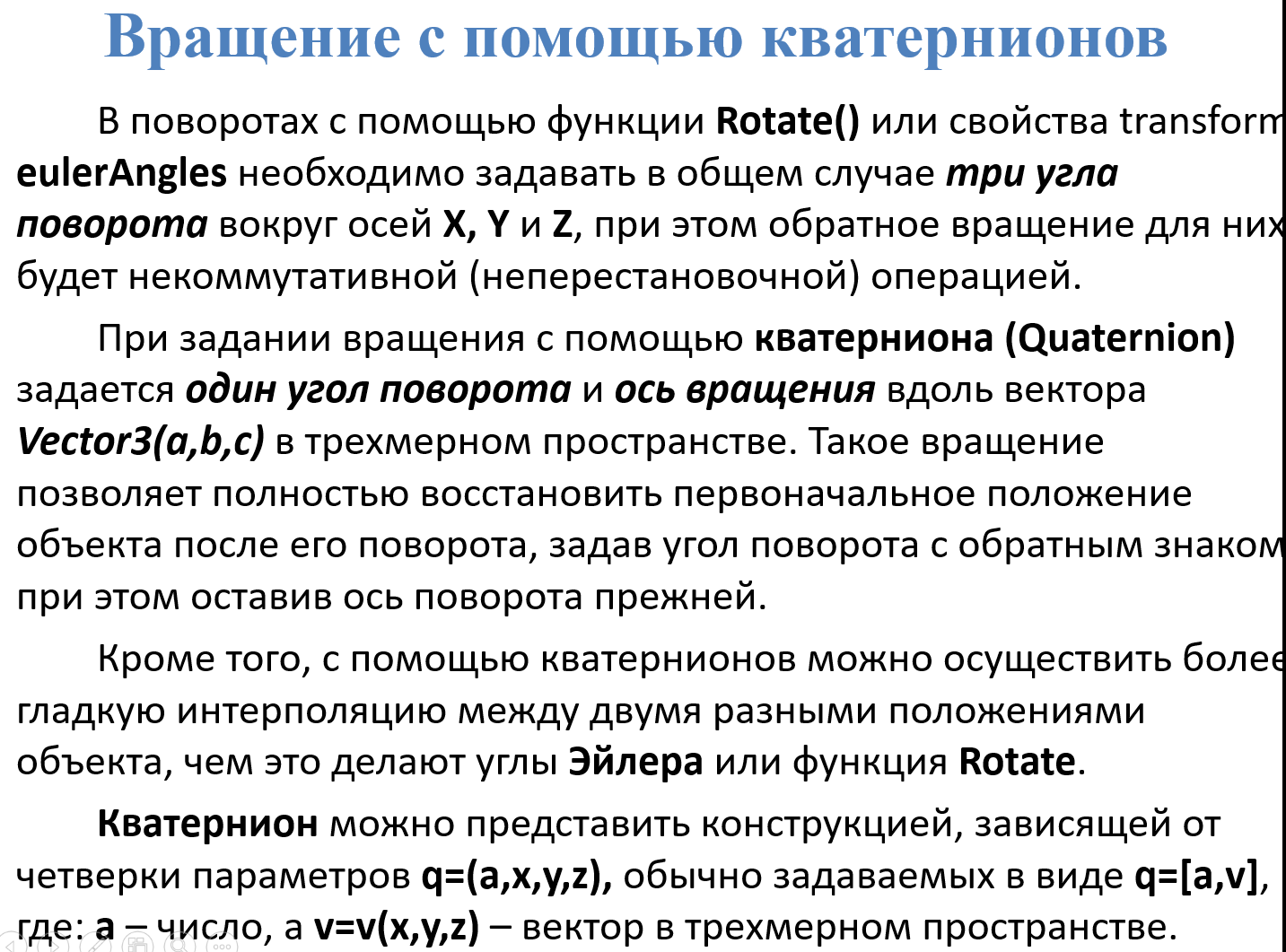
**MPEG-формат  
MPEG-последовательность: I - Intra, P - Predicted, B – Bidirectional**

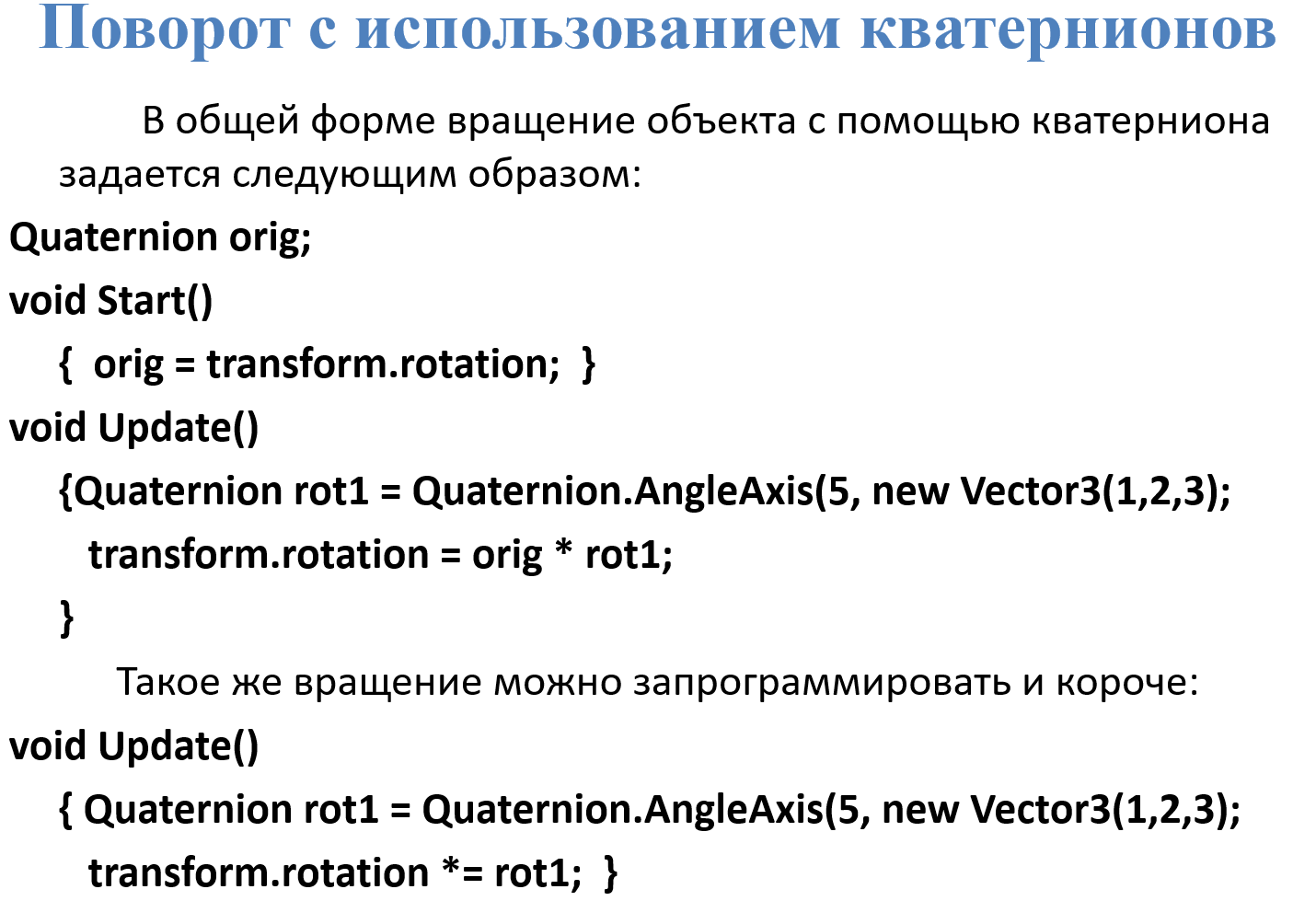


3. Правила создания и использования объектов «Кнопка» в среде Adobe Animate.



4. Понятие о кватернионах, использование кватернионов для программирования поворотов 3D-объектов.





5. Представление базы знаний в виде семантической сети, объекты, субъекты и действия.

В предложениях текстового учебного материала обучающей или инфор-мационной системы необходимо выделять субъект, объект и действие.

Под **СС** для **базы знаний** понимают ***граф***, в вершинах которого находятся ***информационные единицы - объекты*** (сущности), а дуги характеризуют ***отношения - связи*** (действия) между ними, которые интерпретируют эти сущности в заданной предметной области.

СС связывает ***объекты, субъекты и действия***. Основным связующим элементом сети является ***действие***, которое передает ***отношение*** между ***субъектом*** и ***объектом***. Действие связывает **субъект** (носитель действия) и **объект** (на кого/что направлено действие или посредством чего оно реализуется).

**Электрод состоит из металла или из ионов в растворе.**

электрод

из ионов в растворе

из металла

*состоит*

*состоит*

Две связи фрагмента СС, представленного на рисунке, могут быть записаны в виде следующего **списка дуг**, состоящего из двух элементов списка **е1**и **е2**:



При этом таблица, предназначенная для хранения списка дуг СС, будет состоять из ***трех полей*** (столбцов): двух информационных единиц и типа связи.

6. Принцип членения предложений в тексте на триады для записей в базу знаний.

**ЭДС характеризуется температурным коэффициентом.**

Это предложение можно представить следующей схемой:

**Блок**

**существительного**

**ЭДС**

**Блок**

**глагола**

**характеризуется**

**Блок**

**существительного**

**температурным коэффициентом**

*Первый блок* *существительного* можно определить как ***субъектный*** блок: блок субъекта − того, кто или что производит действие.

*Второй блок существительного* можно определить как ***объектный*** блок: блок объекта – того, на кого или на что направлено действие (посредством кого или чего оно реализуется).

Такое представление соответствует стандартной схеме в виде **триады**:

**субъект – действие – объект**

7. Алгоритм генерации семантической сети из сложных текстов информационной системы.

Типы сложных предложений в тексте:

1. **сложносочиненное** – содержит несколько грамматических основ из подлежащего и сказуемого, каждую основу следует рассматривать как самостоятельный блок;
2. содержит **одно подлежащее и несколько однородных сказуемых** – следует рассматривать каждое сказуемое как отдельный блок, который дополняется общим подлежащим;
3. содержит **уточнения в скобках** – предложение без уточнений рассматривается как отдельный блок, каждое уточнение также рассматривается как отдельный блок, который в зависимости от содержания скобок дополняется сказуемым;
4. содержит **одинаковые отношения** одного подлежащего с несколькими дополнительными членами и отношение одного дополнительного члена с несколькими подлежащими.

Принципы дополнения неполных блоков

После разбиения сложного предложения на семантические блоки необходимо дополнить **неполные блоки** до самостоятельных простых предложений (в зависимости от неполноты полученного блока):

* в качестве **сказуемого** может использоваться глагол из предыдущего семантического блока (однородные дополнительные члены) или глагол, зависящий от контекста (уточнения в скобках);
* в качестве **подлежащего** применяется последнее существительное предыдущего блока (уточнение в скобках) или подлежащее предыдущего блока (однородные сказуемые).

В результате дополнения каждый семантический блок будет представлять собой **самостоятельное простое предложение**.

**ПРИМЕР ОБРАБОТКИ ПРЕДЛОЖЕНИЯ СО СКОБКАМИ**

***«Электрод состоит из металла M z- (восстановленная форма системы) и ионов M z+ в растворе (окисленная форма системы)»*.**

***«Электрод состоит из металла M z-.»***

***Металл является восстановленной формой системы.»***

***«Ион является окисленной формой системы.»***

***«Электрод состоит из ионов M z+ в растворе.»***

8. Правила формирования записей базы знаний в текстовый массив на языке JavaScript.

Таким образом, для занесения записей в **Базу знаний** в соответствии с правилами построения семантической сети на основе триад из исходного сложного предложения со скобками в тексте должны быть составлены 4 простых предложения:

***«Электрод состоит из металла M z-»***

***«Электрод состоит из ионов M z+ в растворе»***

***«Металл M z- является восстановленной формой системы»***

***«Ион M z+ является окисленной формой системы»***

Из этих предложений складывается фрагмент семантической сети БАЗЫ ЗНАНИЙ:

5 вершин – **ключевые понятия** (объекты и субъекты): электрод (черный), металл Mz- (серый), ион Mz+ в растворе (белый), восстановленная форма системы (желтый), окисленная форма системы (зеленый);

2 связки – **отношения** (сказуемые)*:* ***состоит*** – красный, ***является*** – синий

9. Использование шаблонов из регулярных выражений для сказуемого и подлежащего в вопросе.

В общем случае, в вопросе подлежащее и сказуемое могут находится в неподходящих падежах и склонениях, в которых они хранятся Базе знаний. => используем регулярные выражения.

Таким образом, в целом процедура поиска точечного ответа на заданный к Базе знаний вопрос будет состоять из следующих этапов:

*На первом этапе* при помощи ***регулярного выражения-сказуемого*** проверяются ячейки **второго** столбца двумерного массива базы знаний.

*На втором этапе* при помощи ***регулярного выражения-подлежащего*** – последовательно проверяются на соответствие ячейки **первого** и **третьего** столбцов базы знаний, т. к. русский язык допускает перестановку блоков подлежащих в предложениях.

Наконец, *на третьем этапе*, если совпадений по подлежащему и сказуемому вместе не найдено, то выполняется поиск ***только по подлежащему***, чтобы учесть возможность использования в вопросе сказуемого, ***синонимичного*** (совпадающему по смыслу, но не совпадающему по корню) записанному в базе знаний.

10. Общая блок-схема алгоритма получения точечного ответа на вопрос к базе знаний.

Текущее слово в вопросе – сказуемое?

Начало

Конец

Разбиение вопроса на отдельные слова

Цикл по массиву слов в вопросе

Цикл по записям базы знаний

Формирование регулярных выражений из подлежащего и сказуемого в вопросе

Добавление в ответ предложения, сформированного конкатенацией ячеек текущей записи

Текущая запись подходит?

11. Структура массива псевдоокончаний, используемого для поиска сказуемых в вопросе.



12. Содержание массива «черный список» для исключения подлежащих при поиске сказуемого.

В используемой методике обработки вопросов на основе псевдо-окончаний сказуемых приходится учитывать слова в БАЗЕ ЗНАНИЙ, ко-торые не являются сказуемыми, но имеют совпадающие со сказуемыми окончания и которые будут распознаваться при анализе как сказуемые по ошибке.

Для исключения такой ситуации необходимо создать отдельный од-номерный текстовый массив со словами в базе знаний с совпадающими окончаниями и использовать его при анализе содержимого базы знаний для исключения таких слов из рассмотрения при обработке вопросов в режиме диалога.

Например, при обработке псевдоокончаний сказуемых: -на, -ны, -ут, -ен, -ет в такой текстовый массив должны быть включены слова, которые имеют те же окончания и встречаются в обрабатываемом текстовом мас-сиве Базы знаний (список элементов массива должен быть заполнен по всему текстовому массиву Базы знаний, созданному по исходному тек-сту информационной системы):

**var blacklist =**

["замена", "замены", "атрибут", "маршрут", "член", "нет"];

13. Структура функции определения сказуемых в вопросе по псевдоокончаниям.



14. Алгоритм работы функции семантического генератора ответов на вопрос к базе знаний.

1. Подготовка вопроса: нормализация текста, обработка разделителей, разбиение на слова, нормализация слов.

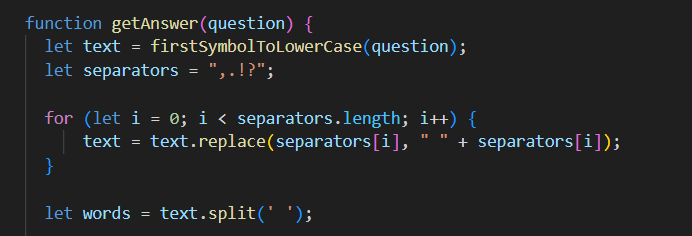
2. Поиск окончания, преобразование окончания

3. Создание шаблонов поиска(для сказуемого и для подлежащего)

4. Сопоставление с БЗ

5. Формирование ответа

15. Процедура подготовки и разбиения текста вопроса на отдельные слова в функции семантического генератора.



1 шаг: приводим все слова в вопросе к нижнему регистру

2 шаг: объявляем строку с знаками препинания, с которым можно столкнуться в вопросе

3 шаг: перед каждым знаком препинания добавляем пробел

4 шаг: разбиваем вопрос, на слова по пробелам.

Предложение «Как дела?», будет преобразовано в [“как”, “дела”, “?”]

16. Процедура поиска сказуемого в вопросе в функции семантического генератора.

3. Теперь будем решать задачу поиска сказуемого в вопросе, для чего выполним цикл, перебирающем все слова в предложении вопроса, записанные в массив **words**. При этом текущее слово в массиве будет считаться сказуемым, если обладает характерным для сказуемых **псевдоокончанием**.

//перебор слов в массиве слов из вопроса

**for (var i = 0; i < words.length; i++)**

**{**

//поиск номера псевдоокончания с использованием //вспомогательной функции **getEnding()** сзаписью //найденного номера в переменную **ending**

**var ending = getEnding(words[i]);**

Если псевдоокончание будет найдено, а это эквивалентно возвращаемому значению функции getEnding() в виде номера в массиве, отличного от -1, то это и есть сказуемое в вопросе, а подлежащее в вопросе будет следовать сразу после него:

**if (ending >= 0)**

**{**

//замена псевдоокончания на набор возможных окончаний,

//хранящихся во втором столбце массива

**words[i] =**

**words[i].substring(0,words[i].length-endings[ending][0].length)**

**+ endings[ending][1];**

**}**

Таким образом, найденное в массиве **words[i]** сказуемое в вопросе заменяется выражением **words[i],** но уже с набором соответствующих ему псевдоокончаний из второго столбца массива **endings[]**, что в конечном итоге превращает найденное в вопросе сказуемое в соответствующий ему шаблон – регулярное выражение для этого сказуемого.

Так например, любые из возможных окончаний сказуемых в вопросе для глаголов 1-го спряжения -ет, -ут, -ют будут заменены на общий шаблон с перечислением возможных вариантов строки в виде «(ет|ут|ют)».

17. Процедура формирования регулярного выражения для сказуемого в функции семантического генератора.

      let ending = getEndingIndex(words[i]); //Получение окончания слова

      if (ending >= 0) { // окончание найдено

          words[i] = words[i].substring(0, words[i].length - endings[ending][0].length) + endings[ending][1]; // замена окончания

          let predicate = new RegExp(".\*" + words[i] + ".\*"); // регулярка для поиска сказуемого

          if (endings[ending][0] == endings[ending][1]) { //учитываем что сказуемое может быть их 2 слов

              predicate = new RegExp(".\*" + words[i] + " " + words[i + 1] + ".\*");

              i++;

          }

18. Процедура формирования регулярного выражения для подлежащего в функции семантического генератора.

          let subjectReg = words.slice(i + 1).join(".\*"); // для извлечения всех слов, следующих за текущим словом (и сказуемым).

          if (subjectReg.length > 3) {

              let subject = new RegExp(".\*" + subjectReg + ".\*");

19. Процедура отбора из Базы знаний триады для получения ответа на основе метода test() в функции семантического генератора и формирования ответа на вопрос.

 for (let j = 0; j < knowledge.length; j++) {

                  if (

                      predicate.test(knowledge[j]?.[1]?.toLowerCase()) && // соответствует ли глагол из бз глаголу из вопроса

                      (subject.test(knowledge[j]?.[0]?.toLowerCase()) ||

                          subject.test(knowledge[j]?.[2]?.toLowerCase()))

                  ) {

                      answer += firstSymbolToUpperCase( // форм. ответ: объедин. всё в одно предложение

                          knowledge[j][0] + " " + knowledge[j][1] + " " + knowledge[j][2]

                      );

                      result = true;

                  }

              }

              if (!result) {

                  for (let j = 0; j < knowledge.length; j++) { //если ответ не нашли, то ищем только по подлежащему

                      if (

                          subject.test(knowledge[j]?.[0]?.toLowerCase()) ||

                          subject.test(knowledge[j]?.[2]?.toLowerCase())

                      ) {

                          answer += firstSymbolToUpperCase(

                              knowledge[j][0] + " " + knowledge[j][1] + " " + knowledge[j][2]

                          );

                          result = true;

                          break;

                      }

                  }

              }

20. Назначение и общая структура функции dialog\_window() для создания диалогового окна

Функция dialog\_window() предназначена для создания диалогового окна, которое позволяет пользователю задавать вопросы, поддерживает распознавание и синтез речи и сохраняет историю взаимодействия.

Структура:

1. Добавление разметки
2. Инициализация Яндекс апи для распознавания и синтеза речи:

Установка апи-ключа

Создание объекта для ввода текста голосом

function dialog\_window() {

  document.body.innerHTML +=

    "<div id='dialog' class='dialog' style='margin-left:-45px;'>" +

    "<div class='label' onclick='openDialog()'>Нажми, чтобы спросить!</div>" +

    "<div class='header'>История:</div>" +

    "<div class='history' id='history'></div>" +

    "<div class='question'><input id='Qdialog' placeholder='Введите вопрос'/> <br><button onclick='ask(\"Qdialog\")'>Спросить</button></div>" +

    "</div>";

  window.ya.speechkit.settings.apikey = "y0\_\_wgBEOzTiYQFGMHdEyCQ0dmBElqmIEgwjL3HEwN2xZBTvQJOmOut";

  var textline = new ya.speechkit.Textline("Qdialog", {

    onInputFinished: function (text) {

      document.getElementById("Qdialog").value = text;

    },

  });

}

21. Структура и стиль оформления общего блока div для диалогового окна.



22. Структура и стиль оформления блока div для полосы с кнопкой «Нажми, чтобы спросить» для запуска функции выдвижения диалогового окна на центр страницы.

*.dialog .label{*

*transform: rotate(270deg);*

*width:430px;*

*height:20px;*

*text-align:center;*

*color:#f00;*

*overflow:hidden;*

*display:inline-block;*

*margin-left:-26%;*

*margin-top:50%;*

*cursor:pointer;*

*}*

23. Структура функции openDialog() плавного выдвижения-закрытия диалогового окна поверх Web-страницы.



24. Структура и стиль оформления блока div для выдачи ответа на вопрос в окно истории ответов.

.dialog input, .dialog button, .dialog .history{

margin:2px 25px;

width:85%;

border:1px #666 solid;

border-radius: 5px;

font-size:20px;

}

.dialog .history .question {

margin:5px;

width:100%;

border-radius: 5px;

padding:5px;

margin-left:40px;

background:#99f;

}

25. Структура и стиль оформления блока div для поля формы input и кнопкой «Спросить» для запуска функции получения ответа из Базы знаний и выдачи его в блок ответа диалогового окна.

.dialog input, .dialog button, .dialog .history{

margin:2px 25px;

width:85%;

border:1px #666 solid;

border-radius: 5px;

font-size:20px;

}

.dialog .history .question {

margin:5px;

width:100%;

border-radius: 5px;

padding:5px;

margin-left:40px;

background:#99f;

}