Реферат

Это пример каркаса расчётно-пояснительной записки, желательный к использованию в РПЗ проекта по курсу РСОИ.

Дополняет краткое пособие по графике в Latex. Данный опус, как и более новые версии этого документа, можно взять по адресу (`http://sevik.ru/latex`). Минимально необходимые пакеты Latex, которые должны стоять: mathtext, amssymb, amsmath, icomma, longtable, graphicx, underscore, cmap, hyperref.

Текст в документе носит совершенно абстрактный характер.

Содержание

Глоссарий

Распределённый — Слово, которое нельзя употреблять. Но надо протестировать длинные строки в глоссарии.

Обозначения и сокращения

АИС — Автоматизированная информационная система. Но надо протестировать длинные строки в определениях.

Введение

Целью работы является создание всякой всячины. Для достижения поставленной цели необходимо решить следующие задачи:

— проанализировать существующую всячину;
— спроектировать свою, новую всячину;
— изготовить всякую всячину;
— проверить её работоспособность.

Вот так-то. А этот абзац вставлен для визуальной оценки отступа от перечня до следующего абзаца.

1 Аналитический раздел

В данном разделе анализируется и классифицируется существующая всячина и пути создания новой всячины. А вот отступ справа в 1 см. — это хоть и по ГОСТ, но ведь диагноз же...

1.1 Анализ того и сего

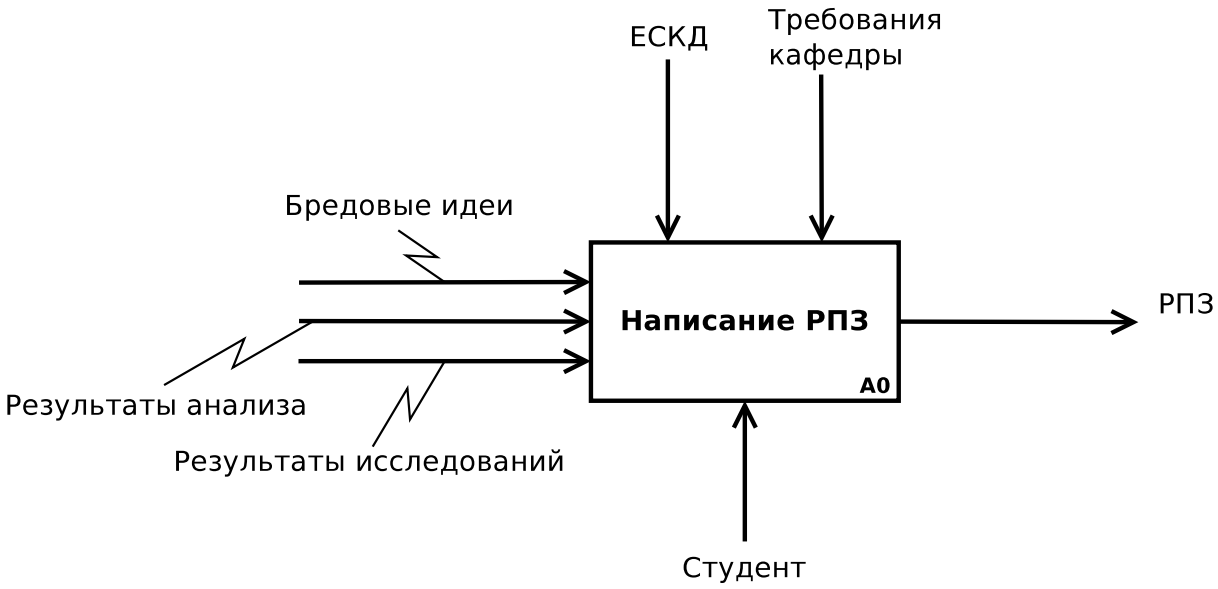

Рисунок 1.1 — Рисунок

В [1] указано, что...

Кстати, про картинки. Во-первых, для фигур следует использовать `[ht]`. Если и после этого картинки вставляются «не по ГОСТ», т.е. слишком далеко от места ссылки, — значит у вас в РПЗ **слишком мало текста**! Хотя и ужасный параметр `!ht` у окружения `figure` тоже никто не отменял, только при его использовании документ получается страшный, как в ворде, поэтому просьба так не делать по возможности.

1.2 Существующие подходы к созданию всячины

Известны следующие подходы...

а) Перечисление с номерами.

б) Номера первого уровня. Да, ГОСТ требует именно так — сначала буквы, на втором уровне — цифры. Чуть ниже будет вариант «нормальной» нумерации и советы по её изменению. Да, мне так нравится: на первом уровне выравнивание элементов как у обычных абзацев. Проверим теперь вложенные списки.

1) Номера второго уровня.
2) Номера второго уровня. Проверяем на длиииином-предлиииииииииииинной строке, что получается.... Сойдёт.

в) По мнению Лукьяненко, человеческий мозг старается подвести любую проблему к выбору из трех вариантов.

г) Четвёртый (и последний) элемент списка.

Теперь мы покажем, как изменить нумерацию на «нормальную», если вам этого захочется. Пара команд в начале документа поможет нам.

1) Изменим нумерацию на более привычную...
2) ... нарушим этим гост.
   а) Но, пожалуй, так лучше.

В заключение покажем произвольные маркеры в списках. Для них нужен пакет **enumerate**.

1. Маркер с арабской цифрой и с точкой.
2. Маркер с арабской цифрой и с точкой.
   I. Римская цифра с точкой.
   II. Римская цифра с точкой.

В отчётах могут быть и таблицы — см. табл. 1.1 и 1.2. Небольшая таблица делается при помощи **tabular** внутри **table** (последний полностью аналогичен **figure**, но добавляет другую подпись).

Таблица 1.1 — Пример короткой таблицы с длинным названием на много длинных-длинных строк

Тело	F	V	E	$F+V-E-2$
Тетраэдр	4	4	6	0
Куб	6	8	12	0
Октаэдр	8	6	12	0
Додекаэдр	20	12	30	0
Икосаэдр	12	20	30	0
Эйлер	666	9000	42	$+\infty$

Для больших таблиц следует использовать пакет **longtable**, позволяющий создавать таблицы на несколько страниц по ГОСТ.

Для того, чтобы длинный текст разбивался на много строк в пределах одной ячейки, надо в качестве ее формата задавать `p` и указывать явно ширину: в мм/дюймах (`110mm`), относительно ширины страницы (`0.22\textwidth`) и т.п.

Можно также использовать уменьшенный шрифт — но, пожалуйста, тогда уж во **всей** таблице сразу.

Таблица 1.2 — Пример длинной таблицы с длинным названием на много длинных-длинных строк

Вид шума	Громкость, дБ	Комментарий
Порог слышимости	0	
Шепот в тихой библиотеке	30	
Обычный разговор	60-70	
Звонок телефона	80	Конечно, это было до эпохи мобильников
Уличный шум	85	(внутри машины)
Гудок поезда	90	
Шум электрички	95	
Порог здоровой нормы	90-95	Длительное пребывание на более громком шуме может привести к ухудшению слуха
Мотоцикл	100	
Power Mower	107	(модель бензокосилки)
Бензопила	110	(Doom в целом вреден для здоровья)
Рок-концерт	115	
Порог боли	125	feel the pain
Клепальный молоток	125	(автор сам не знает, что это)
Порог опасности	140	Даже кратковременное пребывание на шуме большего уровня может привести к необратимым последствиям
Реактивный двигатель	140	
	180	Необратимое полное повреждение слуховых органов
Самый громкий возможный звук	194	Интересно, почему?..

2 Конструкторский раздел

В данном разделе проектируется новая всячина.

2.1 Архитектура всячины

Проверка параграфа. Вроде работает.

Вторая проверка параграфа. Опять работает.
Вот.

— Это список с «палочками».
— Хотя он и не по ГОСТ, кажется.

1) Поэтому для списка, начинающегося с заглавной буквы, лучше список с цифрами.

Формула 2.1 совершено бессмысленна.

$$a = cb \tag{2.1}$$

Окружение `cases` опять работает (см. 2.2), спасибо И. Короткову за исправления..

$$a = \begin{cases} 3x + 5y + z, \text{если хорошо} \\ 7x - 2y + 4z, \text{если плохо} \\ -6x + 3y + 2z, \text{если совсем плохо} \end{cases} \tag{2.2}$$

2.2 Подсистема всякой ерунды

Культурная вставка dot-файлов через утилиту dot2tex (рис. 2.1).

2.2.1 Блок-схема всякой ерунды

Кстати о заголовках

У нас есть и **subsubsection**. Только лучше её не нумеровать.

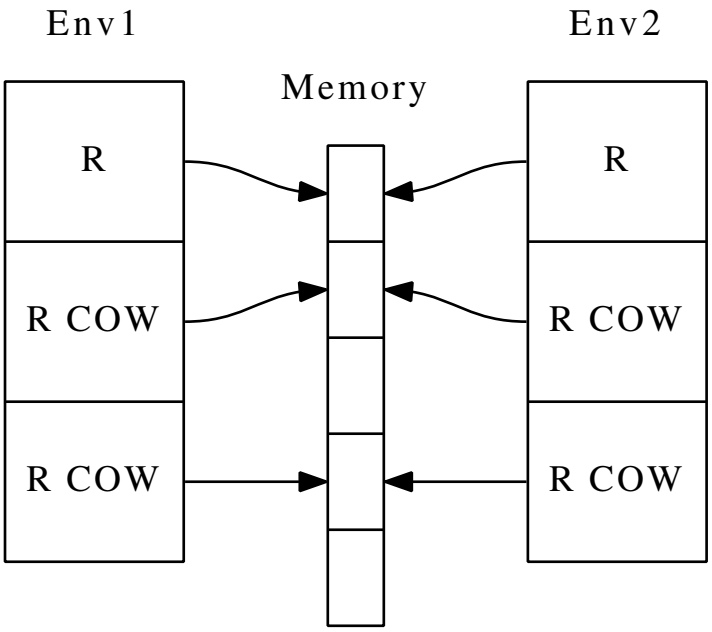

Рисунок 2.1 — Рисунок

3 Технологический раздел

В данном разделе описано изготовление и требование всячины. Кстати, в Latex нужно эскейпить подчёркивание (писать «`some\_function`» для **some_function**).

Для вставки кода есть пакет **listings**. К сожалению, пакет **listings** всё ещё работает криво при появлении в листинге русских букв и кодировке исходников utf-8. В данном примере он (увы) на лету конвертируется в koi-8 в ходе сборки pdf.

Есть альтернатива **listingsutf8**, однако она работает лишь с **\lstinputlisting**, но не с окружением **\lstlisting**

Вот так можно вставлять псевдокод (питоноподобный язык определен в **listings.inc.tex**):

Листинг 3.1 — Алгоритм оценки дипломных работ

```
def EvaluateDiplomas():
    for each student in Masters:
        student.Mark ← 5
    for each student in Engineers:
        if Good(student):
            student.Mark ← 5
        else:
            student.Mark ← 4
```

Еще в шаблоне определен псевдоязык для BNF:

Листинг 3.2 — Грамматика

```
  ifstmt → "if" "(" expression ")" stmt  |
           "if" "(" expression ")" stmt1 "else" stmt2
  number → digit digit *
```

В листинге 3.3 работают русские буквы. Сильная магия. Однако, работает только во включаемых файлах, прямо в TEX нельзя.

Листинг 3.3 — Пример (**test.c**)

```
#include <stdio.h>

int main()
{
    /* Комментарий на русском с пробелами */
    printf("Это строчка с пробелами и русскими буквами");

    return 0;
}
```

Можно также использовать окружение **verbatim**, если **listings** чем-то не устраивает. Только следует помнить, что табы в нём «съедаются». Существует так же команда **\verbatiminput** для вставки файла.

```
a_b = a + b; // русский комментарий
if (a_b > 0)
    a_b = 0;
```

4 Исследовательский раздел

В данном разделе описаны эксперименты, проводимые с разработанным программным обеспечением. Эксперименты проводятся с целью определения скоростных характеристик разработанного программного обеспечения и метода в целом, производится оценка точности обнаружения ситуаций гонок в программах в зависимости от их структуры.

4.1 Условия проведения экспериментов

Проведение экспериментов производилось в следующих условиях:

1) аппаратное обеспечение:
   а) AMD Turion(tm) X2 Ultra Dual-Core Mobile ZM-82 2,2GHz;
   б) 3096 Мб ОЗУ;
2) программное обеспечение:
   а) ОС Fedora GNU/Linux 20.0 3.11.10-301.fc20.i686;
   б) GCC 4.8.2;
   в) Python 2.7.5;
   г) gcc-python-plugin 0.12.

4.2 Исследование скоростных характеристик

Для проведения экспериментов по определению скоростных характеристик за основу была взята задача «читатели-писатели». Предполагается, что есть некоторый общий для всех потоков ресурс. Часть потоков получает к нему доступ только для чтения, а часть - для записи. При этом чтение может осуществляться одновременно из нескольких потоков. Код анализируемой программы представлен в листинге 4.1.

Листинг 4.1 — Код решения задачи "читатели-писатели"(**readers-writers.c**)

```
1  #include <stdio.h>
2  #include <pthread.h>
3
4  int buffer = 0;
5  int reader_count = 0;
6  pthread_mutex_t reader_count_mutex;
7  pthread_mutex_t reader_mutex;
8  pthread_mutex_t writer_mutex;
9
10 void* reader(void* args) {
11     while (1) {
12         pthread_mutex_lock(&reader_count_mutex);
13         if (reader_count == 0) {
14             pthread_mutex_lock(&reader_mutex);
15         }
```

```
16          reader_count += 1;
17          pthread_mutex_unlock(&reader_count_mutex);
18
19          printf("reader: %d\n", buffer);
20
21          pthread_mutex_lock(&reader_count_mutex);
22          reader_count -= 1;
23          if (reader_count == 0) {
24              pthread_mutex_unlock(&reader_mutex);
25          }
26          pthread_mutex_unlock(&reader_count_mutex);
27          sleep(1);
28      }
29      return NULL;
30  }
31
32  void* writer(void* args) {
33      while (1) {
34          pthread_mutex_lock(&reader_mutex);
35          pthread_mutex_lock(&writer_mutex);
36
37          buffer += 1;
38          printf("writer: %d\n", buffer);
39
40          pthread_mutex_unlock(&writer_mutex);
41          pthread_mutex_unlock(&reader_mutex);
42          sleep(1);
43      }
44      return NULL;
45  }
46
47  int main(int argc, char** argv) {
48      pthread_t thread1, thread2, thread3, thread4, thread5, thread6;
49
50      pthread_mutex_init(&reader_count_mutex, NULL);
51      pthread_mutex_init(&reader_mutex, NULL);
52      pthread_mutex_init(&writer_mutex, NULL);
53
54      // create readers
55      pthread_create(&thread1, NULL, reader, NULL);
56      pthread_create(&thread2, NULL, reader, NULL);
57      pthread_create(&thread3, NULL, reader, NULL);
58      pthread_create(&thread4, NULL, reader, NULL);
59
60      // create writers
61      pthread_create(&thread5, NULL, writer, NULL);
62      pthread_create(&thread6, NULL, writer, NULL);
```

```
63
64      pthread_join(thread1, NULL);
65      pthread_join(thread2, NULL);
66      pthread_join(thread3, NULL);
67      pthread_join(thread4, NULL);
68      pthread_join(thread5, NULL);
69      pthread_join(thread6, NULL);
70
71      pthread_destroy(&reader_count_mutex);
72      pthread_destroy(&reader_mutex);
73      pthread_destroy(&writer_mutex);
74
75      return 0;
76  }
```

4.3 Исследование точности

Бла-бла

Заключение

В результате проделанной работы стало ясно, что ничего не ясно...

Список использованных источников

1. *Пупкин, Василий*. LaTeXдля «чайников» / Василий Пупкин, А. Эйнштейн. — М., 2009.

Приложение А Картинки

Рисунок А.1 — Картинка в приложении. Страшная и ужасная.

Приложение Б Еще картинки

Рисунок Б.1 — Еще одна картинка, ничем не лучше предыдущей. Но надо же как-то заполнить место.