

АЗБУКА ХАЛТУРЩИКА-АРМАТУРЩИКА

разработка встраиваемых систем

основы бытовой автоматики,
систем управления и сбора данных

- © ruOpenWrt
- © HackSpace «Чебураторный завод»
- © Консорциум хоббитов России
- © Bill Collis (Part I)

Оглавление

Введение	10
I Введение в практическую электронику	11
An Introduction to Practical Electronics, Microcontrollers and Software Design © Bill Collis . . .	12
1 1 Введение в практическую электронику	13
2 2 Вводная электронная схема	14
3 3 Вводное конструирование печатной платы	15
4 4 Пайка, припой и паяльники	16
5 5 Введение в теорию электроники	17

6	6 Введение в электронику микроконтроллера	63	18
7	7 Входные цепи микроконтроллера	91	19
8	8 Обзор программирования	104	20
9	9 Введение в поток выполнения программы	112	21
10	10 Вводное программирование — использование подпрограмм	126	22
11	11 Вводное программирование — использование переменных	134	23
12	12 Основные дисплеи	161	24
13	13 Проект портативного аудиоусилителя на TDA2822M	174	25
14	14 Основы логического программирования	187	26
15	15 Разработка алгоритма — система сигнализации	202	27
16	16 Основы теории цепей постоянного тока	215	28
17	17 Основы планирования проекта	236	29
18	18 Пример дизайна системы — таймер клеевого пистолета	268	30
19	19 Основные интерфейсы и их программирование	273	31

20 20 Основы интерфейса аналого-цифрового преобразования	295	32
21 21 Основы проектирования системы	314	33
22 22 Основы проектирования системы — тайм-трекер	317	34
23 23 Основы вычислений времени	330	35
24 24 Основы строковых переменных	340	36
25 25 Силовые интерфейсы	353	37
26 26 Теория источников питания	370	38
27 27 Типичные вопросы тестирования 2011/12/13 годов	395	39
28 28 Расширенное программирование — массивы	397	40
29 29 Подтягивающие резисторы AVR	402	41
30 30 Дополнительно: подключение клавиатуры	403	42
31 31 Тонкости циклов Do-Loop & While-Wend	417	43
32 32 Подключение двигателя постоянного тока	423	44
33 33 Пример расширенной системы — будильник	452	45

34 34 Резистивный сенсорный экран 468	46
35 35 System Design Example – Temperature Controller 475	47
36 36 Advanced programming - state machines 478	48
37 37 Alarm clock project re-developed . 501	49
38 38 Advanced window controller student project 514	50
39 39 Alternative state machine coding techniques 524	51
40 40 Complex - serial communications . 526	52
41 41 Radio Data Communication 597	53
42 42 Introduction to I2C 617	54
43 43 Plant watering timer student project 631	55
44 44 Bike audio amplifier project 642	56
45 45 Graphics LCDs 648	57
46 46 GLCD Temperature Tracking Project 660	58
47 47 Interrupts 672	59

48	48 Timer/Counters .	692	60
49	49 LED dot matrix scrolling display project – arrays and timers .	698	61
50	50 Medical machine project – timer implementation	709	62
51	51 Multiple 7-segment clock project – dual timer action	715	63
52	52 The MAX 7219/7221 display driver IC's	739	64
53	53 Cellular Connectivity-ADH8066	744	65
54	54 Data transmission across the internet	778	66
55	55 Assignment – maths in the real world	816	67
56	56 SSD1928 based colour graphics LCD .	825	68
57	57 Traffic Light help and solution	865	69
58	58 Computer programming – low level detail	869	70
59	59 USB programmer - USBASP .	876	71
60	60 USBTinyISP programmer	877	72
61	61 C-Programming and the AVR	881	73

62	62 Object Oriented Programming (OOP) in CPP and the AVR	929	74
63	63 Current (2014) AVR development PCBS	953	75
64	64 Eagle - creating your own library	970	76
65	65 Practical Techniques .	979	77
66	66 CNC	990	78
67	67 Index .	1008	79
 II Основы электроники			80
68	68 Линейные схемы на пассивных элементах, основы электротехники		82
69	69 Симуляция и расчет схем в ngSPICE		83
70	70 KiCAD		84
	70.1 Отрисовка схем в KiCAD		84
	70.2 Библиотеки элементов		84
	70.3 Передача схемы в ngSPICE		84
71	71 Простейшие полупроводниковые элементы		85
	71.1 Оптоэлектроника		85
	71.2 Схемы на биполярных транзисорах		85

71.3 Схемы на на полевых транзисорах	85
72 Операционные усилители	86
73 Источники питания	87
73.1 Батарейное питание	87
73.2 Линейные стабилизаторы	87
73.3 Импульсные преобразователи на ШИМ-контроллерах	87
73.4 Цепи защиты и гашения кондуктивных помех	87
74 Цифровая электроника	88
75 Компьютерные интерфейсы	89
75.1 Поколение 90х: COM, LPT, ISA	90
75.1.1 Резервный программатор AVR “пять проводков”	90
75.2 Сеть CAN	90
75.3 Интерфейсные модули USB	90
75.3.1 Универсальный высокоскоростной конвертер FTDI FT232RL	90
75.3.2 JTAG-адаптер	90
75.3.3 Отладочный модуль CAN	90
75.4 Интерфейсные модули Ethernet	90
76 ПЛИС	91
77 Датчики	92

78 Электропривод и исполнительные устройства	93
 III Основы конструирования РЭС	 94
79 Пакеты моделирования на основе OpenFOAM	95
80 Обеспечение теплового режима	96
81 Электромагнитная совместимость	97
81.1 Кондуктивные помехи	97
81.2 Компонентные модели и оптимизация кабельной сети	97
 IV Технология РЭС	 98
82 Инструменты и оборудование	99
82.1 JTAG-адаптер	99
82.2 Отладочные платы	99
82.2.1 Arduino /Atmel Mega AVR8/	100
82.2.2 Cortex-Mx	100
82.2.3 CubieBoard /Cortex-A8 AllWinner A10/	101
82.2.4 Raspberry Pi /ARM11 BCM3032/	101
82.2.5 BlackSwift /MIPS/	101
82.2.6 VoCore /MIPS/	101
82.3 Монтажный инструмент	101

82.4	Измерительное оборудование	101
82.4.1	Тестер	101
82.4.2	Осциллограф	101
82.4.3	Логический анализатор	101
82.4.4	Генератор сигналов	101
82.4.5	Рыльцемертр	101
82.5	Электроинструмент	102
82.5.1	Дрель	102
83	Трассировка плат и подготовка производства в KiCAD	103
83.1	Технология ЛУТ (Лазерный УТюг)	103
83.2	Технология фоторезиста	103
83.3	Формат Gerber и подготовка промышленного производства	103
84	FreeCAD	104
84.1	Установка под Windows	106
84.2	Чертеж	107
84.3	Эскиз	107
84.4	Деталь	107
84.5	Сборка	107
84.6	Автогенерация конструкторской документации	107
84.7	Скрипты и пользовательские расширения	107
85	Эксплуатация станочного оборудования	108

86 Основы ЧПУ и цифрового производства	109
86.1 САМ-пакеты для FreeCAD	109
 V Основы теории систем автоматического управления	 110
87 Математический аппарат	111
87.1 Передаточная функция	111
87.2 Устойчивость САУ	111
87.3 Сети Петри	111
87.4 Автоматы Маркова	111
 88 Релейное управление	 112
 89 Пропорциональные САУ	 113
 90 ПИДн-регуляторы	 114
 VI Разработка ПО для встраиваемых систем	 115
91 Вспомогательные скрипты на языке Python	116
91.1 Установка под Windows	118
91.2 Запуск	119
91.3 Дополнительные материалы	124

92 Make: управление сборкой проектов	125
93 VCS: системы контроля версий	126
93.1 CVS	126
93.2 Subversion	126
93.3 Git	126
93.3.1 GitHub	126
94 Основы Си и C₊	127
94.0.2 Установка MinGW (win32)	127
94.1 Особенности C ₊ в embedded	127
95 LLVM и разработка собственных компиляторов	128
95.1 Лексический и синтаксический анализ	128
95.2 Применение flex/bison для разбора текстовых форматов данных	128
95.3 Компилятор Паскаля	128
96 Сборка кросс-компилятора GNU toolchain	129
 VII Микроконтроллеры Cortex-Mx	 130
97 Отладочные платы	131
97.1 STM32DISCOVERY /Cortex-M3 STM32F103/	131
97.2 STM32F4DISCOVERY /Cortex-M4 STM32F407/	131

ОГЛАВЛЕНИЕ	12
VIII Периферия	132
IX Встраиваемый emLinux	133
98 cross	134
99 BuildRoot	135
100 Особенности OpenWrt	136
101 Библиотека SDL	137
101.1 Реализация microGUI	137
102 Приложения для X Window	138
103 Программирование сетевых приложений	139
104 Сборка кросс-компилятора GNU мальтийским крестом	140
X IDE	141
105 ☹ ECLIPSE	144
105.1 Проверка орфографии	146
106 Code::Blocks	149

107(g) Vim	150
107.1 Установка под Windows	152
107.2 Выход из (g)Vim	155
107.2.1 Выход с автосохранением	155
107.3 Переход в режим редактирования	155
107.4 Переход в режим команд	155
107.5 Запись редактируемого файла	156
107.6 Перезагрузка файла	156
107.7 Отмена последних изменений (undo)	156
 XI Замечания для участников проекта	 157
107.8 Набор репозитория на GitHub	158
107.9 Верстка в L ^A T _E X	158
 XII Подготовка публикаций в L^AT_EX	 159
107.10 Установка MikTeX под Windows	162
107.11 Структура документа	162
107.11.1 Ваголовочный файл или блок	162
107.11.2 Стили документа	162
107.11.3 Пакеты	162
107.11.4 Автор и название	162
107.11.5 Верстка титульных страниц	162
107.11.6 Оглавление	162

107.1	Верстка слайдов	162
107.1	Список литературы и цитирование	162
107.1	Команды секционирования: часть, глава, раздел,.. . . .	165
107.1	Таблицы	165
107.1	Формулы	165
107.1	Перекрестные ссылки и гиперссылки	165
107.18	Истинги скриптов и текстовых данных	165
107.19	Подготовка иллюстраций	165
107.19.	Графики GNUPLOT	165
107.19.	Схемы и графы в GraphViz	165

XIII Куча 166

Список литературы 167

Введение

Первоначально этот материал задумывался как комплект документации к платам BlackSwift и VoCore, но постепенно превратился в толстенный учебник для студентов ВУЗов и научных работников по специализациям, связанным с применением цифровой электроники и компьютерной техники.

Большой упор был сделан на использование открытого некоммерческого программного обеспечения, с целью удешевления учебного процесса, уменьшения себестоимости ваших проектов¹, и стимулирования вашего участия в развитии этих программных пакетов.

Лицензия на эту книгу пока не выбрана, так что она пока просто пишется в духе OpenSource: любой может использовать ее часть, изменять или дополнять, до тех пор, пока не накладываются какие-либо административные, финансовые или юридические ограничения на распространение и развитие оригинальной версии или ее открытых форков.

Приглашаем всех желающих участвовать в развитии этого учебного пособия на форум [ruOpenWrt](#), нам нужна обратная связь по качеству материала, результаты тестирования на вас или ваших студентах, дополнения и замечания.

Мы признательны Bill Collis за разрешение использовать материалы его книги «[An Introduction to Practical Electronics, Microcontrollers and Software Design](#)» в русскоязычном варианте «Азбуки» (Часть I), и конечно он вполне заслуженно включен в основные соавторы этой книги.

¹ вряд ли ли у вас окажется лишняя пачка килобаксов на покупку пары коммерческих САПР, по крайней мере пока ваш стартап не взлетит в Top\$100K

Часть I

Введение в практическую электронику

Эта часть основана на книге:

An Introduction to Practical Electronics, Microcontrollers and Software Design

Second Edition, 01 May-2014

© Bill Collis

www.techideas.co.nz

Мы признательны автору за разрешение использовать материалы его книги в русскоязычном варианте «Азбуки», и конечно он вполне заслуженно включен в основные соавторы этой книги.

We are grateful to the author for permission to use materials of his book in the russian version of «Azbuka», and of course he was deservedly included in the main co-authors of this book.

From: Bill Collis <Bill.Collis@.....nz>

Date: 2014-11-24 0:53 GMT+04:00

Subject: Electronis Book

To: "dponyatov@gmail.com" <dponyatov@gmail.com>

Hi Dmitry

thanks for your email.

I am looking at the future of the book myself and thinking I will open source it. If you will only be in using it in Russian language then that is ok and you need to reference the original book.

Thanks

Bill

Глава 1

1 Введение в практическую электронику 13

Глава 2

2 Вводная электронная схема 15

Глава 3

3 Вводное конструирование печатной платы 26

Глава 4

4 Пайка, припой и паяльники 41

Глава 5

5 Введение в теорию электроники 49

Глава 6

6 Введение в электронику микроконтроллера 63

Глава 7

7 Входные цепи микроконтроллера 91

Глава 8

8 Обзор программирования 104

Глава 9

9 Введение в поток выполнения программы 112

Глава 10

10 Вводное программирование — использование подпрограмм 126

Глава 11

11 Вводное программирование — использование переменных 134

Глава 12

12 Основные дисплеи 161

Глава 13

13 Проект портативного аудиоусилителя на TDA2822M 174

Глава 14

14 Основы логического программирования 187

Глава 15

15 Разработка алгоритма — система сигнализации 202

Глава 16

16 Основы теории цепей постоянного тока 215

Глава 17

17 Основы планирования проекта 236

Глава 18

18 Пример дизайна системы — таймер клеевого пистолета 268

Глава 19

19 Основные интерфейсы и их программирование 273

Глава 20

20 Основы интерфейса аналого-цифрового преобразования 295

Глава 21

21 Основы проектирования системы 314

Глава 22

22 Основы проектирования системы — тайм-трекер 317

Глава 23

23 Основы вычислений времени 330

Глава 24

24 Основы строковых переменных 340

Глава 25

25 Силовые интерфейсы 353

Глава 26

26 Теория источников питания 370

Глава 27

27 Типичные вопросы тестирования

2011/12/13 годов 395

Глава 28

28 Расширенное программирование — массивы 397

Глава 29

29 Подтягивающие резисторы AVR 402

Глава 30

30 Дополнительно: подключение клавиатуры 403

Глава 31

31 Тонкости циклов Do-Loop & While-Wend 417

Глава 32

32 Подключение двигателя постоянного тока 423

Глава 33

33 Пример расширенной системы — будильник 452

Глава 34

34 Резистивный сенсорный экран 468

Глава 35

35 System Design Example – Temperature Controller 475

Глава 36

36 Advanced programming - state machines
478

Глава 37

37 Alarm clock project re-developed . 501

Глава 38

38 Advanced window controller student project
514

Глава 39

39 Alternative state machine coding techniques 524

Глава 40

40 Complex - serial communications . 526

Глава 41

41 Radio Data Communication 597

Глава 42

42 Introduction to I2C 617

Глава 43

43 Plant watering timer student project 631

Глава 44

44 Bike audio amplifier project 642

Глава 45

45 Graphics LCDs 648

Глава 46

46 GLCD Temperature Tracking Project 660

Глава 47

47 Interrupts 672

Глава 48

48 Timer/Counters . 692

Глава 49

49 LED dot matrix scrolling display project –
arrays and timers . 698

Глава 50

50 Medical machine project – timer
implementation 709

Глава 51

51 Multiple 7-segment clock project – dual
timer action 715

Глава 52

52 The MAX 7219/7221 display driver IC's 739

Глава 53

53 Cellular Connectivity-ADH8066 744

Глава 54

54 Data transmission across the internet 778

Глава 55

55 Assignment – maths in the real world 816

Глава 56

56 SSD1928 based colour graphics LCD . 825

Глава 57

57 Traffic Light help and solution 865

Глава 58

58 Computer programming – low level detail
869

Глава 59

59 USB programmer - USBASP . 876

Глава 60

60 USBTinyISP programmer 877

Глава 61

61 C-Programming and the AVR 881

Глава 62

62 Object Oriented Programming (OOP) in C++ and the AVR 929

Глава 63

63 Current (2014) AVR development PCBS
953

Глава 64

64 Eagle - creating your own library 970

Глава 65

65 Practical Techniques . 979

Глава 66

66 CNC 990

Глава 67

67 Index . 1008

Часть II

Основы электроники

Здесь идет список ссылок на онлайн лекции в edX, Coursera, и т.п.

Глава 68

Линейные схемы на пассивных элементах, основы электротехники

Глава 69

Симуляция и расчет схем в ngSPICE

Глава 70

KiCAD

70.1 Отрисовка схем в KiCAD

70.2 Библиотеки элементов

70.3 Передача схемы в ngSPICE

Глава 71

Простейшие полупроводниковые элементы

71.1 Оптоэлектроника

71.2 Схемы на биполярных транзисорах

71.3 Схемы на на полевых транзисорах

Глава 72

Операционные усилители

Глава 73

Источники питания

73.1 Батарейное питание

73.2 Линейные стабилизаторы

73.3 Импульсные преобразователи на ШИМ-контроллерах

73.4 Цепи защиты и гашения кондуктивных помех

Глава 74

Цифровая электроника

Глава 75

Компьютерные интерфейсы

75.1 Поколение 90х: COM, LPT, ISA

75.1.1 Резервный программатор AVR “пять проводков”

75.2 Сеть CAN

75.3 Интерфейсные модули USB

75.3.1 Универсальный высокоскоростной конвертер FTDI FT2232H

75.3.2 JTAG-адаптер

75.3.3 Отладочный модуль CAN

Глава 76

ПЛИС

Глава 77

Датчики

Глава 78

Электропривод и исполнительные устройства

Часть III

Основы конструирования РЭС

Глава 79

Пакеты моделирования на основе OpenFOAM

Глава 80

Обеспечение теплового режима

Глава 81

Электромагнитная совместимость

81.1 Кондуктивные помехи

81.2 Компоновочные модели и оптимизация кабельной сети

Часть IV

Технология РЭС

Глава 82

Инструменты и оборудование

82.1 JTAG-адаптер

82.2 Отладочные платы

Прежде чем начать работать с отдельными МК, устанавливая их на плату собственной разработки, для быстрого старта используют *отладочные платы*¹

¹ development board, demo board

82.2.1 Arduino /Atmel Mega AVR8/

82.2.2 Cortex-Mx

См. [97](#)

82.2.3 CubieBoard /Cortex-A8 AllWinner A10/

82.2.4 Raspberry Pi /ARM11 BCM3032/

82.2.5 BlackSwift /MIPS/

82.2.6 VoCore /MIPS/

82.3 Монтажный инструмент

82.4 Измерительное оборудование

82.4.1 Тестер

82.4.2 Осциллограф

82.4.3 Логический анализатор

82.4.4 Генератор сигналов

82.4.5 Рыльцемер

82.5 Электроинструмент

82.5.1 Дрель



Дрель ударная сетевая
Praktyl-R PID13D01 400 Вт
(!)395 р.



Дрель безударная сетевая
Интерскол Д-11/530ЭР (с БЗП)
1120 р.

Глава 83

Трассировка плат и подготовка производства в KiCAD

83.1 Технология ЛУТ (Лазерный УТюг)

83.2 Технология фоторезиста

83.3 Формат Gerber и подготовка промышленного производства

Глава 84

FreeCAD



1

В среде специалистов ряда отраслей известна проблема создания полноценной САПР в рамках OpenSource, и хотя FreeCAD ещё не является кандидатом на такую «полноценность», этот продукт может рассматриваться как одна из попыток создания базы для решения этой проблемы. Разработчик FreeCAD Юрген Ригель, работающий в корпорации DaimlerChrysler, позиционирует свою программу как первый бесплатный инструмент проектирования механики (сравнивая свой продукт с такими развитыми проприетарными системами как CATIA версий 4 и 5, SolidWorks), созданный на основе библиотеки **Open CASCADE**. Цель программы — предоставить базовый инструментарий этой библиотеки в интерактивном режиме.

Следует отметить, что имеет место ещё один программный продукт имеющий название freeCAD, его разработчик — Aik-Siong Koh, и он не связан с FreeCAD'ом Юргена Ригеля.

84.1 Установка под ☐Windows

☐+R >> <http://www.freecadweb.org/> >> Download >> ☐Windows >> **FreeCAD 0.14** >> ..._setup.exe

¹ копираста [https://ru.wikipedia.org/wiki/FreeCAD_\(Juergen_Riegel%27s\)](https://ru.wikipedia.org/wiki/FreeCAD_(Juergen_Riegel%27s))

84.2 Чертеж

84.3 Эскиз

84.4 Деталь

84.5 Сборка

84.6 Автогенерация конструкторской документации

84.7 Скрипты и пользовательские расширения

Глава 85

Эксплуатация станочного оборудования

Глава 86

Основы ЧПУ и цифрового производства

86.1 САМ-пакеты для FreeCAD

Часть V

Основы теории систем автоматического управления

Глава 87

Математический аппарат

87.1 Передаточная функция

87.2 Устойчивость САУ

87.3 Сети Петри

87.4 Автоматы Маркова

Глава 88

Релейное управление

Глава 89

Пропорциональные САУ

Глава 90

ПИДn-регуляторы

Часть VI

Разработка ПО для встраиваемых систем

Глава 91

Вспомогательные скрипты на языке Python



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 <http://www.python.org> (до выхода версии 2.5) изображают змеиные головы.

Python¹ — высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода.

Python удобно применять для написания различных вспомогательных скриптов. Часто его используют при разработке сложных программных систем для написания первых версий. В процессе работы над большими программами часто перерабатываются большие объемы кода, поэтому для ускорения разработки требуется максимально высокоуровневый язык. После того как архитектура программы стабилизируется, узким местом становится производительность, и программу переписывают на более низкоуровневом компилируемом языке, чаще всего C_+^+ .

Написание программ упрощают:

- **объектно-ориентированное программирование** облегчает разработку программ, позволяет переопределить стандартные операторы для пользовательских типов данных, упрощая синтаксис
- **динамическая типизация** не требуется заранее упределять переменные, они создаются простым присваиванием
- **обработка исключений** для секции кода можно определить обработчик ошибок
- **высокоуровневые структуры данных** — списки, словари (набор элементов ключ:значение), очереди
- богатая стандартная библиотека и множество дополнительных библиотек на все случаи

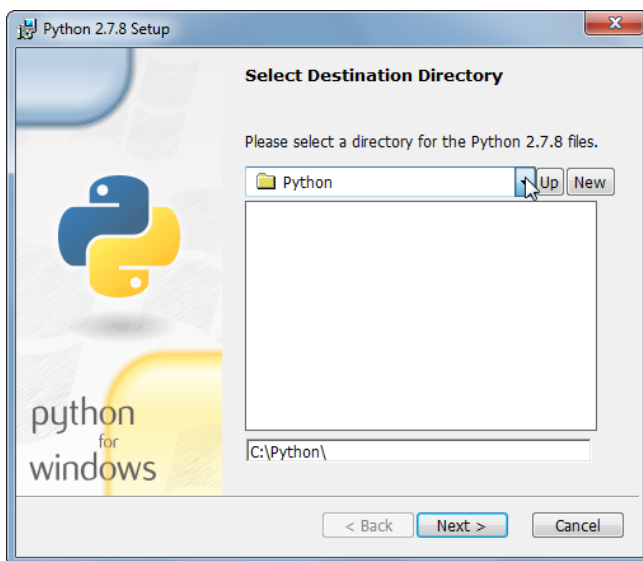
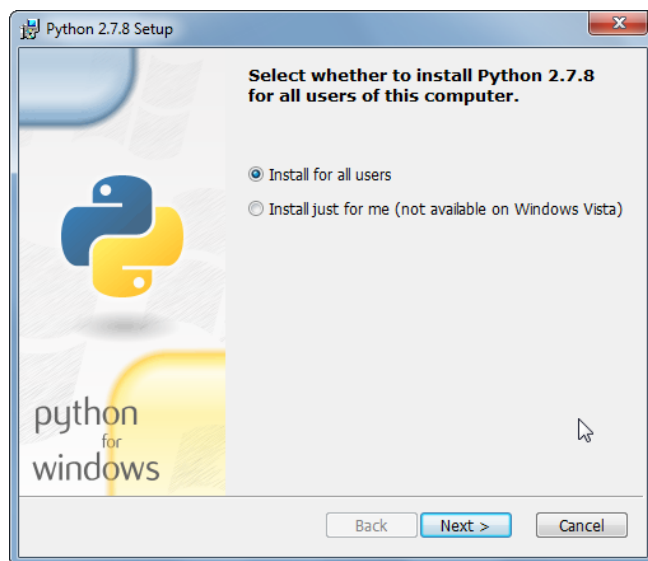
¹ в оригинале читается **пáйтoн**, но давно русифицировался как **питóн**

91.1 Установка под Windows

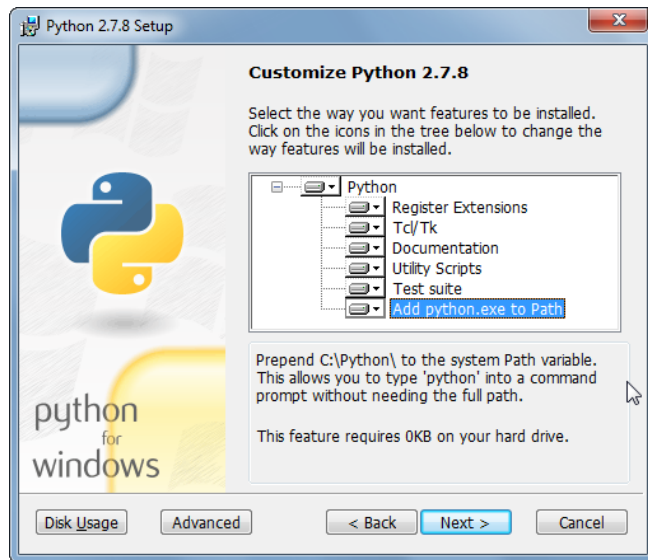
⊞ + R >> <http://www.python.org> >> Downloads >> Python 2.7.8

python-2.7.8.msi >> Setup >> for all users/for me


Destination Directory >> **C:/Python/** >> Next

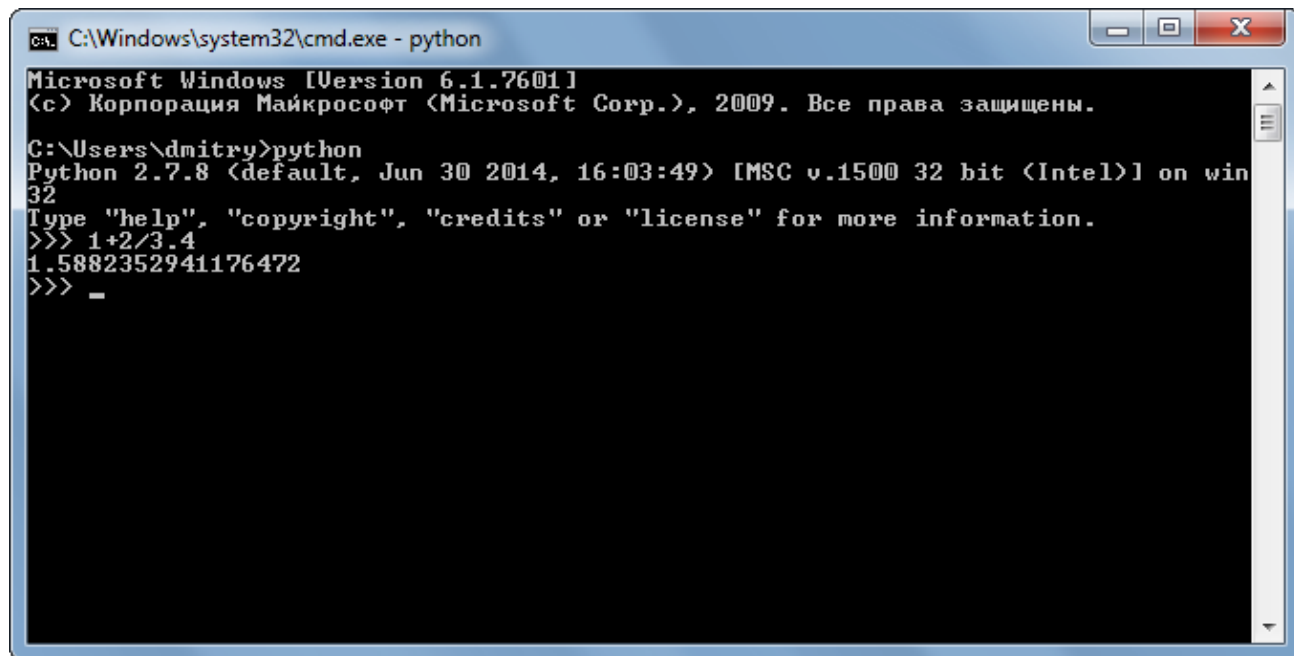


Customize » Python » Add python.exe to PATH » Next » Finish



91.2 Запуск

Из командной строки:  + R » cmd » python

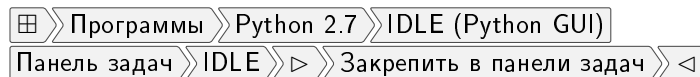


```
C:\Windows\system32\cmd.exe - python

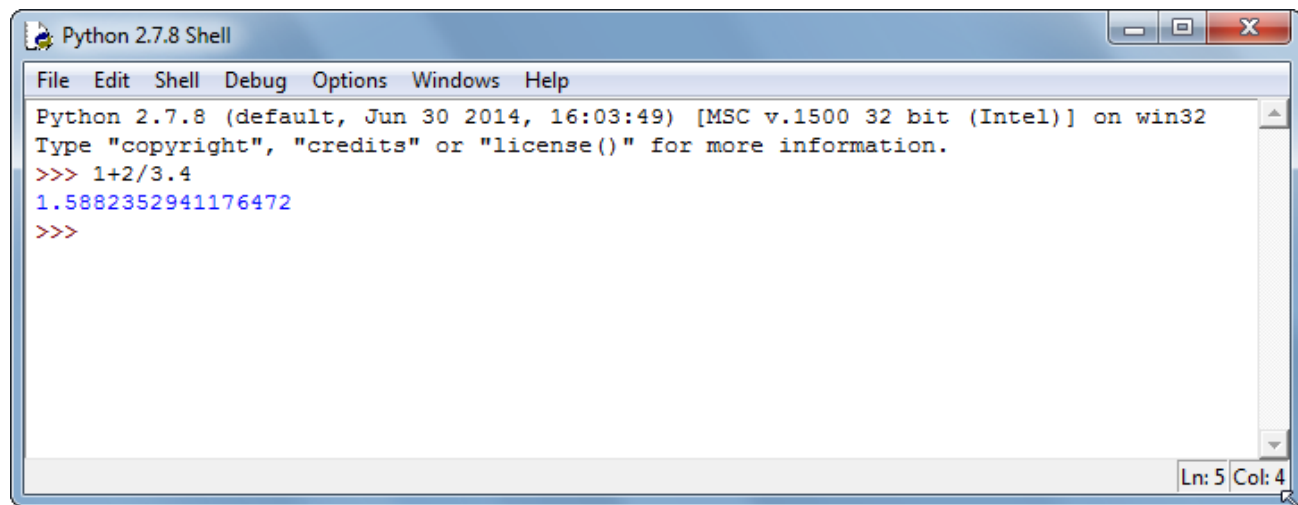
Microsoft Windows [Version 6.1.7601]
(c) Корпорация Майкрософт (Microsoft Corp.), 2009. Все права защищены.

C:\Users\dmitry>python
Python 2.7.8 (default, Jun 30 2014, 16:03:49) [MSC v.1500 32 bit <Intel>] on win
32
Type "help", "copyright", "credits" or "license" for more information.
>>> 1+2/3.4
1.5882352941176472
>>> _
```

Простейшая среда IDLE²:



² на GUI-библиотеке Tkinter, идущей в комплекте



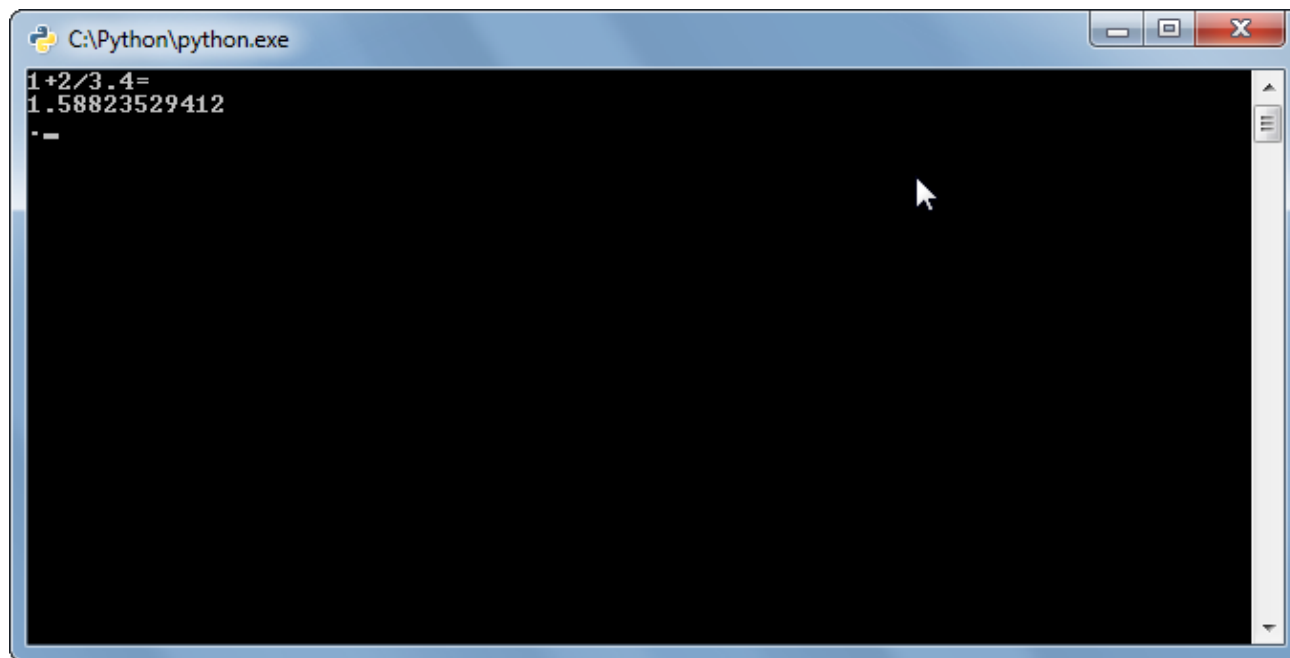
◀◀ по файлу скрипта:

 +  notepad /tmp/py.py

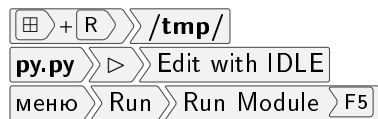
/tmp/py.py

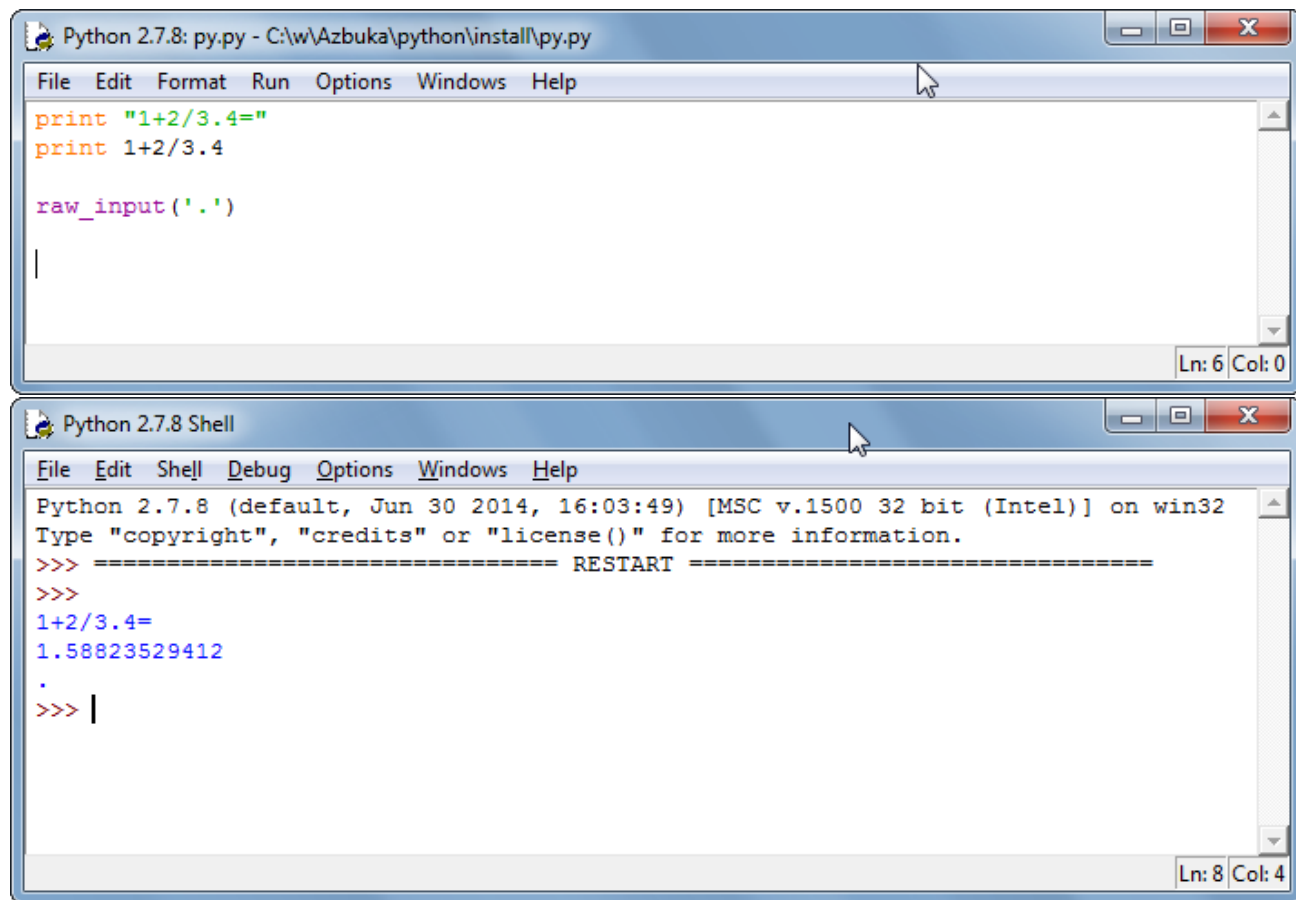
```
1 print "1+2/3.4="
2 print 1+2/3.4
3
4 raw_input( ' . ' )
```

 +  /tmp/py.py



Открытием файла скрипта в IDLE:





91.3 Дополнительные материалы

[pyotkidach] | Г. Россум, Ф.Л.Дж. Дрейк, Д.С. Откидач, [Язык программирования Python](#)

[pythink] | Аллен Дауни [Думать на языке Python: Думать как компьютерный специалист](#)

Глава 92

Make: управление сборкой проектов

Глава 93

VCS: системы контроля версий

93.1 CVS

93.2 Subversion

93.3 Git

93.3.1 GitHub

Глава 94

ОСНОВЫ СИ И C_+^+

94.0.2 Установка MinGW (win32)

94.1 Особенности C_+^+ в embedded

Глава 95

LLVM и разработка собственных компиляторов

95.1 Лексический и синтаксический анализ

95.2 Применение flex/bison для разбора текстовых форматов данных

95.3 Компилятор Паскаля

Глава 96

Сборка кросс-компилятора GNU toolchain

Часть VII

Микроконтроллеры Cortex-Mx

Глава 97

Отладочные платы

97.1 STM32DISCOVERY /Cortex-M3 STM32F103/

97.2 STM32F4DISCOVERY /Cortex-M4 STM32F407/

Часть VIII

Периферия

Часть IX

Встраиваемый emLinux

Глава 98

cross

Глава 99

BuildRoot

Глава 100

Особенности OpenWrt

Глава 101

Библиотека SDL

101.1 Реализация microGUI

Глава 102

Приложения для X Window

Глава 103

Программирование сетевых приложений

Глава 104

Сборка кросс-компилятора GNU мальтийским крестом

Часть X

IDE



IDE — Integrated Development Environment, интегрированная среда разработки.
Программный пакет, включающий

- средства управления проектом,
- отслеживание зависимостей между файлами (в т.ч. с анализом исходного текста программ на конструкции типа `#include`, `module`, `uses`),
- автозапуском компиляторов для изменившихся файлов,
- GUI для отладчиков (gdb),
- специализированный редактор plain text¹ файлов с
 - цветовой и шрифтовой *подсветкой синтаксиса*,
 - *автодополнением*: дописываются имена объектов программ, синтаксические конструкции и параметры функций,
 - *автоформатированием*: фрагмент текста переформатируется в соответствии с синтаксисом языка редактируемого файла, проставляются отступы в зависимости от вложенности синтаксических конструкций типа циклов и условных блоков)
 - выделением строк, на которые указывают сообщения об ошибках компиляторов,
 - маркеры точек останова отладчика
- отображение структуры программ, например деревья классов и структур данных

¹ файлы не включающие непечатаемых символов и бинарных данных, которые можно причитать простым выводом на экран командами типа **type**, **cat**, **more**

- контекстные справочники по используемым языкам программирования, автоматический вывод списка параметров при вводе имени функции
- отображение дизассемблерных листингов для компилируемых языков
- отображение браузера как вкладки или MDI окна
- отображение вывода *статических анализаторов* программ с кликабельными ссылками
- вывод компиляторов и трансляторов с цветовым выделением и переход на ошибочную строку в редакторе при щелчке на ошибке
- ...

В этой книге рассмотрены три бесплатных мультиплатформенных OpenSource IDE, в порядке навороченности, универсальности, и требуемым ресурсам для работы самой среды:

1.  ECLIPSE [105](#): самая навороченная и ресурсоемкая IDE, написана на Java, имеет десятки дополнительных модулей на все случаи, умеет работать со всеми распространенными языками программирования, жрет память, и требует современного компьютера минимум с 2+ Гб ОЗУ. Последний релиз  ECLIPSE Luna работает заметно быстрее (особенно при запуске).
2. Code::Blocks [106](#): легкая среда для разработки на C/C₊⁺, для других языков может потребоваться написать свои модули или файлы описания синтаксиса
3. (g)Vim [107](#): самый легкий и *портатбельный* универсальный текстовый редактор с расширенными функциями, работает на всех существующих платформах (кроме совсем уж embedded), использует минимум ресурсов, но требует некоторого обучения даже чтобы выйти из vim ☺

Глава 105

☾ECLIPSE



105.1 Проверка орфографии

1

То, что проверка орфографии очень удобная вещь вряд ли нужно объяснять. Есть конечно люди, которые не обращают на неё внимание, но это чаще всего из-за экономии времени и отсутствия удобных средств проверки.

Действительно, удобная автоматическая проверка орфографии есть в офисных пакетах, но мне сложно представить разработчика, который будет переносить комментарии в Word и обратно ☹.

Поэтому очень удобно иметь *проверку правописания прямо в IDE*. И ☹ECLIPSE в этом смысле полностью соответствует ожиданиям.

Долго объяснять что к чему нет смысла. Проверка орфографии встроена в ☹ECLIPSE и если вы пишете только на английском, то может быть не захотите ничего менять.

Кроме того, есть **статья Aaron'a** (en) в которой автор рассказывает о подключении дополнительных словарей и плагине **eSpell**.

Но *русских словарей в дистрибутиве нет*, а при подключении внешних есть нюансы. Поэтому мы максимально подробно рассмотрим *подготовку и добавление русских словарей*.

Первый вопрос. В каком виде должны быть словари и где их взять?

Тут всё просто. Формат словаря — обычный текстовый файл, в котором каждое слово начинается с новой строки. И нам вполне подойдут свободно распространяемые словари **aSpell**.

Установка состоит из 4 шагов:

1. качаем aSpell и словари для нужных языков



¹ копияста <http://www.simplecoding.org/proverka-orfografii-v-eclipse.html>

Binaries » Full installer

Precompiled dictionaries » English

Precompiled dictionaries » Russian

2. устанавливаем сначала **aSpell**, потом отдельно каждый словарь

Aspell-0-50-3-3-Setup.exe » Setup GNU Aspell » Next » License » Next

Directory » **C:/GnuWin32/Aspell** » Next » Next

Additional » Next » Install » Next » ☐ View manual » Finish

Aspell-en-0.50-2-3.exe » Aspell English Dictionary » Next » License » Next

Directory » **C:/GnuWin32/Aspell** » Next » Next » Install » Finish

Aspell-ru-0.50-2-3.exe » Aspell Russian Dictionary » Next » License » Next

Directory » **C:/GnuWin32/Aspell** » Next » Next » Install » Finish

3. делаем дампы словарей, перекодировем из koi8r в utf8 и объединяем

 + R cmd

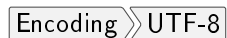
```
1 cd \GnuWin32\Aspell
2 bin\aspell dump master en > en.dict
3 bin\aspell dump master ru > ru.koi8
4 iconv -f koi8-r -t utf-8 < ru.koi8 > ru.dict
5 copy en.dict + ru.dict enru.dict
```

4. настраиваем *spell-checker* ☹ECLIPSE

☹ECLIPSE > Window > Preferences > Editors > Text editors > Spelling



User defined dictionary > **C:/GnuWin32/Aspell/enru.dict**



Encoding > UTF-8



Apply > OK

Глава 106

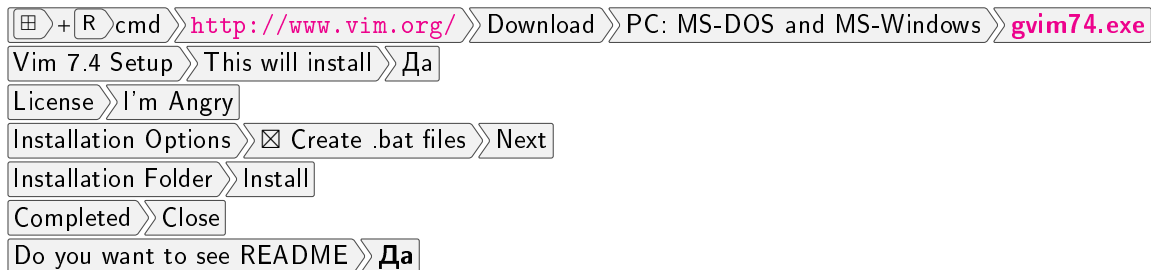
Code::Blocks

Глава 107

(g)Vim



107.1 Установка под Windows



Теперь можно настроить темную тему и выключение подстветки синтаксиса, по умолчанию после установки используется светлая тема и подстветка выключена:



Переходим в конец файла и включаем *режим вставки*



```
1 syntax on
2 colorscheme pablo
```

Выходим в *режим команд* и принудительно сохраняем




Выходим из (g)Vim



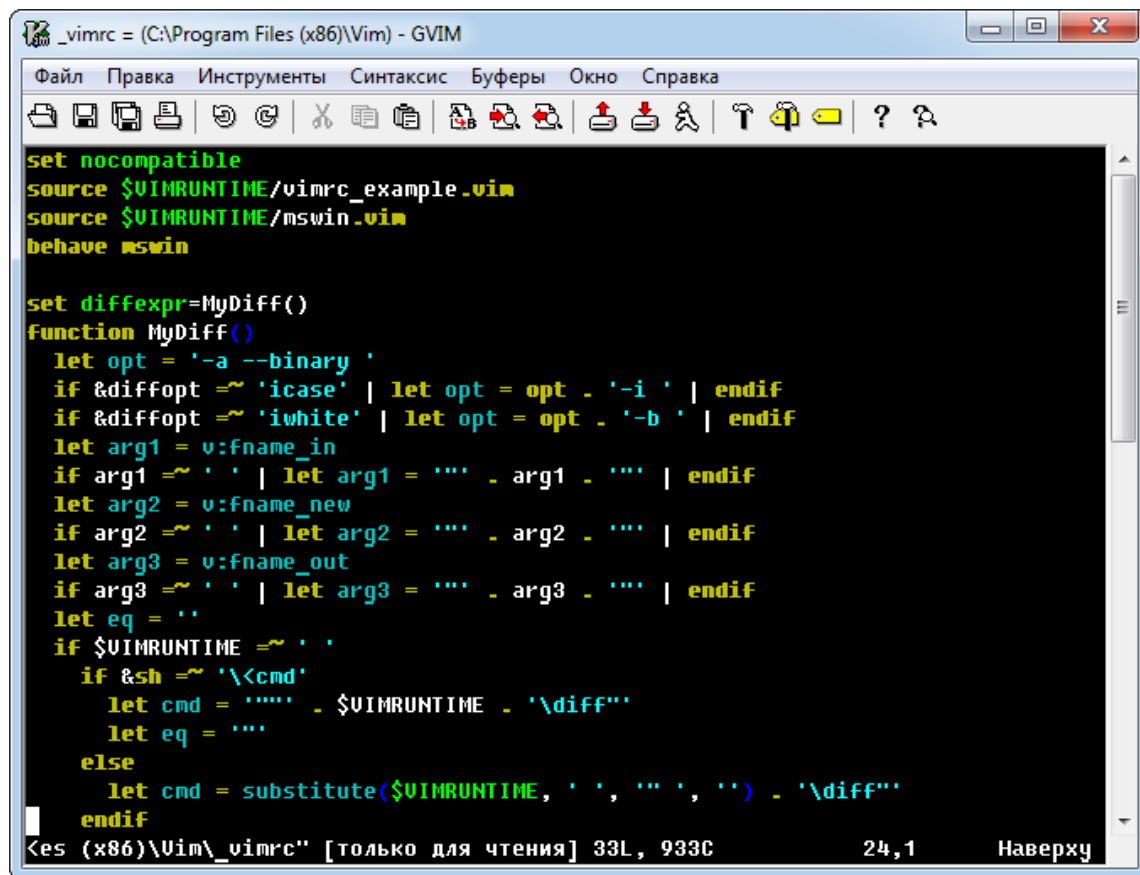
Если не получилось (под Windows 7):

 +  cmd >> /Program Files (x86)/Vim/

Копируем файл **_vimrc** в любой каталог, например в **/tmp/**, затем  Edit with Vim, и повторяем редактирование еще раз.

Затем копируем **_vimrc** обратно в **/Program Files (x86)/Vim/** с заменой.

Если теперь открыть на редактирование тот же файл, или любой другой текстовый, получим более удобный вид: для файлов известных типов будет работать подсветка синтаксиса.



The screenshot shows the GVIM editor window titled "_vimrc = (C:\Program Files (x86)\Vim) - GVIM". The menu bar includes "Файл", "Правка", "Инструменты", "Синтаксис", "Буферы", "Окно", and "Справка". The toolbar contains icons for file operations, editing, and navigation. The main text area displays the contents of the vimrc file with syntax highlighting: keywords in green, variables in yellow, and strings in red. The code includes settings for compatibility, sourcing example and mswin vimrc files, and a custom diff function. The status bar at the bottom shows the file path, line and column numbers, and a "Наверх" (Up) button.

```
set nocompatible
source $VIMRUNTIME/vimrc_example.vim
source $VIMRUNTIME/mswin.vim
behave mswin

set difffexpr=MyDiff()
function MyDiff()
  let opt = '-a --binary '
  if &diffopt =~ 'icase' | let opt = opt . '-i ' | endif
  if &diffopt =~ 'iwhite' | let opt = opt . '-b ' | endif
  let arg1 = v:fname_in
  if arg1 =~ ' ' | let arg1 = '"' . arg1 . '"' | endif
  let arg2 = v:fname_new
  if arg2 =~ ' ' | let arg2 = '"' . arg2 . '"' | endif
  let arg3 = v:fname_out
  if arg3 =~ ' ' | let arg3 = '"' . arg3 . '"' | endif
  let eq = ''
  if $VIMRUNTIME =~ ' '
    if &sh =~ '\<cmd'
      let cmd = '""' . $VIMRUNTIME . '\diff"'
      let eq = ''
    else
      let cmd = substitute($VIMRUNTIME, ' ', '" ', '') . '\diff"'
    endif
  endif
endfunction
<es (x86)\Vim\_vimrc" [только для чтения] 33L, 933C      24,1      Наверх
```


107.2 Выход из (g)Vim

`Esc` : `!` `q` `Enter`

107.2.1 Выход с автосохранением

`Esc` `Shift` + `Z` `Shift` + `Z`

107.3 Переход в режим редактирования

(g)Vim запускается в *командном режиме*, для перехода в режим редактирования используются следующие клавиатурные команды:

- `Ins` или `i`: включение *режима вставки* по текущему положению курсора
- `Ins Ins` или `r`: включение *режима перезаписи* поверх текста после курсора
- `Shift` + `A`: включение режима вставки *в конец текущей строки*

107.4 Переход в режим команд

`Esc`

107.5 Запись редактируемого файла

`Esc` : `w` `Enter`

Если выводится предупреждение типа “файл защищен от записи” или подобное, может сработать принудительная запись:

`Esc` : `!` `w` `Enter`

107.6 Перезагрузка файла

Для перезагрузки возможно измененного извне файла или отмены всех несохраненных изменений

`Esc` : `e` `Enter`

107.7 Отмена последних изменений (undo)

`Esc` `u` `u` ...

Часть XI

Замечания для участников проекта

107.8 Набор репозиторий на GitHub

https://github.com/ponyatov/Azbuka	основная репа
https://github.com/ponyatov/bib	библиографические базы данных
https://github.com/ponyatov/scratcher	журнал, используются некоторые материалы

Для работы с проектом сделайте собственный форк основной репы, библиографическую базу и журнал можете клонировать напрямую. Создайте каталог и склонируйте репы:

```

1 D:
2 cd \
3 mkdir w
4 cd \w\
5 git clone --depth=1 -o gh git@github.com:username/Azbuka.git
6
7 git clone --depth=1 -o gh git@github.com:ponyatov/bib.git
8 git clone --depth=1 -o gh git@github.com:ponyatov/scratcher.git

```

107.9 Верстка в L^AT_EX

Часть XII

Подготовка публикаций в L^AT_EX

LaTeX (по-русски произносится **латéx**) — наиболее популярный набор макрорасширений (или макропакет) системы компьютерной вёрстки T_EX, который облегчает набор сложных документов. В типографском наборе форматируется как L^AT_EX.

Главная идея L^AT_EX состоит в том, что авторы должны думать о содержании, о том, что они пишут, не беспокоясь о конечном визуальном облике (печатный вариант, текст на экране монитора или что-то другое). Готовя свой документ, автор указывает логическую структуру текста (разбивая его на главы, разделы, таблицы, изображения), а L^AT_EX решает вопросы его отображения. Так содержание отделяется от оформления. Оформление при этом или определяется заранее (стандартное), или разрабатывается для конкретного документа.

В практическом смысле использование L^AT_EX позволяет (в порядке уменьшения важности):

- с помощью макросов и T_EX-программирования реализовывать любые стили и самую сложную верстку, существует множество готовых пакетов для верстки графических химических формул, разнообразных схем, транскрипционных знаков, внезапно электронных схем, цветных листингов и т.п.
- автоматизировать работу с документами: пересобирать выходные файлы через [Make](#), генерировать части документов с помощью своих скриптов²
- получить выходной документ в .pdf .html .txt .PostScript .djvu ... с кликабельными ссылками, анимированными, а иногда и интерактивными элементами
- не использовать файлы документов в закрытом формате

¹ копияста <https://ru.wikipedia.org/wiki/LaTeX>

² отчеты, стандартные формы, результаты работы любых программ

- легко держать набор файлов в [VCS](#)
- не покупать текстовый процессор

Особенно важен пункт про сложную верстку: она всегда нужна в крупных технических публикациях, особенно в учебной литературе, или отчетных работах. Вам обязательно понадобится вставлять графики экспериментальных данных, тематически специфичные схемы, листинги, выходные данные работы ваших программ и т.п.

Традиционно \LaTeX любим математиками, и всеми кто готовит публикации с большим количеством формул и перекрестных ссылок: после небольшого обучения формулы вводятся с листа со скоростью набора текста, особенно если ваш редактор умеет автодополнение, и никакой мышью возни.

Естественно всякие чисто автоматические вещи типа автонумерации ссылок и формул, сборки оглавлений и индексов, цветовая подсветка синтаксиса в листингах программ, размещение плавающих иллюстраций и т.п. выполняются автоматически \TeX -процессором в пакетном режиме, и на выходе получается красивый печатный или электронный (.pdf) документ.

Единственная область, не удобная в \LaTeX -верстке — создание сложных таблиц. Для этого были созданы визуальные редакторы, позволяющие отрисовать структуру таблицы мышью, а затем заполнить готовый шаблон данными.

107.10 Установка MikTeX под ☐Windows

107.11 Структура документа

107.11.1 Заголовочный файл или блок

107.11.2 Стили документа

107.11.3 Пакеты

107.11.4 Автор и название

107.11.5 Верстка титульных страниц

107.11.6 Оглавление

107.12 Верстка слайдов

107.13 Список литературы и цитирование

Л^AT_EX умеет мощную подсистему управления цитированием и списками литературы. В простейшем случае, например при написании единственной статьи, раздел *библиографии* можно создать в том же документе, добавив в конец `thebibliography`:

```
\documentclass{article}
```



```

\input{header}

\author{Вася Пупкин}
\title{Пример статьи с цитатами}

\begin{document}
\maketitle

В статье используются книги: \cite{A} и \cite{B}

\begin{thebibliography}{99}

\bibitem{A} Книга А

\bibitem{B} Книга В

\end{thebibliography}
\end{document}

```

Но если вы регулярно работаете с документацией, или часто пишете статьи, возникает естественное желание вынести весь список литературы в отдельную базу данных, прописать авторов, названия, издательства и т.п. Это делается с помощью программы **biber** и пакета **biblatex**.

Пример использования этой системы вы легко найдете в исходниках этой книги:

- файл **header.tex** содержит секцию подключения пакета и подгрузки библиофайлов:

```
% books bib management
\usepackage{biblatex}
\addbibresource{../bib/python.bib}
\addbibresource{../bib/eskd.bib}
...
```

- библиофайлы хранятся в **соседнем** репозитории `../bib`, клонированном с <https://github.com/ponyatov/bib>.
- порядок вызова **pdflatex** и **biber** см. **Makefile**

- 107.14 Команды секционирования: часть, глава, раздел,..
- 107.15 Таблицы
- 107.16 Формулы
- 107.17 Перекрестные ссылки и гипессылки
- 107.18 Листинги скриптов и текстовых данных
- 107.19 Подготовка иллюстраций
 - 107.19.1 Графики GNUPLOT
 - 107.19.2 Схемы и графы в GraphViz

Часть XIII

Куча

В этот раздел собраны все материалы, не вошедшие в основную часть потому что слишком сложны для начинающих, не попадают не в один раздел по тематике, или не вписались по каким-то другим параметрам.

Все новые материалы также сначала попадают сюда, а потом принимается решение об их переносе в основную часть.

Часто сюда пишут статьи те, кто принимает участие в создании книги эпизодически, или те, у кого нет достаточно времени заниматься их подготовкой.