1. Секция

Исходный код 1: Name

```
Text
```

Замечание. Тут замечание.

***Определение* 1** (Lol)
тут что-то определить

***Теорема* 2** (A Great Theorem)
Something Great

2. Секция

***Следствие* 3**
Lol

4
Lol

***Теорема* 5**
Lol

```
int main(int argc, char **argv) {
    return 0;
}
```

Алгоритм 1: My Code

Замечание. Тут замечание.[1]

В этой главе — вероятно самой «информатической» в нашей книге — мы введем некоторые алгоритмические понятия и опишем несколько получисленных алгоритмов, необходимых для реализации алгебраиче ских методов. Точное определение алгоритмических границ, в которых будут из

***Определение* 6**
В этой главе — вероятно самой «информатической» в нашей книге — мы введем некоторые алгоритмические понятия и опишем несколько получисленных алгоритмов, необходимых для реализации алгебраиче ских методов. Точное определение алгоритмических границ, в которых будут из

[1] Lol

укоризненным способом семантику построений, используемых в алго ритмах 1 . После изложения общих направлений[1] мы приступим к исследова нию первого фундаментального алгоритма этой книги: дихотомиче ского алгоритма возведения в степень. Для этого будут использоваться два метода построения алгоритмов. Один очень близок к математиче ским рассуждениям, другой — чисто алгоритмический. При этом у нас появится возможность открыть первый рекурсивный алгоритм, пред ставленный в данной работе. Начав с первого примера алгоритма, мы перейдем к изучению на чальных понятий программирования на языке Ада. И здесь тоже речь пойдет не об изложении полного курса языка Ада, а только о пред-

[1] Лол Кек Чебурек!

ставлении нескольких ключевых понятий, необходимых для понимания приведенных программ. Вообще говоря, все понятия будут представлены (по мере надоб ности) на примерах, во избежание их формального описания (по по следнему вопросу лучше обратиться к справочному изданию: Reference manual for the Ada[1] programming language 1). И наконец, завершит эту главу разработка совершенно неэффек тивного метода вычисления определителей целочисленных матриц. Об суждаемый метод, основанный на математическом определении детер минанта, позволяет наилучшим образом использовать все возможности современных компьютеров и получить идеальное приближение к беско нечности на машине! Кроме того, обсуждение ситуации позволит нам несколько пополнить представление о программировании на языке Ада.

3. Введение в алгоритмику

Прежде чем приступить к описанию любого алгоритма, нужно ого ворить общепринятые соглашения, используемые в работе. В инфор матике чаще, чем в любой другой дисциплине, встречается взаимное непонимание между пользователем и разработчиком новых программ. Поэтому так важно сформулировать аксиоматику, позволяющую связ но рассуждать об алгоритмах. Множество аксиом, которое мы собираемся сейчас описать, извест но как аксиоматика Хоара [86], в честь исследователя, который первым формализовал семантику структур и действий последовательной алго- ритмики.

3.1. Терминология и обозначения

А л г о р и т м — это последовательность более или менее элементарных действий, которая позволяет пошагово решить поставленную задачу за конечное время. Можно формально по индукции определить, что та кое алгоритм, нижеследующим образом. Цель этого введения в алго ритмику — изучение и представление всех терминов, фигурирующих в предыдущем определении (стиль определения, который в информатике называется гр а м м а ти к ой). Можно описать алгоритм в терминах состояний системы: состояние системы до применения алгоритма, состояние той же системы после

[1] Heh, ADA

применения алгоритма. Эти два состояния образуют то, что называет
ся сп ец иф икац ией задачи. Таким же образом можно, описывая поэ