
1.1. Программа

1.1.1. Общие сведения

Основная структурная единица языка Кумир – *алгоритм*. Программа на языке КуМир в простейшем случае состоит из нескольких алгоритмов, следующих один за другим. Перед первым алгоритмом может располагаться *вступление* – любая неветвящаяся последовательность команд. Например, это могут быть строки с комментариями, описаниями общих величин программы, командами присваивания им начальных значений.

Алгоритмы в программе должны располагаться вплотную друг к другу, между ними могут быть только пустые строки и строки с комментариями.

1.1.2. Выполнение программы

Выполнение программы состоит в выполнении вступления, если оно присутствует, а затем – первого алгоритма (он называется *основным алгоритмом программы*). Остальные алгоритмы будут выполняться при вызове из первого алгоритма или из других ранее вызванных алгоритмов.

Скопировать пример

```
| Это вступление
цел длина, ширина
длина := 10
ширина := 15
| Это - основной алгоритм.
```

```
| У него может не быть имени
алг
нач
  вывод "Площадь равна ", площадь
кон
| Это - вспомогательный алгоритм.
| При выполнении он вызывается из основного.
| У вспомогательного алгоритма обязательно
| должно быть имя и могут быть параметры.
алг цел площадь
нач
  знач := длина*ширина
кон
```

Пример 1.1. Программа со вступлением, основным алгоритмом без имени и вспомогательным алгоритмом

Скопировать пример

```
| Это вступление
вещ длина, ширина, масса
длина := 10
ширина := 15
алг
нач
  вещ S
  S := площадь
  вещ плотность, масса
  плотность := 6.8 | г/см**2
  найти массу пластинки(плотность, S, масса)
  вывод "Масса пластинки равна ", масса
кон

| Это - вспомогательный алгоритм.
| При выполнении он вызывается из основного алгоритма.
| У вспомогательного алгоритма
| обязательно должно быть имя.
алг вещ площадь
нач
  знач := длина*ширина
кон

| Это - еще один вспомогательный алгоритм.
| При выполнении он вызывается из
| другого вспомогательного алгоритма.
| У вспомогательного алгоритма
| обязательно должно быть имя.
| У вспомогательного алгоритма могут быть параметры
алг найти массу пластинки(арг вещ p, S, рез вещ m)
нач
```

```
  m := p*S
кон
```

Пример 1.2. Программа со вступлением, основным алгоритмом и вспомогательным алгоритмом

1.2. Описание алгоритма

1.2.1. Общий вид описания

Алгоритм на языке Кумир записывается в следующем виде:

```
алг тип_алгоритма имя_алгоритма (описание_параметров)
  дано условие_применимости_алгоритма
  надо цель_выполнения_алгоритма
нач
  последовательность команд
кон
```

Описание алгоритма состоит из:

- заголовка (часть до служебного слова **нач**);
- тела алгоритма (часть между словами **нач** и **кон**).

1.2.2. Алгоритмы-процедуры и алгоритмы-функции

Алгоритмы делятся на *алгоритмы-процедуры* и *алгоритмы-функции*. Алгоритм-функция после выполнения возвращает значение-результат. Правила описания алгоритмов-процедур и алгоритмов-функций имеют два отличия.

Во-первых, для алгоритмов-функций на месте тип_алгоритма должен быть указан один из простых типов алгоритмического языка (например, **вещ**, **цел** и т. д.), определяющий тип значений, которые принимает данная функция. Для алгоритмов-процедур тип_алгоритма должен быть опущен.

Во-вторых, в теле алгоритма-функции необходимо использовать служебную величину **знач**, в которую записывается вычисленное значение функции. В теле алгоритма-процедуры величину **знач** использовать нельзя.

Алгоритмы-функции и алгоритмы-процедуры отличаются также по способу вызова .

Скопировать пример

```
алг гипотенуза (вещ a, b, рез вещ c)
  дано a>=0 и b>=0 | длины катетов треугольника
  надо | c = длина гипотенузы этого треугольника
нач
  c := sqrt( a**2 + b**2 )
```

```
кон
```

Пример 1.3. Пример алгоритма-процедуры

Скопировать пример

```
алг вещ площадь (вещ a, b, c)
  дано a>=0 и b>=0 и c>=0 | длины сторон треугольника
  надо | значение функции равно площади этого треугольника
нач
  вещ p | полупериметр
  p := (a+b+c)/2
  знач := sqrt(p*(p-a)*(p-b)*(p-c))
кон
```

Пример 1.4. Пример алгоритма-функции

1.3. Параметры алгоритма

Если алгоритм не имеет параметров (аргументов и результатов), то в строке **алг** записывается только имя алгоритма.

Если у алгоритма есть параметры, то их описание заключается в круглые скобки после имени алгоритма в строке **алг**. Описание содержит информацию о типах параметров и о том, являются они аргументами или результатами:

- **арг** – описания параметров-аргументов;
- **рез** – описания параметров-результатов;
- **аргрез** (или **арг рез**) – описания параметров, которые одновременно являются и аргументами, и результатами.

После каждого из служебных слов **арг**, **рез**, **аргрез** должно располагаться одно или несколько описаний одной или нескольких величин. Имена величин и описания разделяются запятыми. Если в начале описания нет служебных слов **арг**, **рез**, **аргрез**, то предполагается, что первыми идут описания аргументов (**арг**).

Скопировать пример

```
алг
нач
  вещ число
  цел целое, сотые
  лит запись
  число := 3.14
  тест(целое, сотые, запись, число)
```

```
кон

алг тест (рез цел m, n, лит т, арг вещ y)
нач
  вещ r
  m := int(y)
  r := (y - m)*100
  n := int(r)
  т := вещ_в_лит(y)
кон
```

Пример 1.5. Описание алгоритма с параметрами

В заголовке алгоритма **алг тест (рез цел** m, n, **лит т, арг вещ** y) служебное слово **рез** относится к описаниям **цел** m, n и **лит т**, а параметр **вещ** y будет аргументом.

ВНИМАНИЕ: Запрещается писать в теле алгоритма команды, изменяющие значения параметров-аргументов (описанных как **арг**). Результаты алгоритма (**рез**, но не **аргрез**) в начале выполнения алгоритма принимают неопределенные значения.

1.4. Команды и строки

В простейшем случае каждая простая команда и каждое ключевое слово в составных командах пишется на отдельной строке. Однако, чтобы сделать программу более компактной, можно ‹‹склеивать›› несколько строк в одну. Это можно сделать в следующих случаях.

1.4.1. Использование точки с запятой

Точка с запятой приравнивается к переносу строки.

Скопировать пример

```
| Программа 1 – сжатое написание
алг
нач
  цел a; вещ в
  a := 5; в := 0.1
кон
| Программа 2 – полное написание
алг
нач
  цел a
  вещ в
  a := 5
  в := 0.1
кон
```

Пример 1.6. Программа 1 и Программа 2 имеют одинаковый смысл

1.4.2. Неявные переносы строк

Для многих ключевых слов можно догадаться, что перед ними или после них должен быть перевод строки.

‹‹Неявные›› переносы строк вставляются в следующих случаях:

- перед словами **все**, **кц**, **кц_при**;
- после слов **нач**, **выбор**, **нц** (только в случае цикла нц-кц), **раз**;
- перед и после слов **то**, **иначе**, **при**;
- перед словом **при** и после двоеточия в при-строке.

Скопировать пример

```
алг
нач цел знак, вещ модуль
  вещ щ
  ввод щ
  модуль :=0; знак := 0
  если щ > 0 то
    модуль :=щ; знак := 1
  все
  если щ < 0 то модуль :=щ; знак := 1 все
кон
```

Пример 1.7. Использование неявных переносов строк

1.5. Комментарии

Скопировать пример

```
алг
  | Это алгоритм вычисления суммы двух чисел
нач
  цел а, б | объявляем величины
  ввод а, б | вводим значения с клавиатуры
  вывод а+б | посчитаем сумму чисел
кон
```

Пример 1.8. Использование комментариев

В этом алгоритме после знака | в некоторых строках записаны комментарии. Такие комментарии разрешается помещать в конце любой строки, отделяя их знаком |. Если комментарий занимает несколько строк, то знак | перед комментарием надо ставить в каждой строке. Комментарии могут записываться в любой удобной для человека форме. При выполнении алгоритма компьютер полностью пропускает комментарии (алгоритм выполнятеся так же, как если бы комментариев вообще не было).

Таким образом, комментарии предназначены исключительно для человека – они облегчают понимание алгоритма.