

Рекурсия

Рекурсия

- Рекурсивным называется объект частично состоящий или определенный с помощью самого себя.
- Примеры: Факториал $n! = n \cdot (n-1)!$, $0! = 1$
- Мощностъ рекурсивного определения заключается в том, что оно позволяет с помощью конечного высказывания определить бесконечное множество объектов.

Рекурсия

- В общем виде рекурсивную процедуру или функцию **P** можно выразить как некоторую композицию из множества операторов **S**, не содержащих **P**, и самой процедуры или функции **P**:
- $$P \equiv P[S,P]$$

Рекурсия

- Если некоторая процедура или функция P содержит явную ссылку на саму себя, то ее называют **прямо рекурсивной**
- $P \equiv P[S, P]$
- Если же P ссылается на другую процедуру или функцию Q , содержащую ссылку на P , то P называют **косвенно рекурсивной**
- $P \equiv P[S1, Q]$ и $Q \equiv Q[S2, P]$

Рекурсия

- Подобно операторам цикла, рекурсивные процедуры могут приводить к незакончивающимся вычислениям!!!
- Чтобы избежать этого, нужно на рекурсивное обращение к P поставить некоторое условие B, которое в некоторый момент становится ложным:
- $\text{if } B \text{ then } P$

Рекурсия

- Function fact(n:integer):longint;
- begin
- if $n=0$ then
- fact:=1
- else
- fact:=n*fact(n-1);
- end;

- $\text{fact}(3)=3*\text{fact}(2)=3*2*\text{fact}(1)=3*2*1*\text{fact}(0)=$
- $= 3*2*1*1$

Рекурсия

- В практических приложениях важно убедиться, что максимальная глубина рекурсии не только конечна, но и достаточно мала.
- Поскольку рекурсивный вызов процедуры или функции P требует памяти для размещения локальных переменных и для сохранения текущего состояния вычислений.

Рекурсия

- Именно по этой причине (не эффективное использование ресурсов ЭВМ) рекомендуется где это возможно заменять рекурсию на итерацию (использование циклов).
- Однако это не означает, что от рекурсии нужно избавляться любой ценой.

Рекурсия vs Итерация

- Function fact2(n:integer):longint;
- var i,s:integer;
- begin
- s:=1;
- for i:=1 to n do s:=s*i;
- fact2:=s;
- end;