Пусть a и n – целые неотрицательные числа. Составить программу-функцию возвращающую величину a^n.

Способ1. Функция power(a,n) дает решение задачи за n рекурсивных вызовов:

$$power(a,n) := \begin{cases} 1 \ if \ \ n = 0 \\ power(a, n-1) \cdot a \end{cases}$$

```
program stepen;
 var x,y:byte;
function power(a,n:byte):longint;
begin  if n=0 then power:=1
else power:=a*power(a,n-1);
end;
begin readln(x,y); writeln(power(x,y)); readln; end.
```

Способ2 Можно составить программу, при исполнении которой значения переменных a и n не меняются, а значение некоторой другой переменной (например, b) становится равным a в степени n. Число действий (выполняемых операторов присваивания) будет порядка log2 n (то есть не превосходило бы C * log2 n для некоторой константы C; log2 n — это степень, в которую нужно возвести 2, чтобы получить n).

Каждый второй раз (не реже) будет выполняться первый вариант оператора выбора (если k нечетно, то после вычитания единицы становится четным), так что за два цикла величина k уменьшается по крайней мере вдвое.

Алгоритм анализа показателя степени	
Без рекурсии	С рекурсией

var n,a,k,b,c:longint;
begin
readln(a,n);
k := n; b := 1; c := a;
while k <> 0 do
begin
if k mod 2 = 0 then
begin k := k div 2;
c := c * c
end
else begin k := k - 1;
b := b * c
end ;
end;
writeln(b); readln;
end.
	var n,a:longint;
	function f(x,y:longint):longint;
	var z:longint;
	begin
	if y=0 then f:=1
	else if y mod 2 = 0 then f:=f(x*x,y div 2)
	else f:=f(x,y-1)*x;
	end;
	begin
	readln(a,n);
	writeln(f(a,n)); readln;
	end.