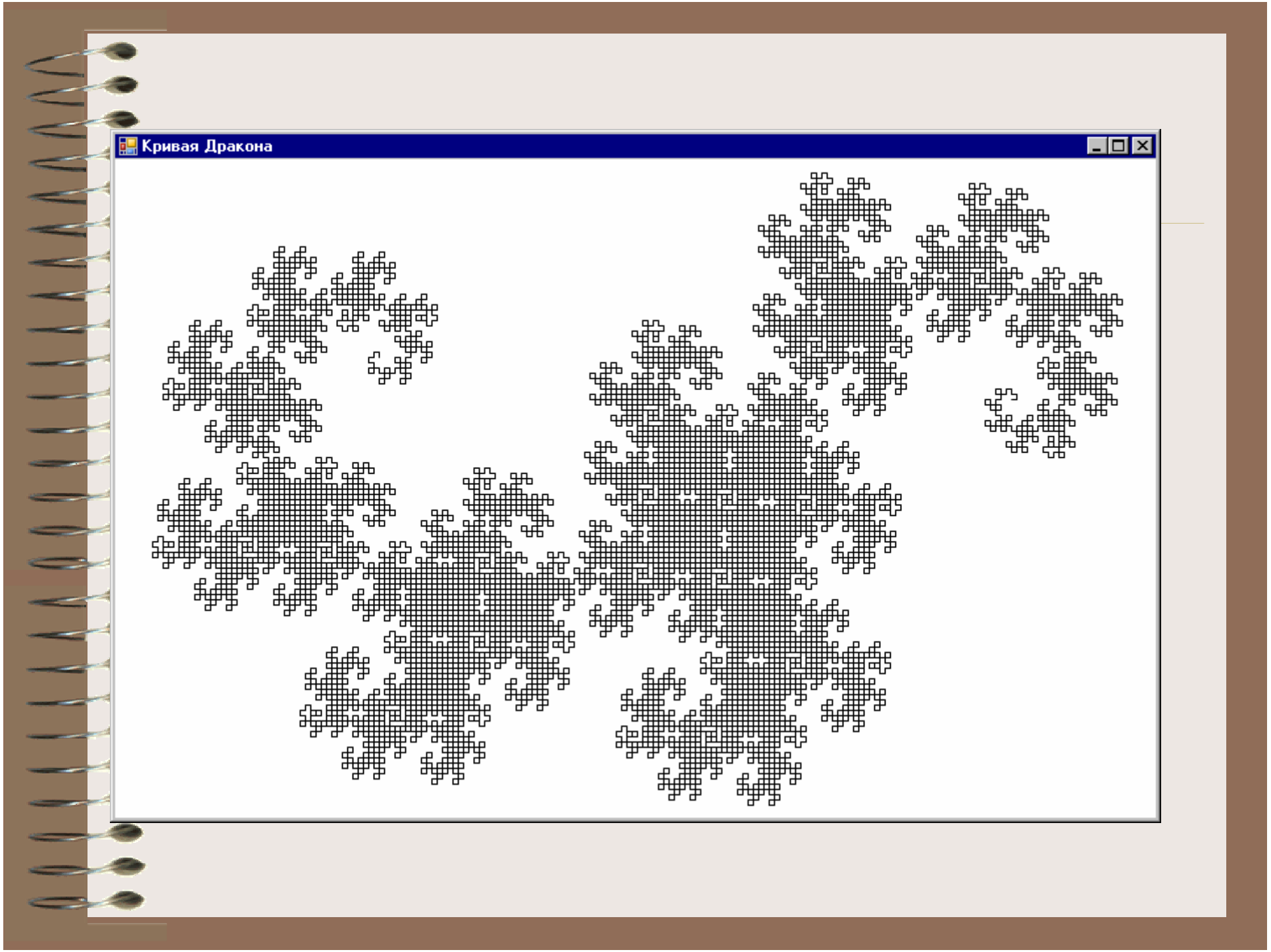
Рекурсия

Рекурсивный подход в программировании

2014


Кривая Дракона

Рекурсия – это способ определения объекта, при котором он частично определяется через такой же объект.

Определение, содержащее рекурсию, называется *рекурсивным определением*.

Примеры

- <идентификатор> ::= <буква>|
  <идентификатор><буква>|
  <идентификатор><цифра>|
  <идентификатор>_
- <Список фактических параметров>::=
  <параметр>|
  <Список фактических параметров> ,
  <параметр>

Примеры

$$n!=\begin{cases} n\cdot(n-1)!, \text{ если } n>0, \\ 1, \text{ если } n=0. \end{cases}$$

$$a^n=\begin{cases} a\cdot a^{n-1}, \text{ если } n>0, \\ 1, \text{ если } n=0. \end{cases}$$

В программировании *рекурсия* – это описание подпрограммы, содержащее прямой или косвенный вызов этой подпрограммой самой себя.

Простые примеры

Пример 1.

```
void p()
{
 cout<<1;
  p();
}
```

Вывод: 11111111..... - зацикливание

Простые примеры

Пример 2.

```
void p(int n)
{
   cout<<n<<' ';
   if (n>0)  p(n-1);
}
```

Вывод при вызове `p(5)`: 5 4 3 2 1 0

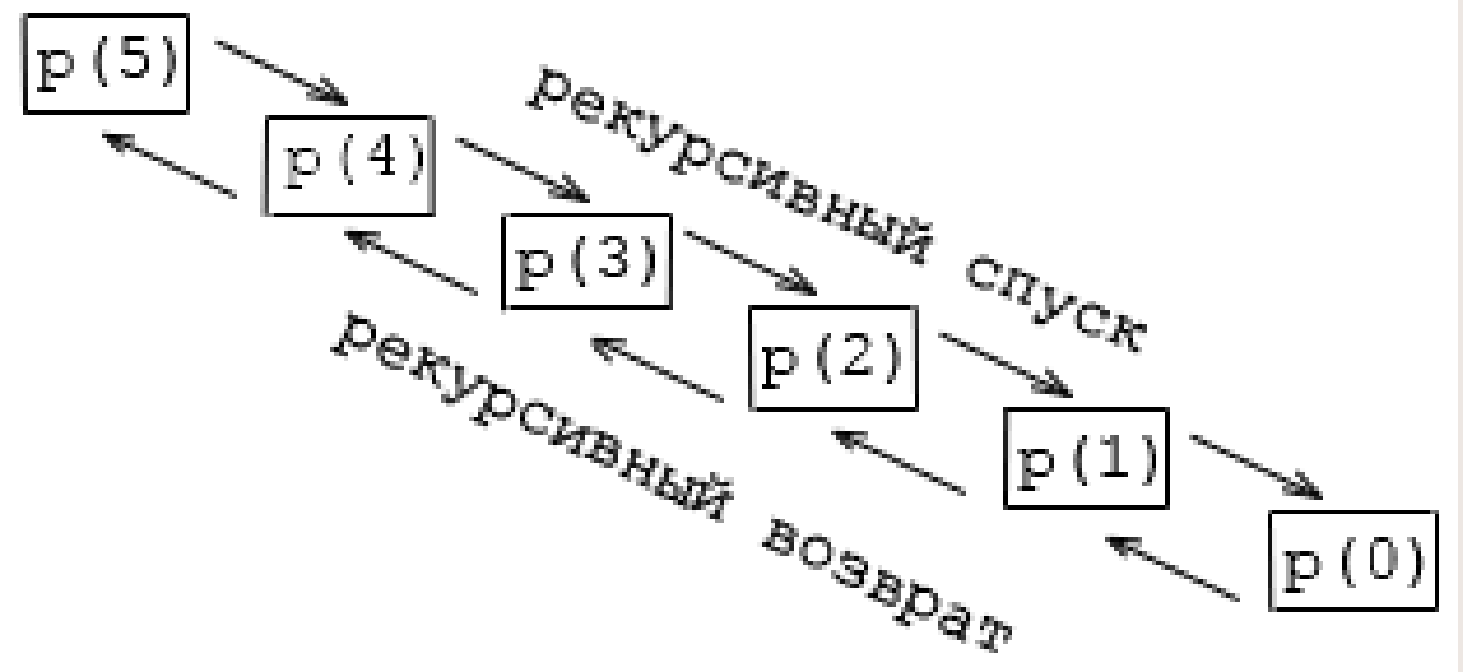
Простые примеры

Пример 3.

```
void p(int n)
{
  if (n>0)  p(n-1);
  cout<<n<<' ';
}
```

Вывод при вызове `p(5)`: 0 1 2 3 4 5

Схема рекурсивных вызовов в примерах 2 и 3



Пример 4.

Вычисление n!

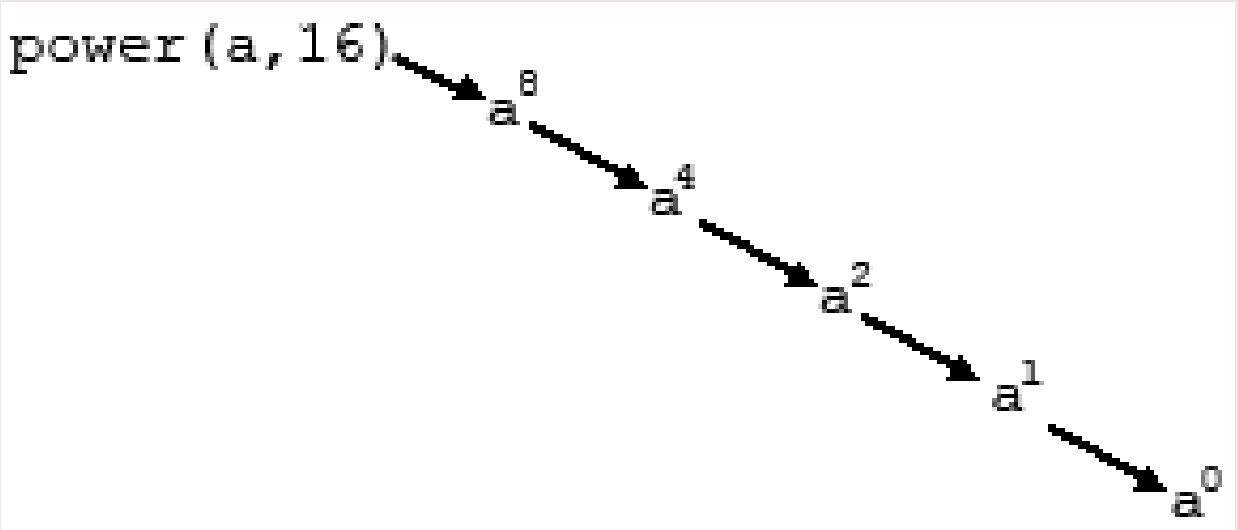
```
int  fact ( int n)
{
    if (n == 0) return 1;
      return n * fact(n-1);
}
```

Пример 5.

$$a^n = \begin{cases} 1, \text{ если } n = 0, \\ (a^{n/2})^2, \text{ если } n > 0,\ n - \text{четное}, \\ a \cdot a^{n-1}, \text{ если } n > 0,\ n - \text{нечетное}, \\ 1/a^n, \text{ если } n < 0. \end{cases}$$

```
bool odd (int n);
double sqr (double x);
double power(double a, int n)
{
   if (n==0)    return 1;
   if (n<0)    return power(a,-n);
   if (odd(n)) return a*power(a,n-1);
   return sqr(power(a, n/2));
}
```

Глубина рекурсии



Глубина рекурсии = 5

Для power(a,15) глубина рекурсии = 7

Виды рекурсивных подпрограмм

1) Действия осуществляется на рекурсивном спуске

```
void p (int n)
{
   S(n);
   if (B(n))  p(n-1);
}
```

Виды рекурсивных подпрограмм

2) Действия осуществляются на рекурсивном возврате.

```
void p (int n)
{
  if (B(n))  p(n-1);
  S(n); // отложенные действия

}
```

Виды рекурсивных подпрограмм

3) Действия осуществляются на рекурсивном спуске и возврате.

```
void p (int n)
{
   S1(n);
   if (B(n))  p(n-1);
   S2(n);
}
```

Виды рекурсивных подпрограмм

4) Каскадная рекурсия.

```
void p  (int n)
{
    S(n);
    if (B1(n)) p(n-1);
    if (B2(n)) p(n-1);
}
```

Виды рекурсивных подпрограмм

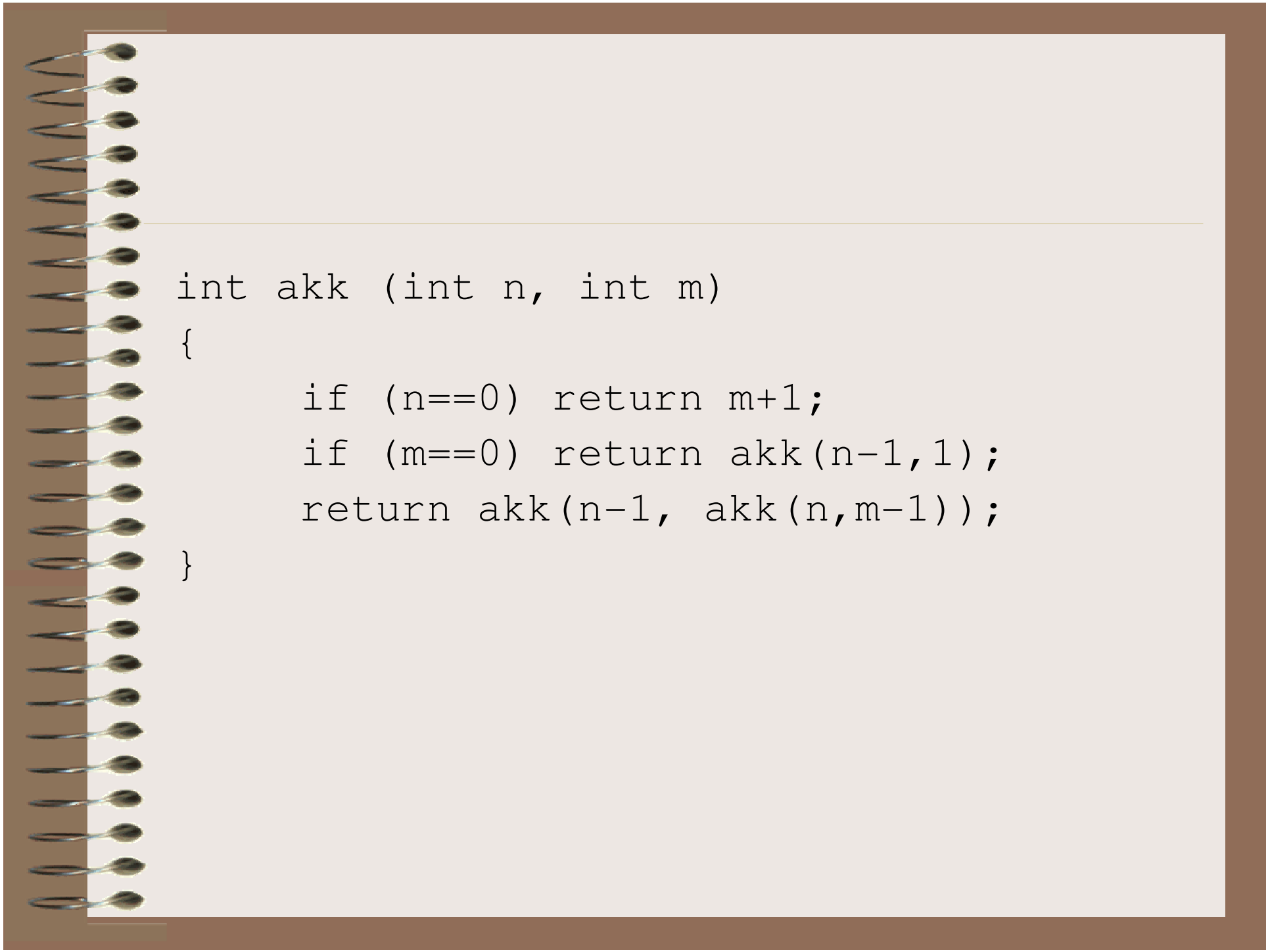
5) Удаленная рекурсия.

```
int f (int i)
{
  if (B1(i)) return ...
  return f(f(i-1));
}
```

Пример удаленной рекурсии

Функция Аккермана

$$A(n,m)=\begin{cases} m+1, \text{ если } n=0, \\ A(n-1,1), \text{ если } n>0, m=0, \\ A(n-1,A(n,m-1)), \text{ если } n>0, m>0. \end{cases}$$



```
int akk (int n, int m)
{
      if (n==0) return m+1;
      if (m==0) return akk(n-1,1);
      return akk(n-1, akk(n,m-1));
}
```

Пример плохого использования

$$f_n = \begin{cases} 1, \; n = 1,2, \\ f_{n-1} + f_{n-2}, \; n > 2. \end{cases}$$

Дерево рекурсивных вызовов

При вызове Fib(7)