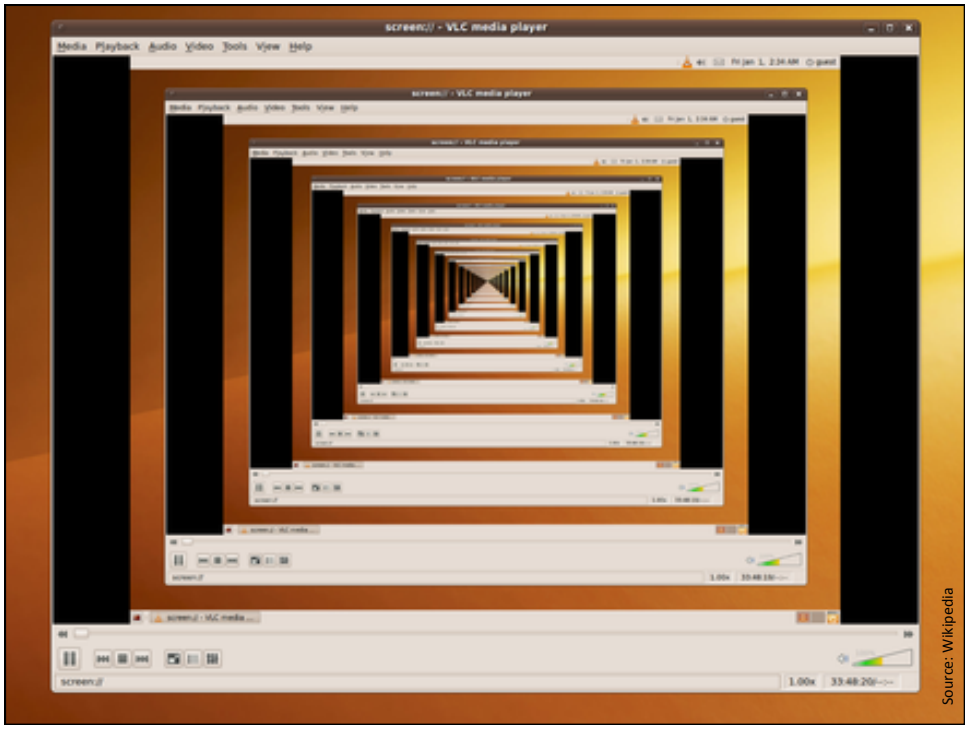
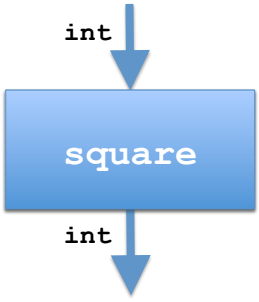
RECURSIVE METHODS



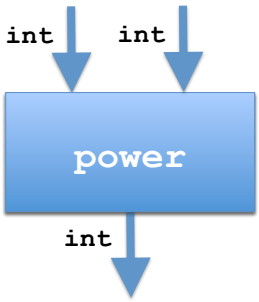
Square: x^2

```
int square (int x)
{
  return x*x;
}
```



Power: x^y

```
int power (int x, int y)
{
  …
}
```



Power: x^y

```
int power (int x, int y)
{
  if (y==0)       {return 1;}
  else if (y==1) {return x;}
  else if (y==2) {return square(x);}
  else if (y==3) {return cube(x);}
  … // until when?
}
```

Power: x^y

```
int power (int x, int y)
{
  if (y==0)       {return 1;}
  else if (y==1) {return x;}
  else if (y==2) {return x*x;}
  else if (y==3) {return x*x*x;}
  … // not very intelligent!
}
```

Power: x^y

Provided y>=0:

$$x^y = \underbrace{x * \ldots * x}_{y}$$

Power: x^y

```
int power (int x, int y)
{ // y>=0
  int z=1;
  for (int i=1; i<=y; i++)
    {z=x*z;}
  return z;
}
```

Table for `y` equal 3

i	i<=y	z	return
		1	
1	true	x	
2	true	x^2	
3	true	x^3	
4	false		
			x^3

Table for `y` equal 0

i	i<=y	z	return
		1	
1	false		
			1

Recurrence Equation

- Precondition: **y>=0**

- $x^y = 1$ **if (y==0)**
- $x^y = x * x^{y-1}$ **if (y>0)**

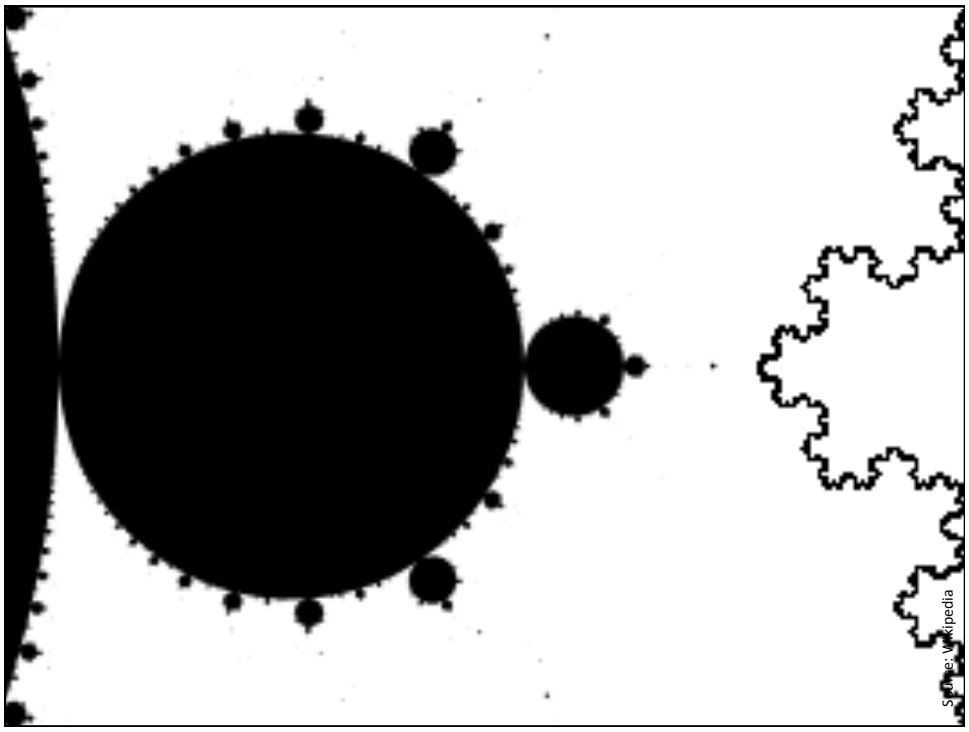
- **power(x,y)=1** **if (y==0)**
- **power(x,y)=**
  x*power(x,y-1) **if (y>0)**

Precondition satisfied ← y-1>=0 ← y>=1 ← y>0

Recursive Method

```
int power (int x, int y)
{ // y>=0
  if (y==0)
    return 1;
  else
    return x*power(x,y-1);
}
```

Can we do that?


Source: Wikipedia