File Edit View Help

File Edit View Help

Recursion
With Fractals

Recursion

- Solve a problem by solving smaller versions of the same problem
  - Divide and Conquer Algorithms
  - Backtracking
- Recursive Method – a method that calls itself
  - "Loop-like"
  - Call stack
- Recursive Methods Required
  - Halting Condition
  - Recursive Call

Example

```
public static void countDown(int i)
{
     if(i < 0 )//Halting Condition
          return;
     System.out.println(i);
     countDown(i-1);//Recursive Call
}
```

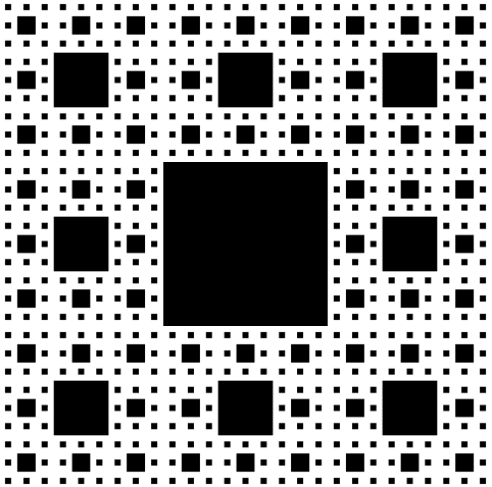
File Edit View Help

Sierpinski's Carpet

Concept

- Cut area in to 9 equal squares
  - 3 Horizontal
  - 3 Vertical
- Fill in the Center Square
- Repeat this process for the 8 surrounding squares until a limit has been reached
  - Recursive Depth
  - Pixel Limit

Example

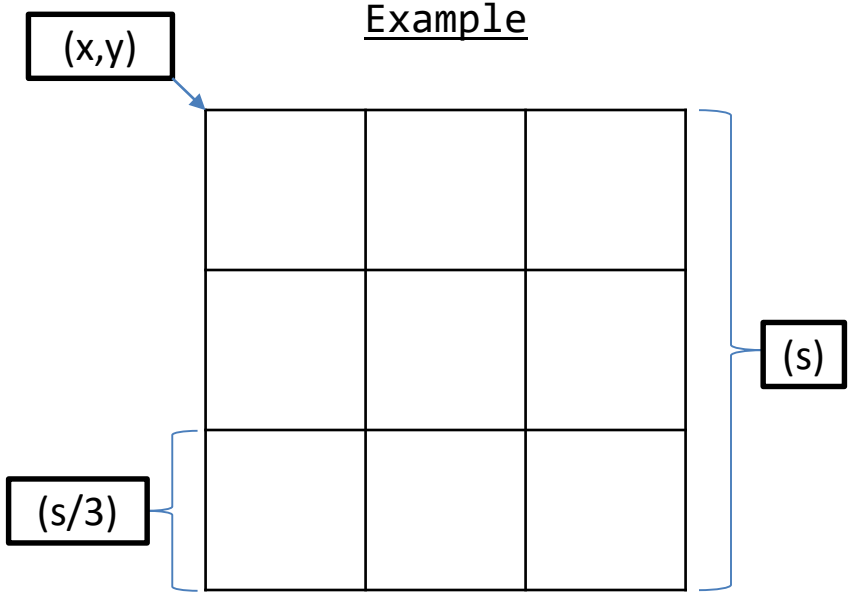


Sierpinski's Carpet

Solution of the Small Problem

- Cut area in to 9 equal squares given the length (s) of a side and a starting top left coordinates (x,y)
  - 3 Horizontal
  - 3 Vertical
- Fill in the Center Square using that length
  - Assuming drawing requires Top Left Coordinates
  - Size is s/3 x s/3
  - Draw from Top left using the Size

Example

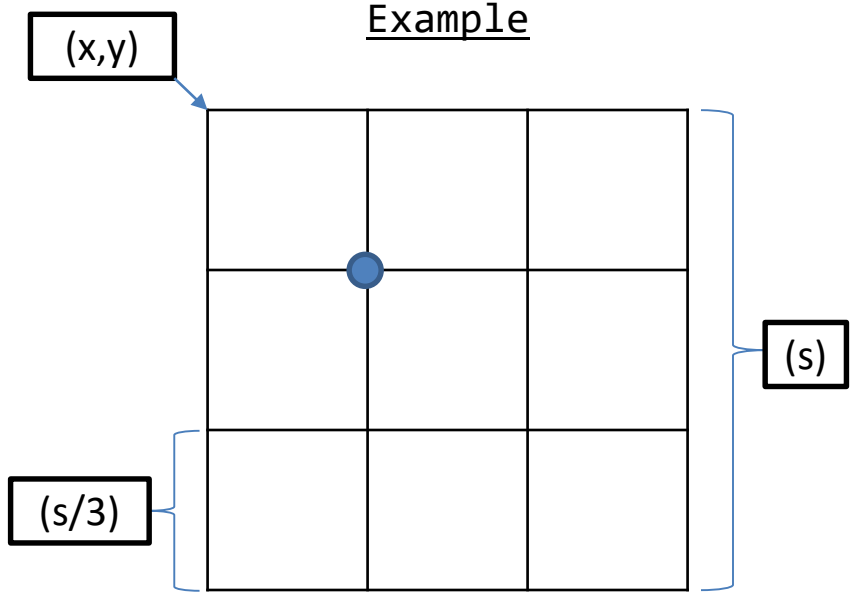


Sierpinski's Carpet

Solution of the Small Problem

- Cut area in to 9 equal squares given the length (s) of a side and a starting top left coordinates (x,y)
  - 3 Horizontal
  - 3 Vertical
- Fill in the Center Square using that length
  - Assuming drawing requires Top Left Coordinates
  - Size is s/3 x s/3
  - Draw from Top left using the Size

Example

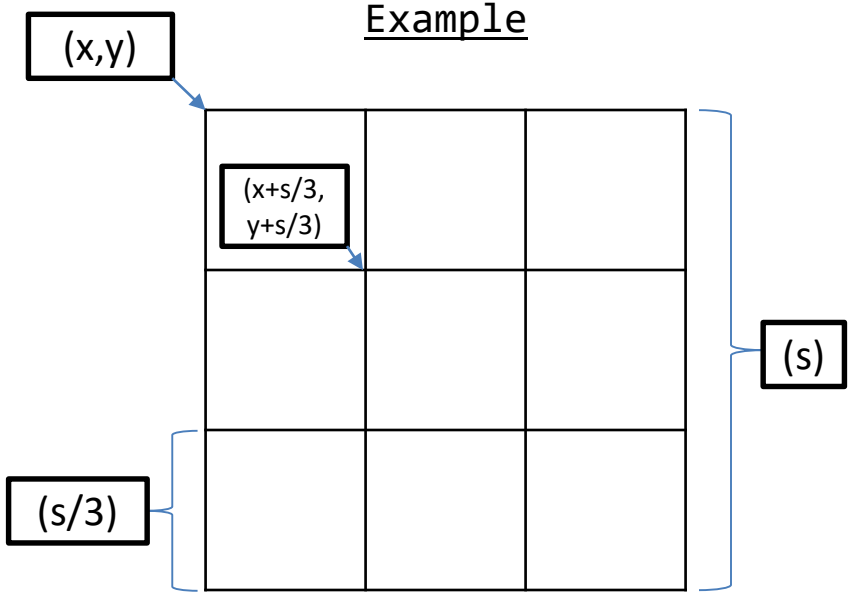


Sierpinski's Carpet

Solution of the Small Problem

- Cut area in to 9 equal squares given the length (s) of a side and a starting top left coordinates (x,y)
  - 3 Horizontal
  - 3 Vertical
- Fill in the Center Square using that length
  - Assuming drawing requires Top Left Coordinates
  - Size is s/3 x s/3
  - Draw from Top left using the Size

Example

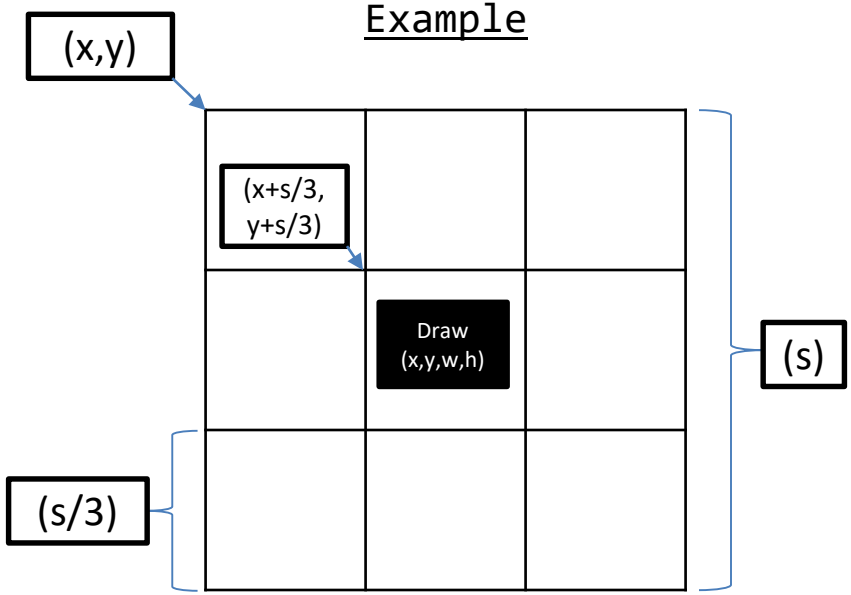


Sierpinski's Carpet

Solution of the Small Problem

- Cut area in to 9 equal squares given the length (s) of a side and a starting top left coordinates (x,y)
  - 3 Horizontal
  - 3 Vertical
- Fill in the Center Square using that length
  - Assuming drawing requires Top Left Coordinates
  - Size is s/3 x s/3
  - Draw from Top left using the Size

Example

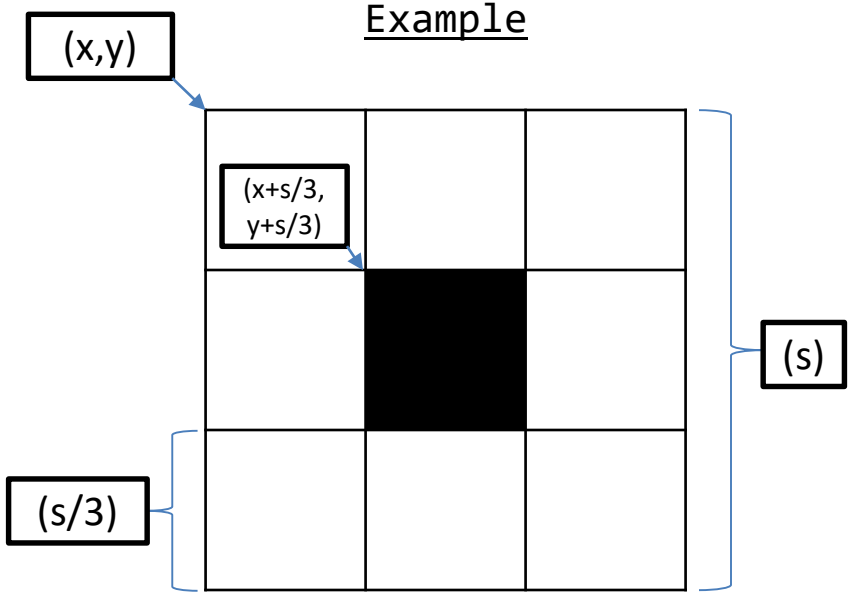


Sierpinski's Carpet

Solution of the Small Problem

- Cut area in to 9 equal squares given the length (s) of a side and a starting top left coordinates (x,y)
  - 3 Horizontal
  - 3 Vertical
- Fill in the Center Square using that length
  - Assuming drawing requires Top Left Coordinates
  - Size is s/3 x s/3
  - Draw from Top left using the Size

Example

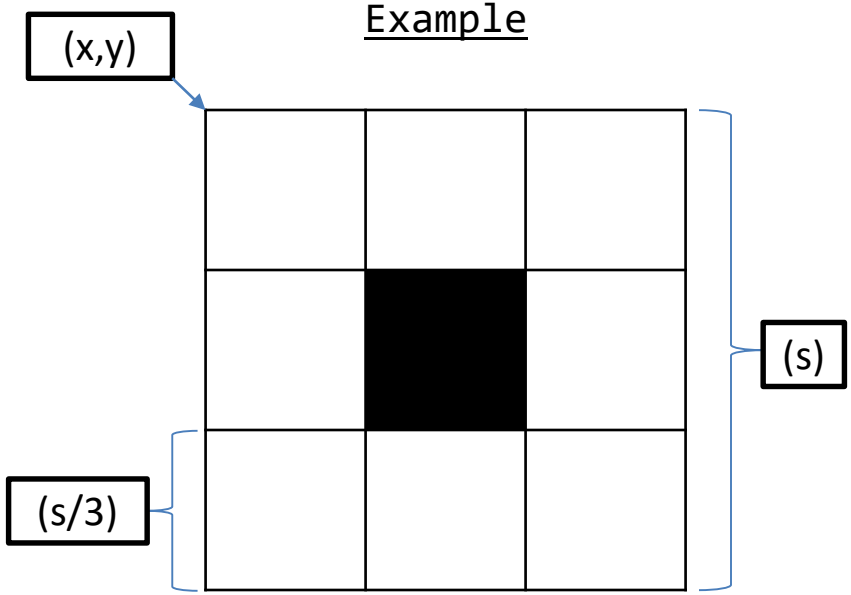


Sierpinski's Carpet

Using Recursion to Solve Bigger Problem

- Repeat this process for the 8 surrounding squares until a limit has been reached
  - If a pixel limit or recursive depth has been reached then return (Halting Condition)
  - Assume recursive method is ordered (x-coordinate, y-coordinate, length of the side)
  - Top Left (x,y,s/3)
  - Top Middle (x+s/3, y, s/3)
  - Top Right(x+s*2/3, y, s/3)
  - Middle Left(x, y+s/3, s/3)
  - Middle Right(x+s*2/3, y+s/3, s/3)
  - Bottom Left(x, y+s*2/3, s/3)
  - Bottom Middle(x+s/3, y+s*2/3, s/3)
  - Bottom Right(x+s*2/3, , y+s*2/3, s/3)

Example

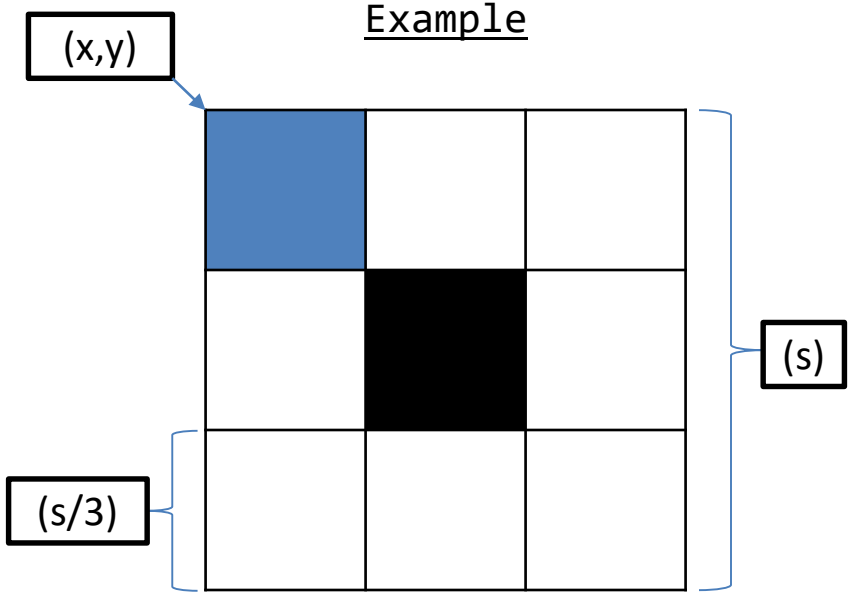


Sierpinski's Carpet

Using Recursion to Solve Bigger Problem

- Repeat this process for the 8 surrounding squares until a limit has been reached
  - If a pixel limit or recursive depth has been reached then return (Halting Condition)
  - Assume recursive method is ordered (x-coordinate, y-coordinate, length of the side)
  - Top Left (x,y,s/3)
  - Top Middle (x+s/3, y, s/3)
  - Top Right(x+s*2/3, y, s/3)
  - Middle Left(x, y+s/3, s/3)
  - Middle Right(x+s*2/3, y+s/3, s/3)
  - Bottom Left(x, y+s*2/3, s/3)
  - Bottom Middle(x+s/3, y+s*2/3, s/3)
  - Bottom Right(x+s*2/3, , y+s*2/3, s/3)

Example

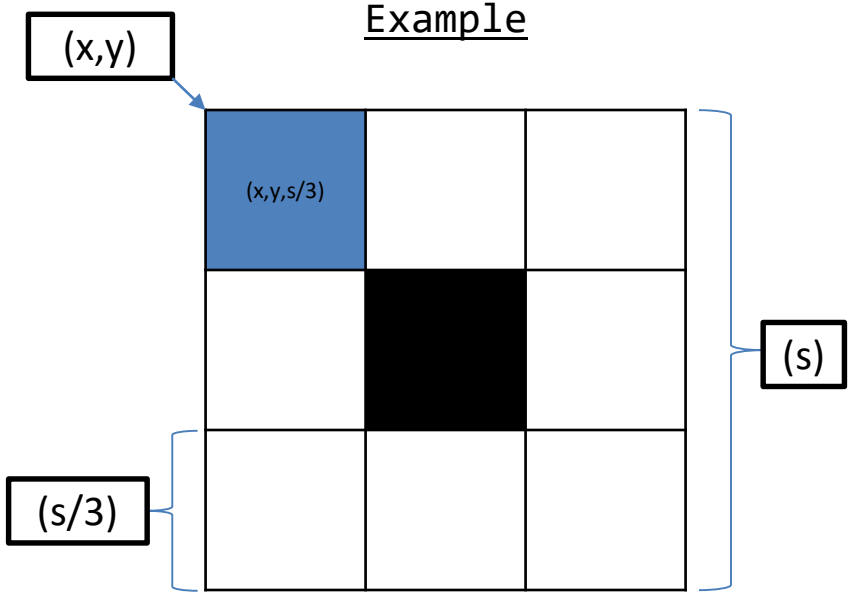


Sierpinski's Carpet

Using Recursion to Solve Bigger Problem

- Repeat this process for the 8 surrounding squares until a limit has been reached
  - If a pixel limit or recursive depth has been reached then return (Halting Condition)
  - Assume recursive method is ordered (x-coordinate, y-coordinate, length of the side)
  - Top Left (x,y,s/3)
  - Top Middle (x+s/3, y, s/3)
  - Top Right(x+s*2/3, y, s/3)
  - Middle Left(x, y+s/3, s/3)
  - Middle Right(x+s*2/3, y+s/3, s/3)
  - Bottom Left(x, y+s*2/3, s/3)
  - Bottom Middle(x+s/3, y+s*2/3, s/3)
  - Bottom Right(x+s*2/3, , y+s*2/3, s/3)

Example

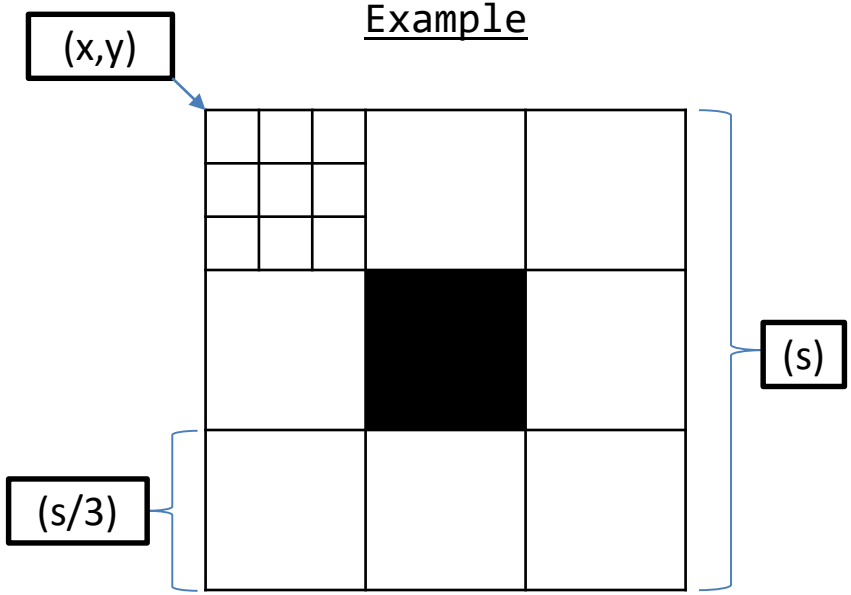


Sierpinski's Carpet

Using Recursion to Solve Bigger Problem

- Repeat this process for the 8 surrounding squares until a limit has been reached
  - If a pixel limit or recursive depth has been reached then return (Halting Condition)
  - Assume recursive method is ordered (x-coordinate, y-coordinate, length of the side)
  - Top Left (x,y,s/3)
  - Top Middle (x+s/3, y, s/3)
  - Top Right(x+s*2/3, y, s/3)
  - Middle Left(x, y+s/3, s/3)
  - Middle Right(x+s*2/3, y+s/3, s/3)
  - Bottom Left(x, y+s*2/3, s/3)
  - Bottom Middle(x+s/3, y+s*2/3, s/3)
  - Bottom Right(x+s*2/3, , y+s*2/3, s/3)

Example

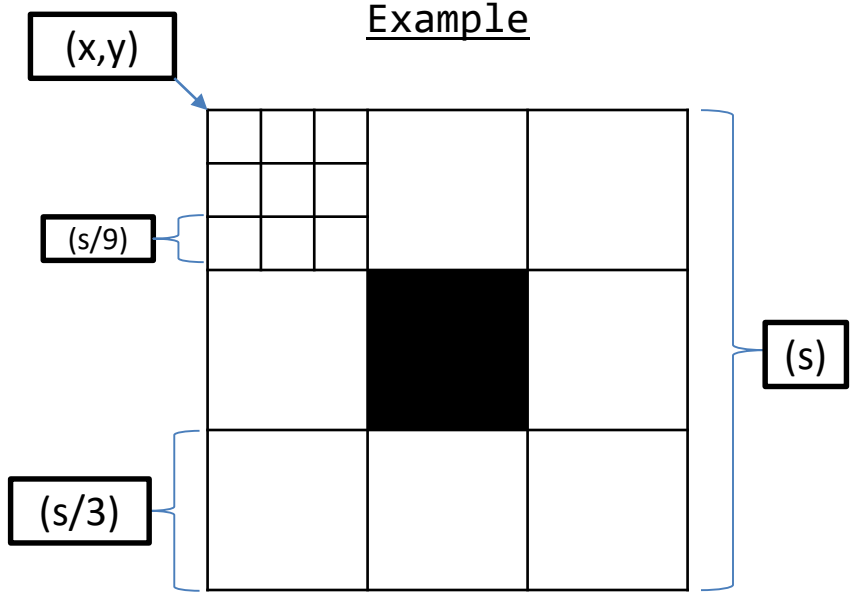


Sierpinski's Carpet

Using Recursion to Solve Bigger Problem

- Repeat this process for the 8 surrounding squares until a limit has been reached
  - If a pixel limit or recursive depth has been reached then return (Halting Condition)
  - Assume recursive method is ordered (x-coordinate, y-coordinate, length of the side)
  - Top Left (x,y,s/3)
  - Top Middle (x+s/3, y, s/3)
  - Top Right(x+s*2/3, y, s/3)
  - Middle Left(x, y+s/3, s/3)
  - Middle Right(x+s*2/3, y+s/3, s/3)
  - Bottom Left(x, y+s*2/3, s/3)
  - Bottom Middle(x+s/3, y+s*2/3, s/3)
  - Bottom Right(x+s*2/3, , y+s*2/3, s/3)

Example

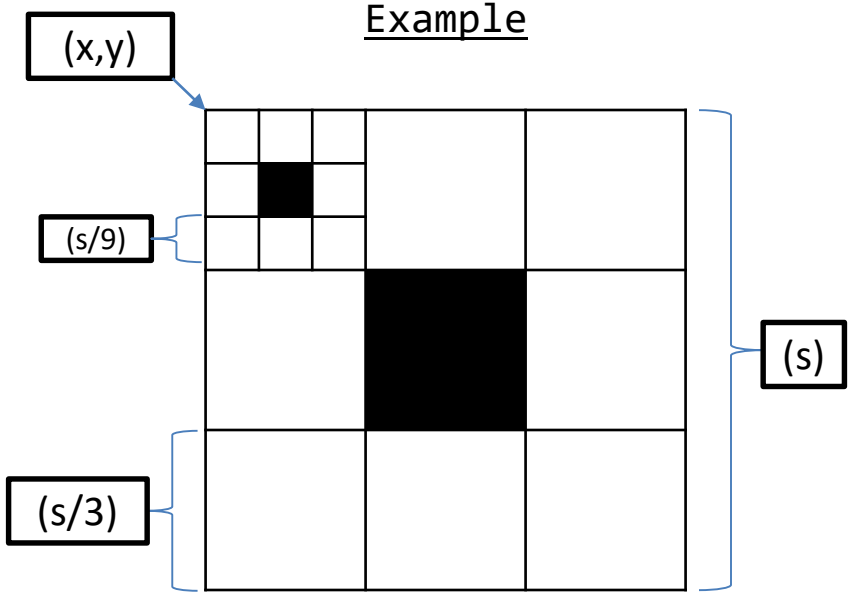


Sierpinski's Carpet

Using Recursion to Solve Bigger Problem

- Repeat this process for the 8 surrounding squares until a limit has been reached
  - If a pixel limit or recursive depth has been reached then return (Halting Condition)
  - Assume recursive method is ordered (x-coordinate, y-coordinate, length of the side)
  - Top Left (x,y,s/3)
  - Top Middle (x+s/3, y, s/3)
  - Top Right(x+s*2/3, y, s/3)
  - Middle Left(x, y+s/3, s/3)
  - Middle Right(x+s*2/3, y+s/3, s/3)
  - Bottom Left(x, y+s*2/3, s/3)
  - Bottom Middle(x+s/3, y+s*2/3, s/3)
  - Bottom Right(x+s*2/3, , y+s*2/3, s/3)

Example

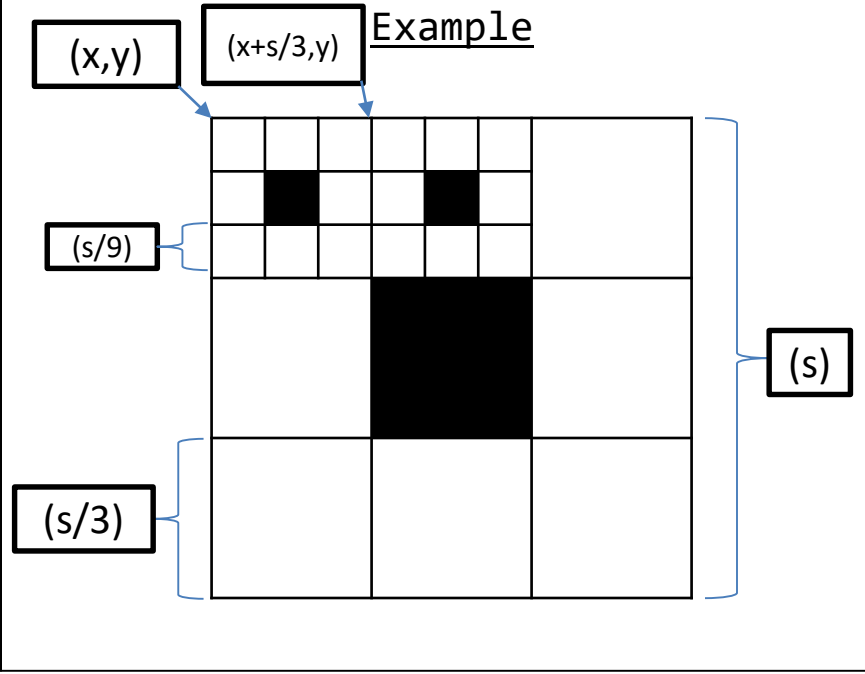


File Edit View Help

Sierpinski's Carpet

Using Recursion to Solve Bigger Problem

- Repeat this process for the 8 surrounding squares until a limit has been reached
  - If a pixel limit or recursive depth has been reached then return (Halting Condition)
  - Assume recursive method is ordered (x-coordinate, y-coordinate, length of the side)
  - Top Left (x,y,s/3)
  - Top Middle (x+s/3, y, s/3)
  - Top Right(x+s*2/3, y, s/3)
  - Middle Left(x, y+s/3, s/3)
  - Middle Right(x+s*2/3, y+s/3, s/3)
  - Bottom Left(x, y+s*2/3, s/3)
  - Bottom Middle(x+s/3, y+s*2/3, s/3)
  - Bottom Right(x+s*2/3, , y+s*2/3, s/3)



Sierpinski's Carpet

Using Recursion to Solve Bigger Problem

- Repeat this process for the 8 surrounding squares until a limit has been reached
  - If a pixel limit or recursive depth has been reached then return (Halting Condition)
  - Assume recursive method is ordered (x-coordinate, y-coordinate, length of the side)
  - Top Left (x,y,s/3)
  - Top Middle (x+s/3, y, s/3)
  - Top Right(x+s*2/3, y, s/3)
  - Middle Left(x, y+s/3, s/3)
  - Middle Right(x+s*2/3, y+s/3, s/3)
  - Bottom Left(x, y+s*2/3, s/3)
  - Bottom Middle(x+s/3, y+s*2/3, s/3)
  - Bottom Right(x+s*2/3, , y+s*2/3, s/3)

Example