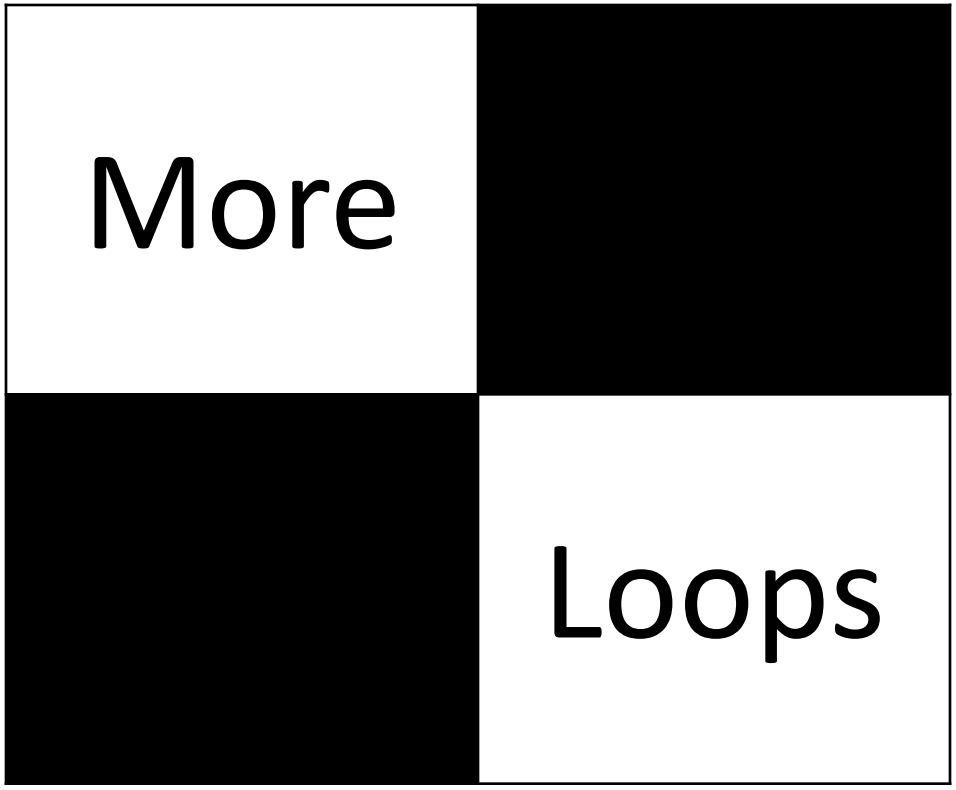
Announcements

- Project 1 will be posted today on webpage and cvs
  - Due Tuesday


More
Loops

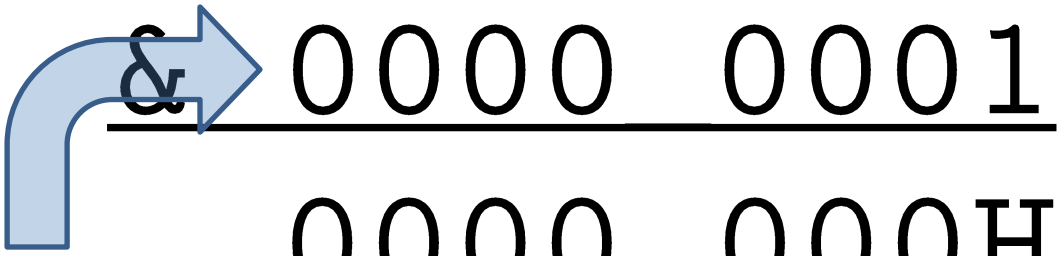
Printing in base 2

- A – H: either 0 or 1

```
   ABCD_EFGH
& 0000 0001
---------
   ????_????
```

Printing in base 2

- A – H: either 0 or 1

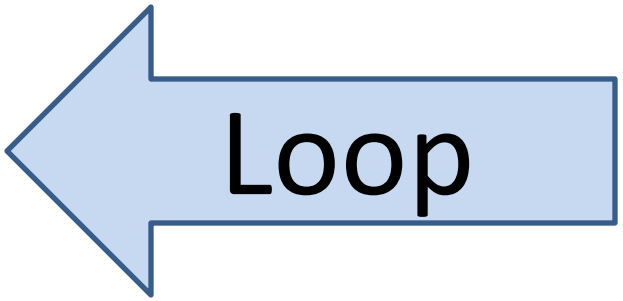


"Mask"

Printing in base 2

- A – H: either 0 or 1

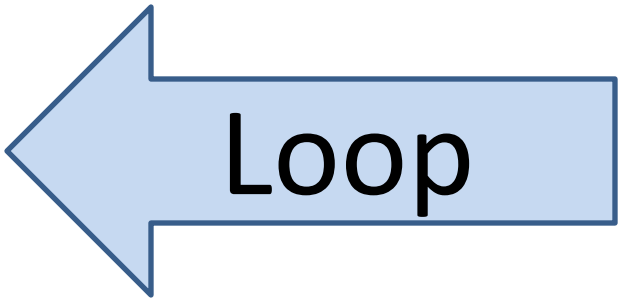
```
  ABCD_EFGH
& 0000 0001
-----------
  0000_000H
```



Printing in base 2

- A – H: either 0 or 1

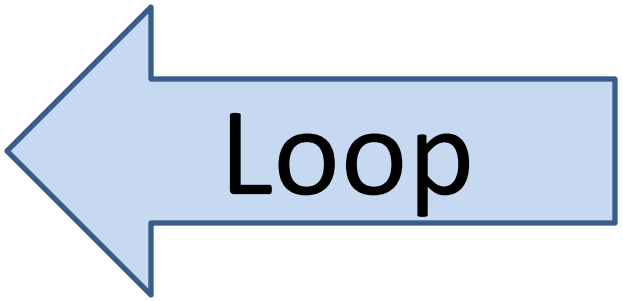
```
  ABCD_EFGH
& 0000 0010 <<
-----------
  0000_00G0
```



Printing in base 2

- A – H: either 0 or 1

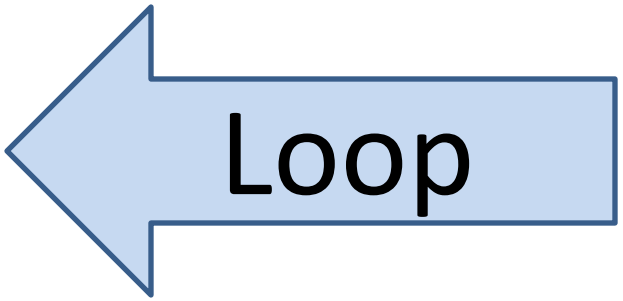
```
  ABCD_EFGH
& 0000_0100  <<
-----------
  0000_0F00
```



Printing in base 2

- A – H: either 0 or 1

```
  ABCD_EFGH
& 0000_1000  <<
-----------
  0000_E000
```



Printing in base 2

- A – H: either 0 or 1

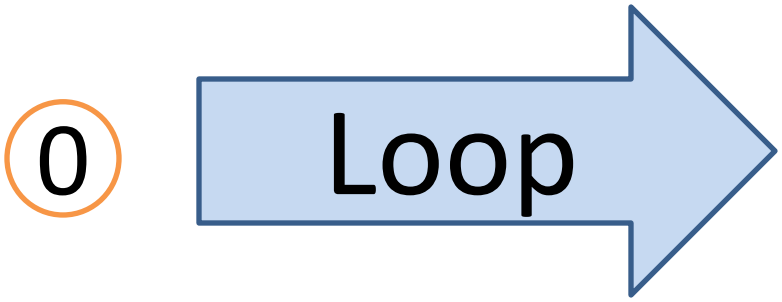
```
  ABCD_EFGH
& 0000 1000
-----------
  0000_E000
```

Another way?

Printing in base 2

- A – H: either 0 or 1

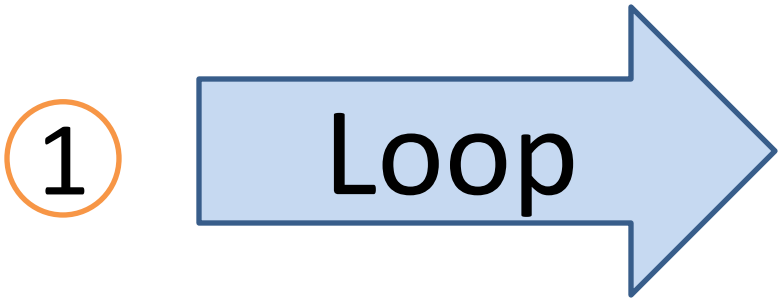
```
  ABCD_EFGH >>>
& 0000 0001
-----------
  0000_000H
```



Printing in base 2

- A – H: either 0 or 1

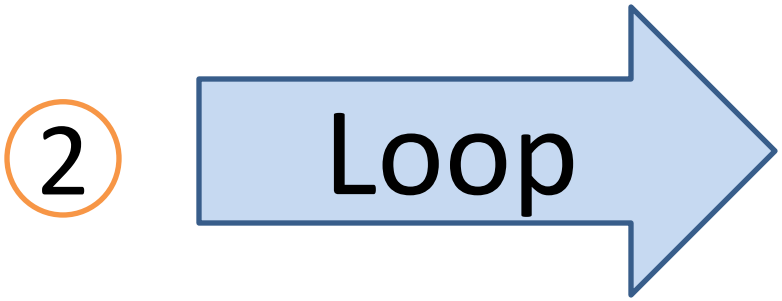
```
  0ABC_DEFG >>>
& 0000 0001
-----------
  0000_000G
```



Printing in base 2

- A – H: either 0 or 1

```
   00AB_CDEF >>>
& 0000 0001
-----------
   0000_000F
```



Printing in base 2

- A – H: either 0 or 1

```
   000A_BCDE >>>
& 0000 0001
-----------
   0000_000E
```

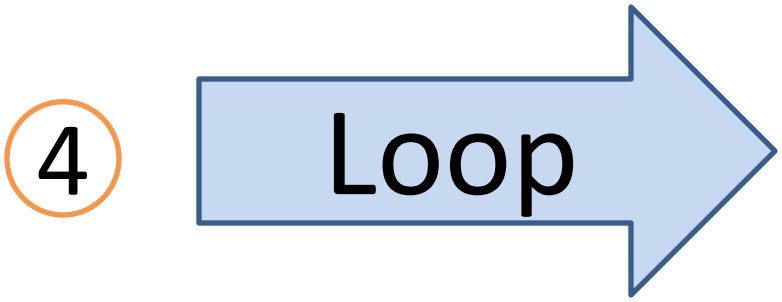
3

Loop

Printing in base 2

- A – H: either 0 or 1

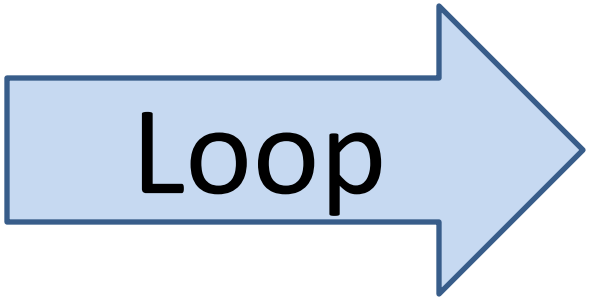
```
  0000_ABCD >>>
& 0000 0001
-----------
  0000_000D
```



Printing in base 2

- A – H: either 0 or 1

```
    ????_ABCD >>
& 0000 0001
---------------
    0000_000D
```



Bitwise shift operators

What arithmetic operation (+, -, *, /, %) do the following represent?

- n >> s == ??
- n & 1 == ??

Bitwise shift operators

What arithmetic operation (+, -, *, /, %) do the following represent?

- $n >> s == n / 2^s$
- $n \& 1 == n \% 2$

Printing in base B

- Base 2 (i^{th} digit): Shift then mask

Printing in base B

- Base 2 (i^{th} digit): (n >> i) & 1

Printing in base B

- Base 2 (i^{th} digit): $(n / 2^i) \% 2$

Printing in base B

- Base 2 (i^{th} digit): $(n / 2^i) \% 2$
- Base 10 (i^{th} digit): $(n / 10^i) \% 10$
- Base B (i^{th} digit): $(n / B^i) \% B$

Signed two's complement visualization

Primality testing

19 is divisible by 1

19 is not divisible by 2

19 is not divisible by 3

…

19 is not divisible by 17

19 is not divisible by 18

19 is divisible by 19

“While Loops”

```
while(condition) {
    do something
}
```

- *Init* : you handle this case by case
- *condition* : if this condition is true, repeat the loop body
- *Update* : you also handle this case by case
- *do something* : execute a sequence of commands

While loops

```
//A for loop
for(int i = 0; i < 10; i++) {
     //do something
}

//An equivalent while loop
int i = 0;
while(i < 10) {
     //do something
     i++;
}
```

“Do while Loops”

```
do {
    something
} while(condition);
```

- *condition* is checked *after* each repetition of the loop body, not before
  - In a do-while loop, the body is always executed at least once.

`break` and `continue`

- `break` : If encountered in a loop body, the loop will terminate immediately
- `continue` : If encountered in a loop body, the *current repetition* will terminate immediately; the next repetition will begin

While example

- Casino game with loops
  - User places a bet
  - User wins or loses based on biased coin flip
  - Repeat until user quits or has no money left

Nested loops

- Performing an iterative task with a loop

Nested loops

- Performing an iterative task with a loop
  - Every iteration requires its own repetitive sub-task

Nested loops

- Performing an iterative task with a loop
  - Every iteration requires its own repetitive sub-task
    - Every sub-task iteration requires its own repetitive sub-sub-task

Nested loops

- Performing an iterative task with a loop
  - Every iteration requires its own repetitive sub-task
    - Every sub-task iteration requires its own repetitive sub-sub-task
      - Every sub-sub-task iteration requires its own repetitive sub-sub-sub-task…

Nested loops

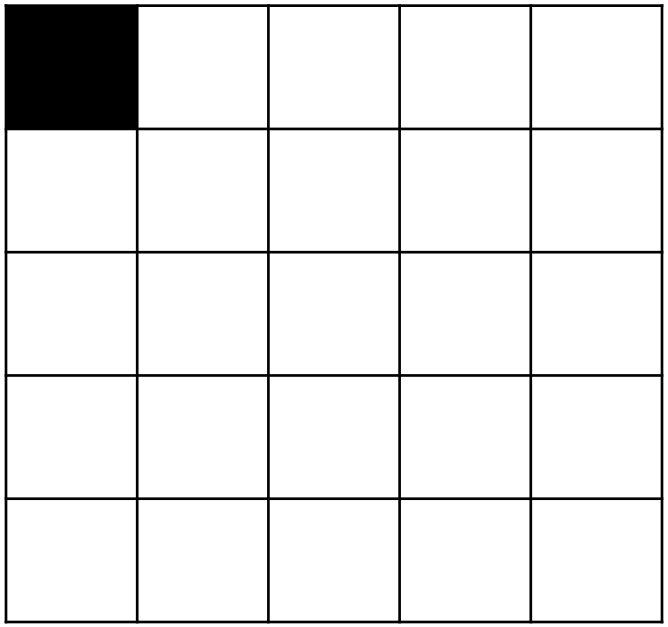
```
//printBinary with method call:
for(int num = 0; num < 32; num++) {
  printBinary(num, 8, '0', '1');
}

//with method code “inline”:
for(int num = 0; num < 32; num++) {
  for(int dig = 0; dig < numDigits; dig++) {
    System.out.print((num >> dig & 1) == 1 ? ‘1’ : ‘0’);
  }
  System.out.println();
}
```

Loops  Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 0

col 0

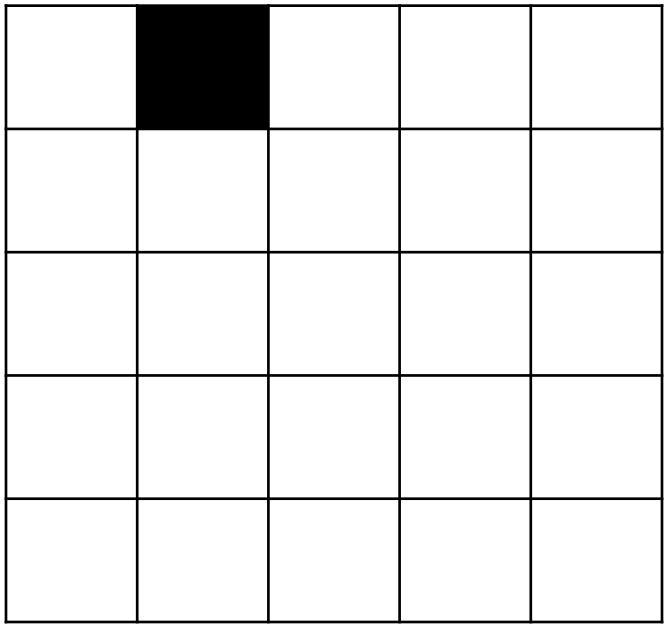


Loops ⟺ Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 0

col 1

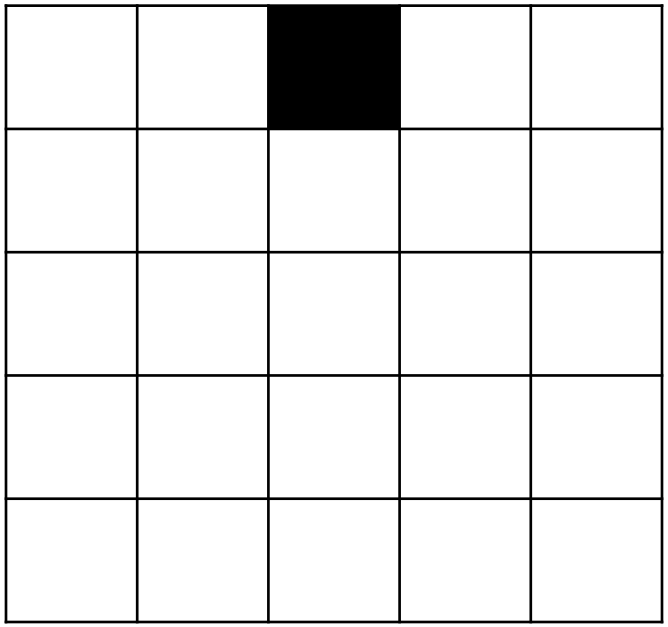


Loops ⟺ Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 0

col 2

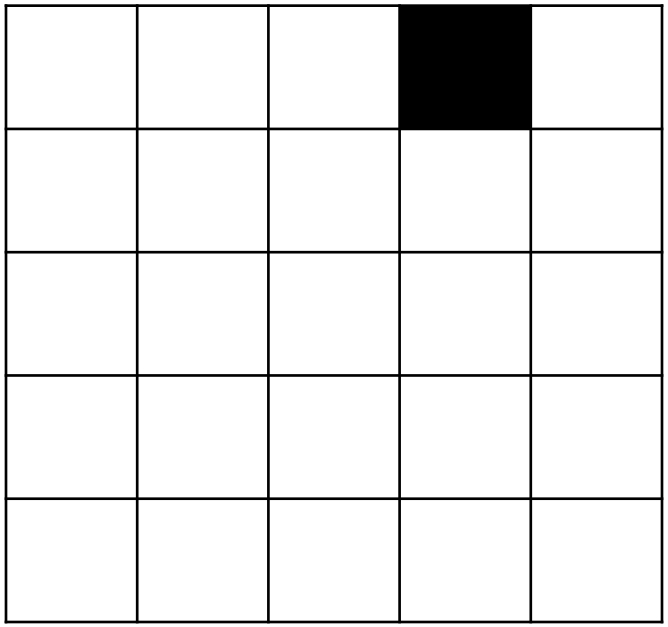


Loops ⟺ Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 0

col 3

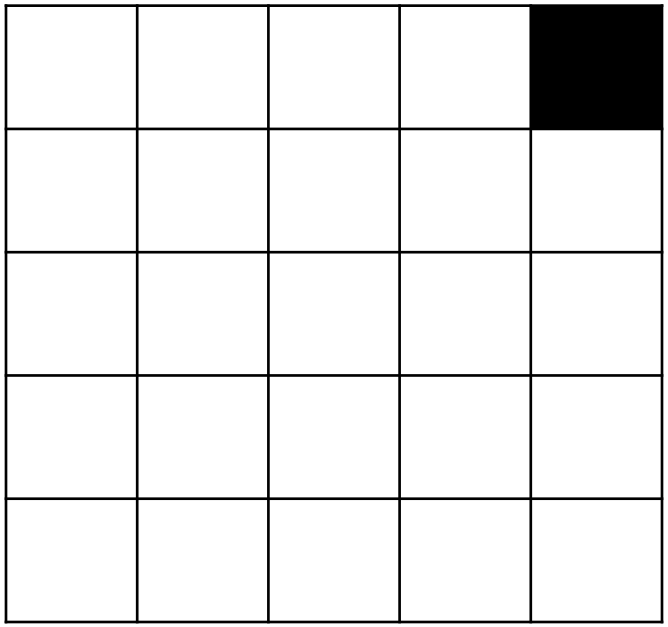


Loops ⟺ Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 0

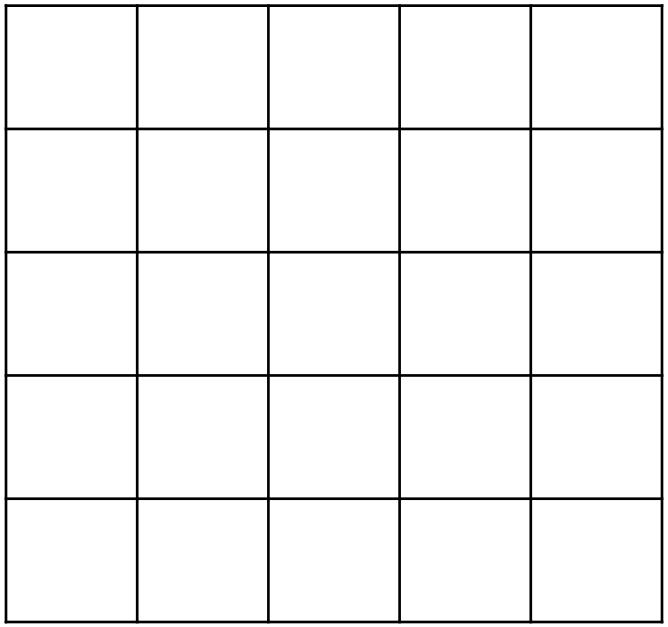
col 4



Loops ⟷ Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 1

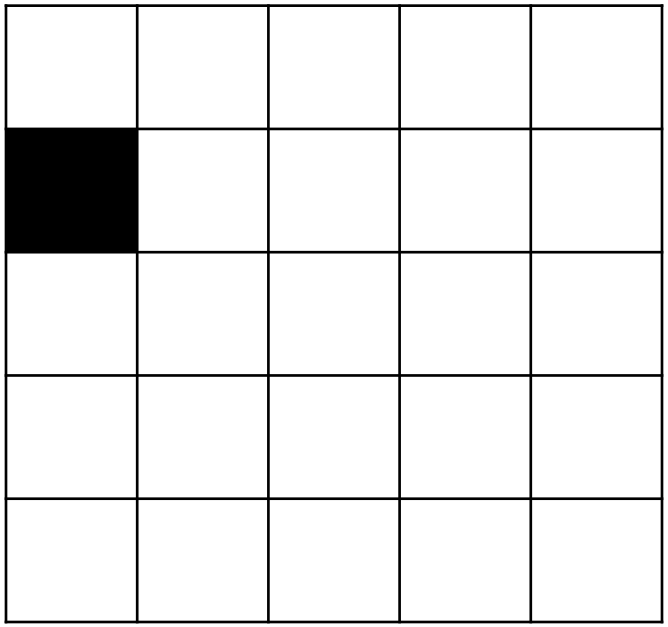


Loops ⟺ Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 1

col 0

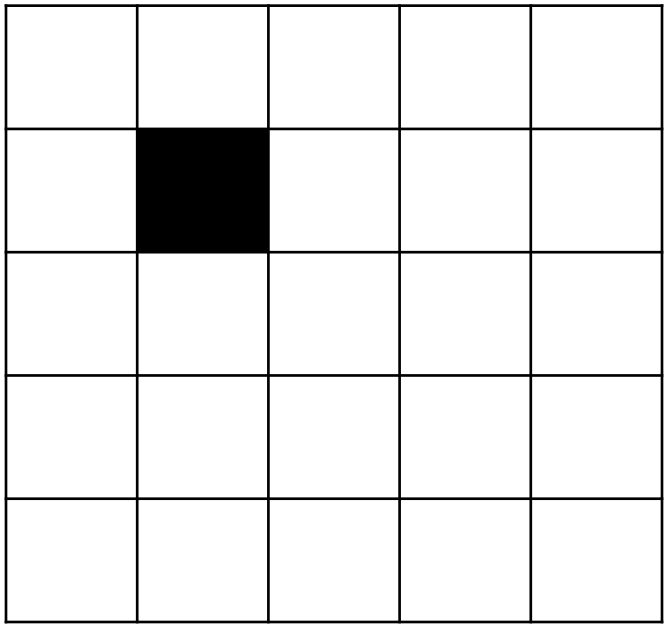


Loops ⟺ Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 1

col 1

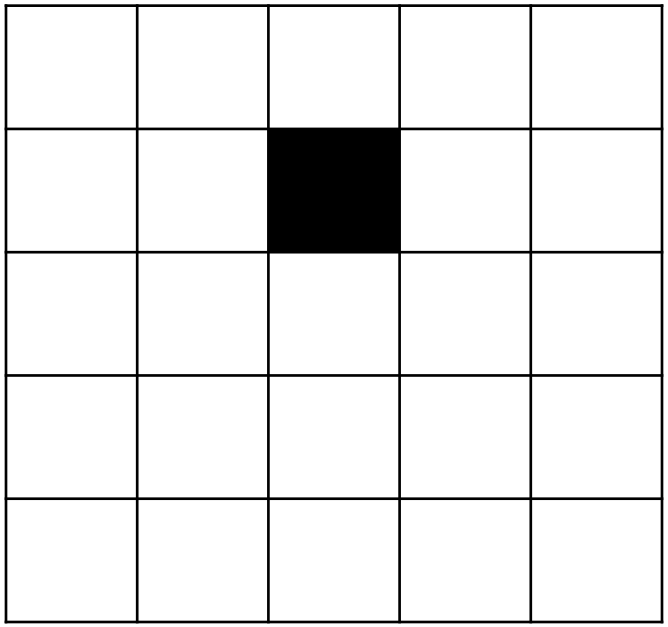


Loops ⟺ Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 1

col 2

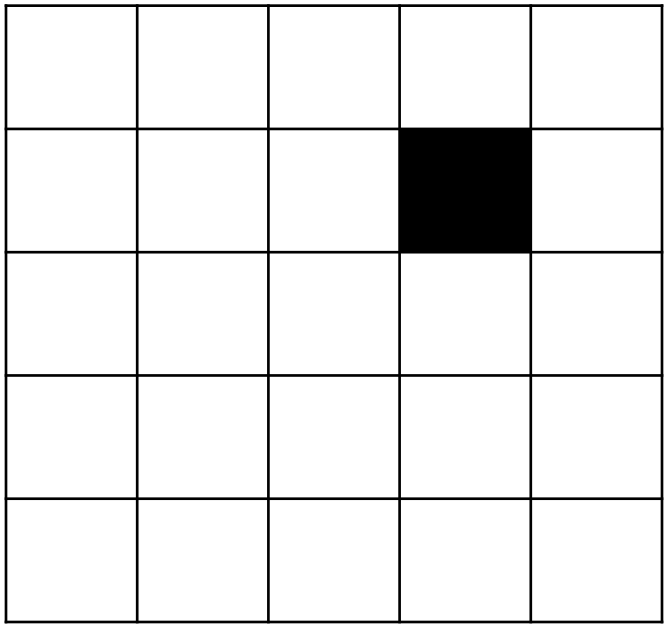


Loops ⟺ Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 1

col 3

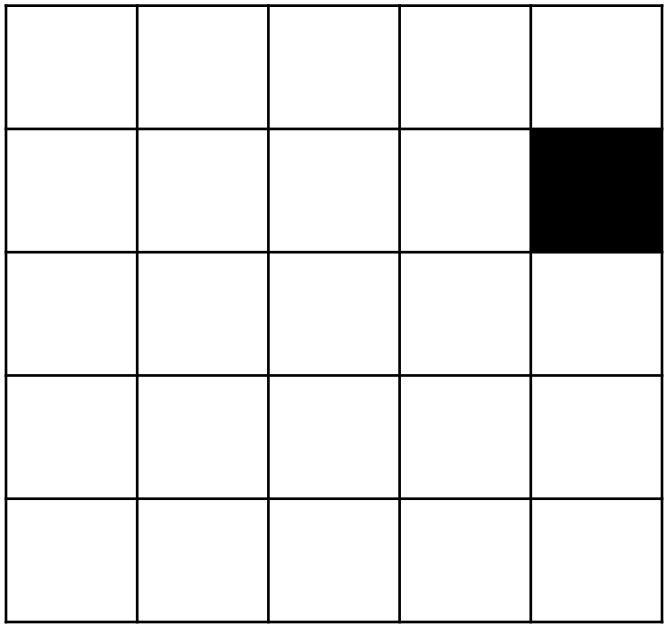


Loops ⟺ Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 1

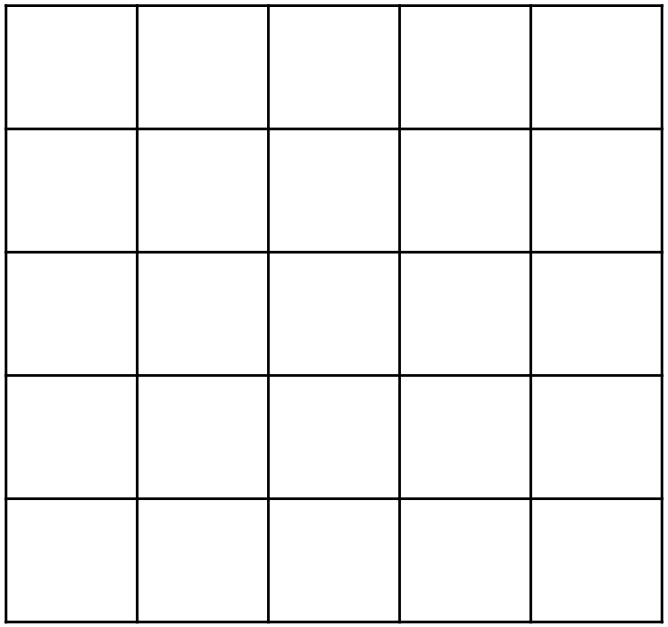
col 4



Loops ⟺ Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 2

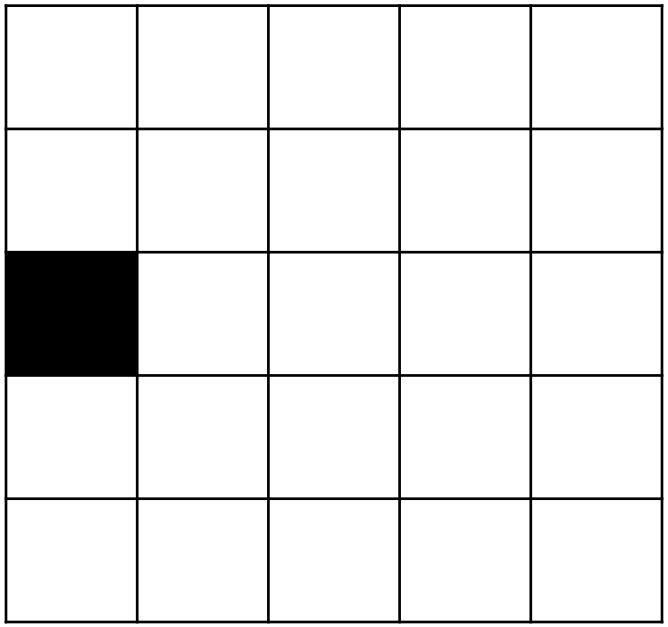


Loops ⟺ Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 2

col 0

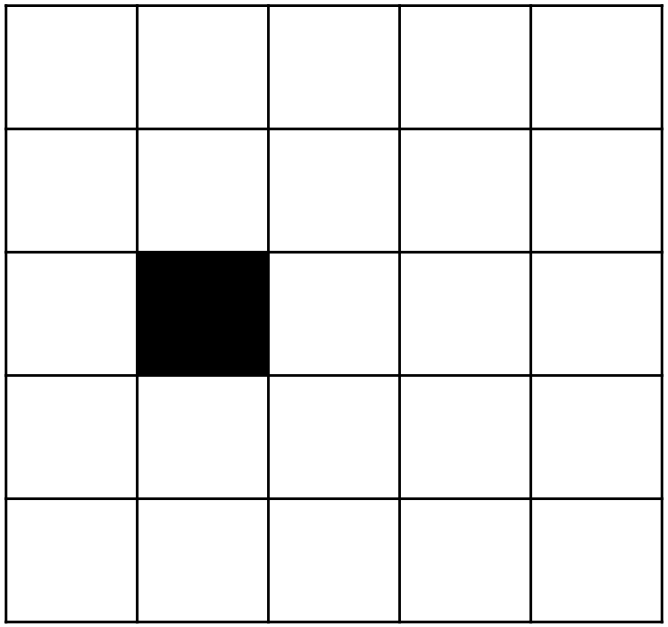


Loops ⟺ Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 2

col 1

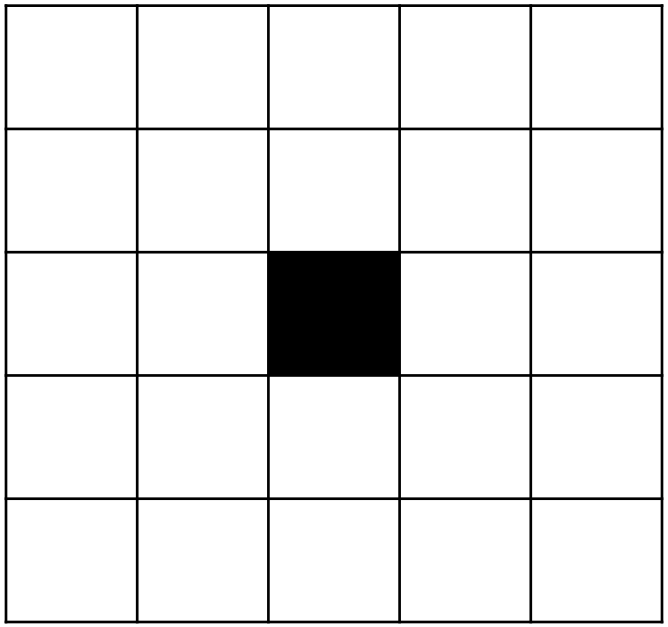


Loops ⟺ Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 2

col 2

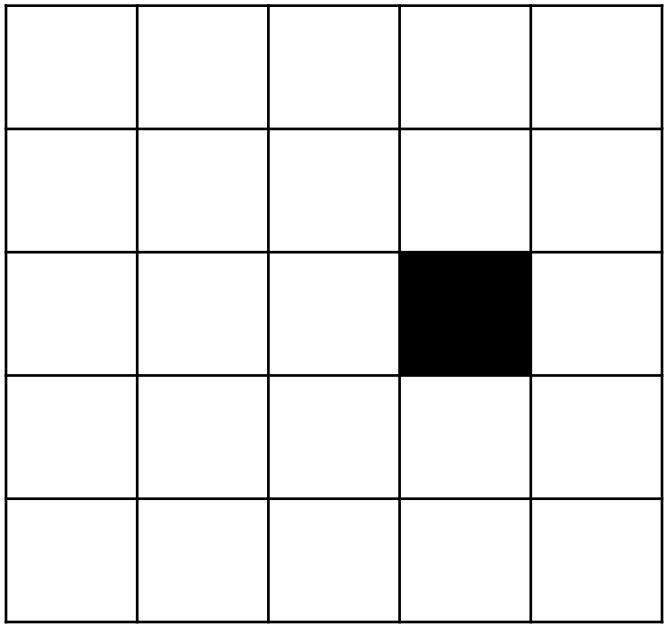


Loops ⟺ Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 2

col 3

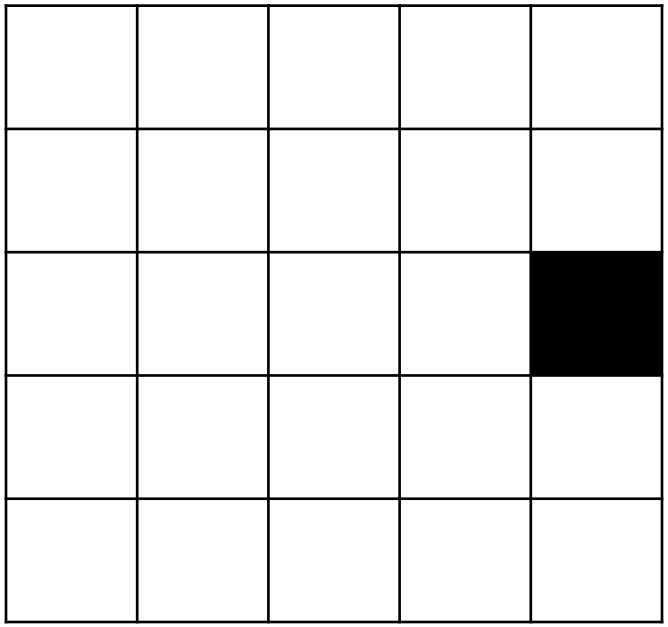


Loops ⟺ Dimensions

```
//2 layers: 2 dimensional
for(int row = 0; row < R; row++) {
  for(int col = 0; col < C; col++) {
    myMethod(row, col);
  }
}
```

row 2

col 4



Nested loops

- Examples: printing shapes in the console

```
#
##
###
####
```

  - Using loop boundary conditions
  - Using `myMethod(row, col)`

More Loop Examples

- Calculate e (Brothers)

$$e = \sum_{n=0}^{\infty} \frac{2n+2}{(2n+1)!}$$

- Calculate π (Wallis)

$$\frac{\pi}{2} = \prod_{n=1}^{\infty} \frac{2n}{2n-1} \cdot \frac{2n}{2n+1}$$

$$= \frac{2}{1} \cdot \frac{2}{3} \cdot \frac{4}{3} \cdot \frac{4}{5} \cdot \frac{6}{5} \cdot \frac{6}{7} \cdots$$

More Loop Examples

- Integration
  - Parabola: $f(x) = x^2$
  - Bell curve: $\rho(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}$
- Histogram of Math.random()
- Calculate φ (Ratio of successive Fibonaccis)
- Hailstone sequence (Collatz conjecture)

$$N_{i+1} = \begin{cases} \frac{N_i}{2} & \text{if } N_i \text{ is even} \\ 3N_i + 1 & \text{if } N_i \text{ is odd} \end{cases}$$

More Loop Examples

- Print Pascal's triangle in the console

  1
  1 1
  1 2 1
  1 3 3 1
  1 4 6 5 1

- Reflect the binary print-out
- Use an infinite loop to print a continuous "wave" in the console
  - Super-impose multiple waves
  - Zap impulse: speed up the wave over time