

## Exam 2

- String Manipulations
- Functions
- Multidimensional Arrays
- Transformations
- Image Processing
- Recursion
- Classes and Inheritance
- ArrayLists and HashMaps
- Problem Solving Strategies

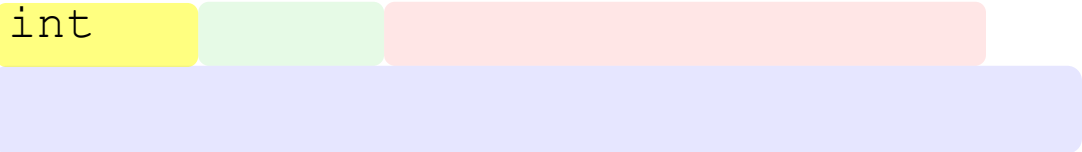
# Strings

- Declare and initialize
- Methods
  - `charAt()`
  - `equals()`
  - `substring()`
  - `length()`
- Functions
  - `split()`
  - `trim()`

# Functions

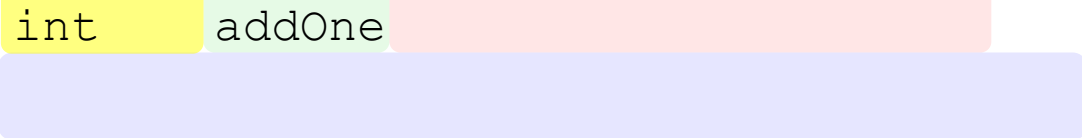
1. Return type.

`int`



2. Function name.

`int` `addOne`



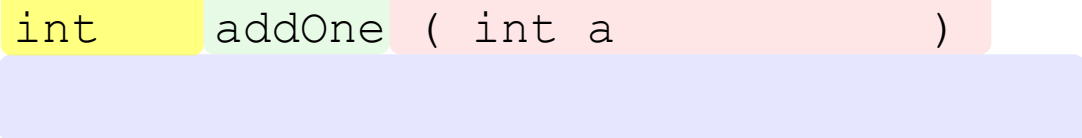
3. Parentheses.

`int` `addOne` ( )



4. Argument declarations (optional)

`int` `addOne` ( `int a` )



# Functions (Cont'd)

5. Curly brackets for the body of the function.

```
int addOne( int a )  
{  
  
}
```

6. Statements defining what the function should do.

```
int addOne( int a )  
{  
    int b;  
    b = a + 1;  
}
```

7. A return statement that returns a variable/value matching function type.

```
int addOne( int a )  
{  
    int b;  
    b = a + 1;  
    return b;  
}
```

# Declaring a Function vs. Calling a Function

```
void setup()
{
  size(500, 500);
  background(255);
}

void draw() { }

void mousePressed()
{
  float secret = secretFunction( mouseX, mouseY );
  fill(0);
  background(255);
  text(secret, mouseX, mouseY);
}
```

```
float secretFunction( float x, float y )
{
  float r, x2, y2;

  x2 = x - (0.5*width);
  y2 = y - (0.5*height);
  r = sqrt(x2*x2 + y2*y2);

  return r;
}
```

Names of passed variables do not have to match names of variables in function. Values are copied.

The single value returned by the function is preceded by the 'return' keyword.