Note: I have not taught CS so I made minor modifications to the tutorials on w3schools (these have helped me learn the material) to write this lesson outline.
https://www.w3schools.com/java/java_methods.asp

Lesson objectives:

- ☐ Create methods with parameters and arguments
- ☐ Call methods with parameters and arguments
- ☐ Practice using vocabulary function, method, parameter, argument

Aim:

- ☐ I can create and call a method with parameters and arguments

---

Warm up: Time: 5 minutes Review- fill in the blanks

(word bank: once, many, functions, method, parameter)

1. A _____ is a block of code which only runs when it is called.
2. You can pass data, known as _____, into a method.
3. Methods are used to perform certain actions, and they are also known as _____.
4. Why use methods? To reuse code: define the code _____, and use it _____ times.

Answers

1. A **method** is a block of code which only runs when it is called.
2. You can pass data, known as **parameters**, into a method.
3. Methods are used to perform certain actions, and they are also known as **functions**.
4. Why use methods? To reuse code: define the code **once**, and use it **many** times.

Lesson Content time: 10 min

(similar to https://www.w3schools.com/java/java_methods.asp)

Review - 3 min

Create a method inside Main:

```
public class Main {
  static void myMethod() {
    // code to be executed
```

```
  }
}
```

Example Explained

- myMethod() is the name of the method
- static means that the method belongs to the Main class and not an object of the Main class.
- void means that this method does not have a return value

Call a Method: To call a method in Java, write the method's name followed by two parentheses () and a semicolon; In the following example, myMethod() is used to print a text (the action), when it is called:

Example Inside main, call the myMethod() method:

A method can also be called multiple times:

```
public class Main {
  static void myMethod() {
    System.out.println("a 100 on my test!");
  }
  public static void main(String[] args) {
    myMethod();
    myMethod();
    myMethod();
  }
}

// I just got a 100 on my test!
// I just got a 100 on my test!
// I just got a 100 on my test!
```

New content: 7 min

- ➢ Parameters and Arguments: Information can be passed to methods as parameter. Parameters act as variables inside the method.
- ➢ Parameters are specified after the method name, inside the parentheses. You can add as many parameters as you want, just separate them with a comma.
- ➢ The following example has a method that takes a String called fname as parameter. When the method is called, we pass along a first name, which is used inside the method to print the full name:
- ➢ **When a parameter is passed to the method, it is called an argument. So, from the example : fname is a parameter, while Brian, Johan, Lizbeth are arguments.**

```
public class Main {
  static void myMethod(String fname) {
    System.out.println(fname + " present");
  }

  public static void main(String[] args) {
    myMethod("Brian");
    myMethod("Johan");
    myMethod("Lizbeth");
  }
}
// Brain present
// Johan present

// Lizbeth present
```

You can have as many parameters as you like:

```
public class Main {
  static void myMethod(String fname, int age) {
    System.out.println(fname + " is " + age);
  }

  public static void main(String[] args) {
    myMethod("Brian", 11);
    myMethod("Johan", 12);
    myMethod("Lizbeth", 11);
  }
}

// Brian is 11
```

```
// Johan is 12
// Lizbeth is 11
```

When you are working with multiple parameters, the method call must have the same number of arguments as there are parameters, and the arguments must be passed in the same order.

Lesson Activity time: 20 minutes

Task 1: Insert the missing part to call myMethod from main and modify message in println.

```
static void myMethod() {
  System.out.println("I just got ____!");
}

public static void main(String[] args) {
  ;
}
```

Task 2: Add a fname parameter of type String to myMethod, and output "John Doe":

```
static void myMethod( ) {
  System.out.println( + " Doe");
}

public static void main(String[] args) {
  myMethod("John");
}
```

Task 3: Change any of the programs in this lesson or another lesson to do something different. Include comments, parameters and arguments. Try to write the code without looking at notes.

Closing time: 10 min

Share your code with your group and see if they can predict the output. Quiz each other locating these parts of the code: function, method, parameter, argument.

Additional practice:
Beginner: https://www.w3schools.com/java/java_methods_param.asp
Intermediate: https://chortle.ccsu.edu/Java5/index.html#09
Advance: https://books.trinket.io/thinkjava/chapter4.html

Equity framework-Historically Responsive Literacy -need to figure out how to do this; use current events/find diverse texts written by diverse authors
1) identity/cultural competence (who we say we are, who others say we are, and the people we desire to be)
2) skill development - see objectives
3) intellectual development - see objectives
4) criticality/socio-political consciousness *(read, write, think, and speak in ways to understand power and equity and promote anti-oppression)*

UDL:
visuals-images/videos/real world examples/anchor charts/reference reinforcement opportunities/pronunciation/call and response/checklists/modified reading/chunk down verbal instruction/hetero grouping/redirection/timed activities, breaks to support mental fatigue/read alouds/limited distractions/preview/frontload vocabulary/repetition; modeling; small group instruction; frequent check ins/ graphic organizers/preferential seating; translations

Standards: need to complete

Answers to Practice

Task 1

Insert the missing part to call `myMethod` from `main`.

```
static void myMethod() {
  System.out.println("I just got executed!");
}

public static void main(String[] args) {
  ;
}
```

Task 2

Add a `fname` parameter of type `String` to `myMethod`, and output "John Doe".

```
static void myMethod( ) {
  System.out.println( + " Doe");
}

public static void main(String[] args) {
  myMethod("John");
}
```

Task 3: various responses