Syntax,
Control Logic,
Classes, and Objects

AP Computer Science
Victor Kaiser-Pendergrast

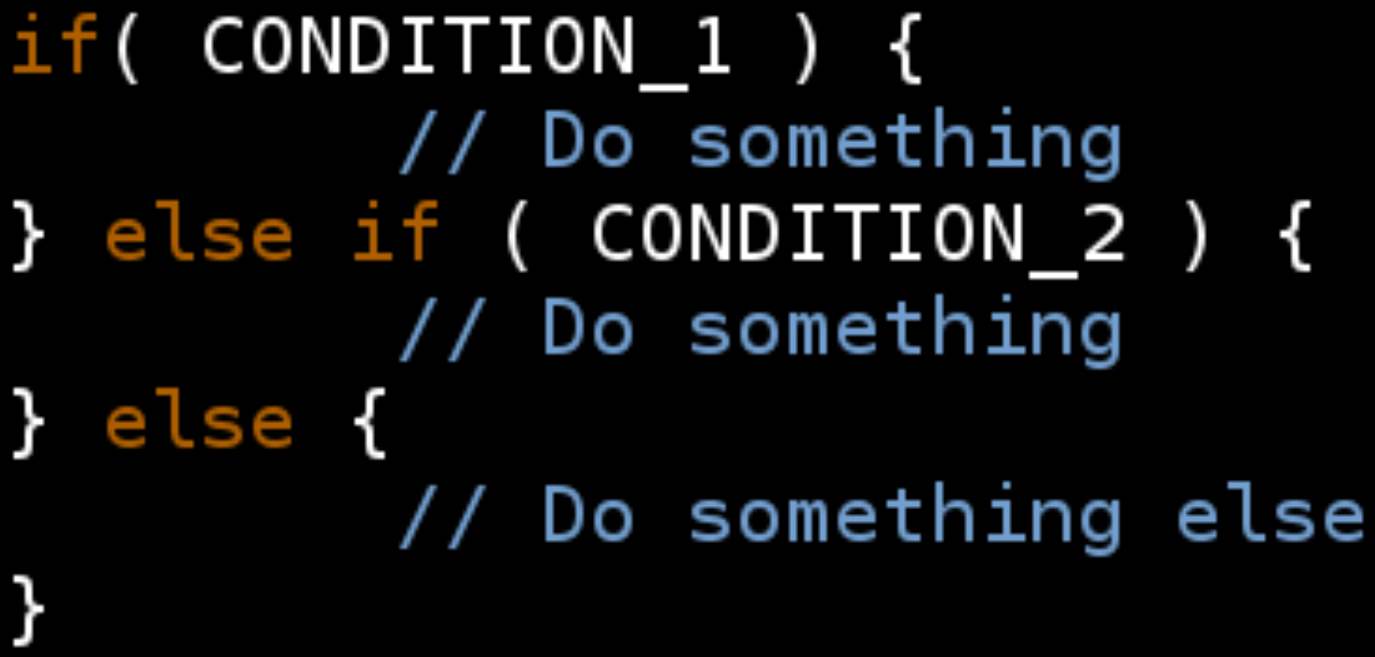
Basic Syntax

- Basically the same as C / C++

- Statements end with semicolons
  - int x = 5;

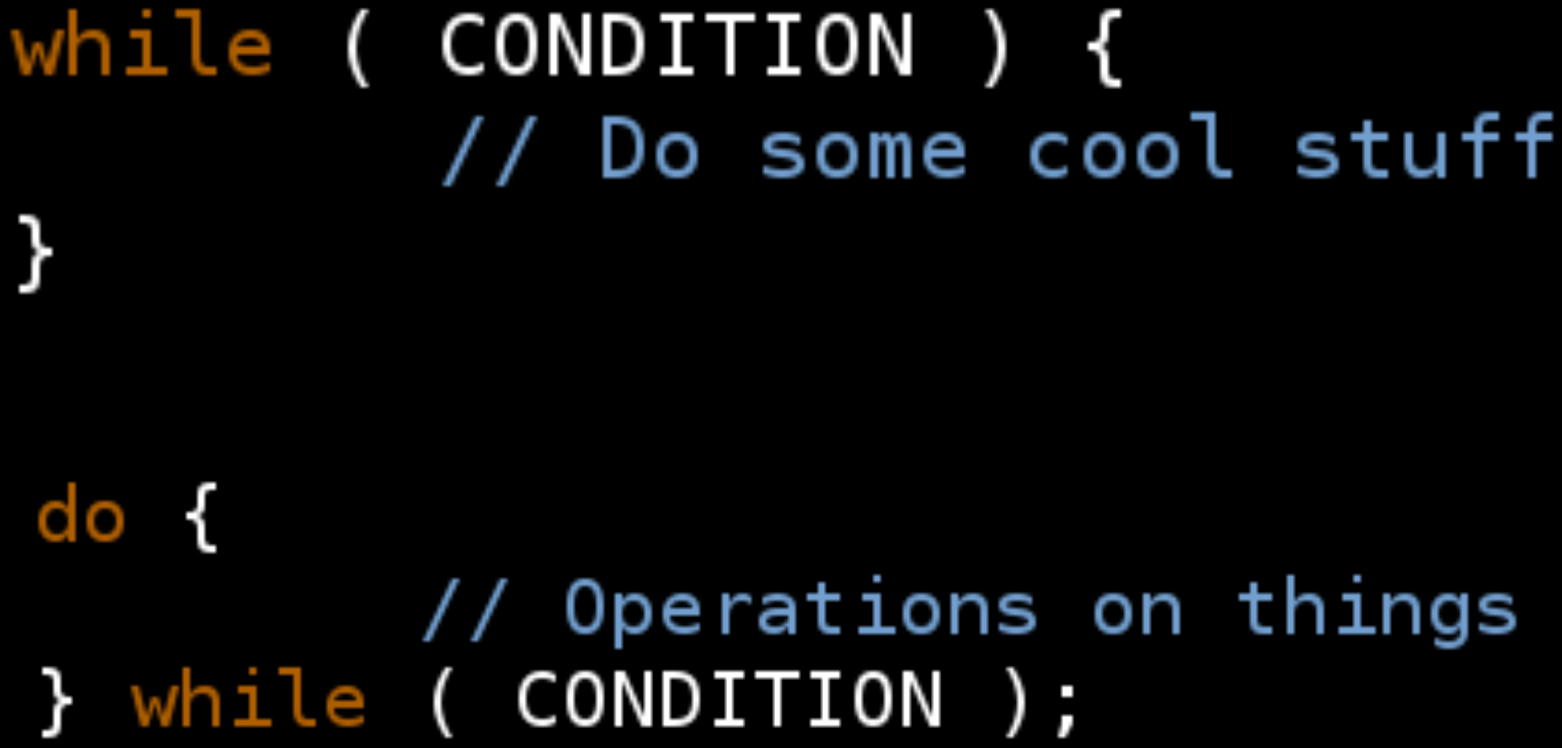
- = vs ==

If Statements



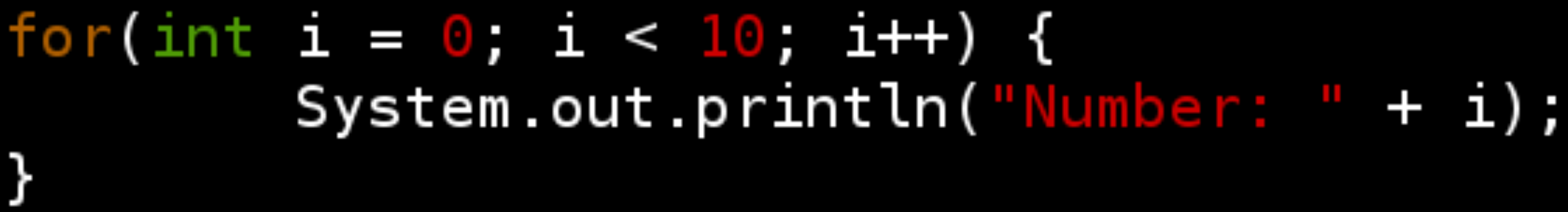
```
if( CONDITION_1 ) {
        // Do something
} else if ( CONDITION_2 ) {
        // Do something
} else {
        // Do something else
}
```

While Loops



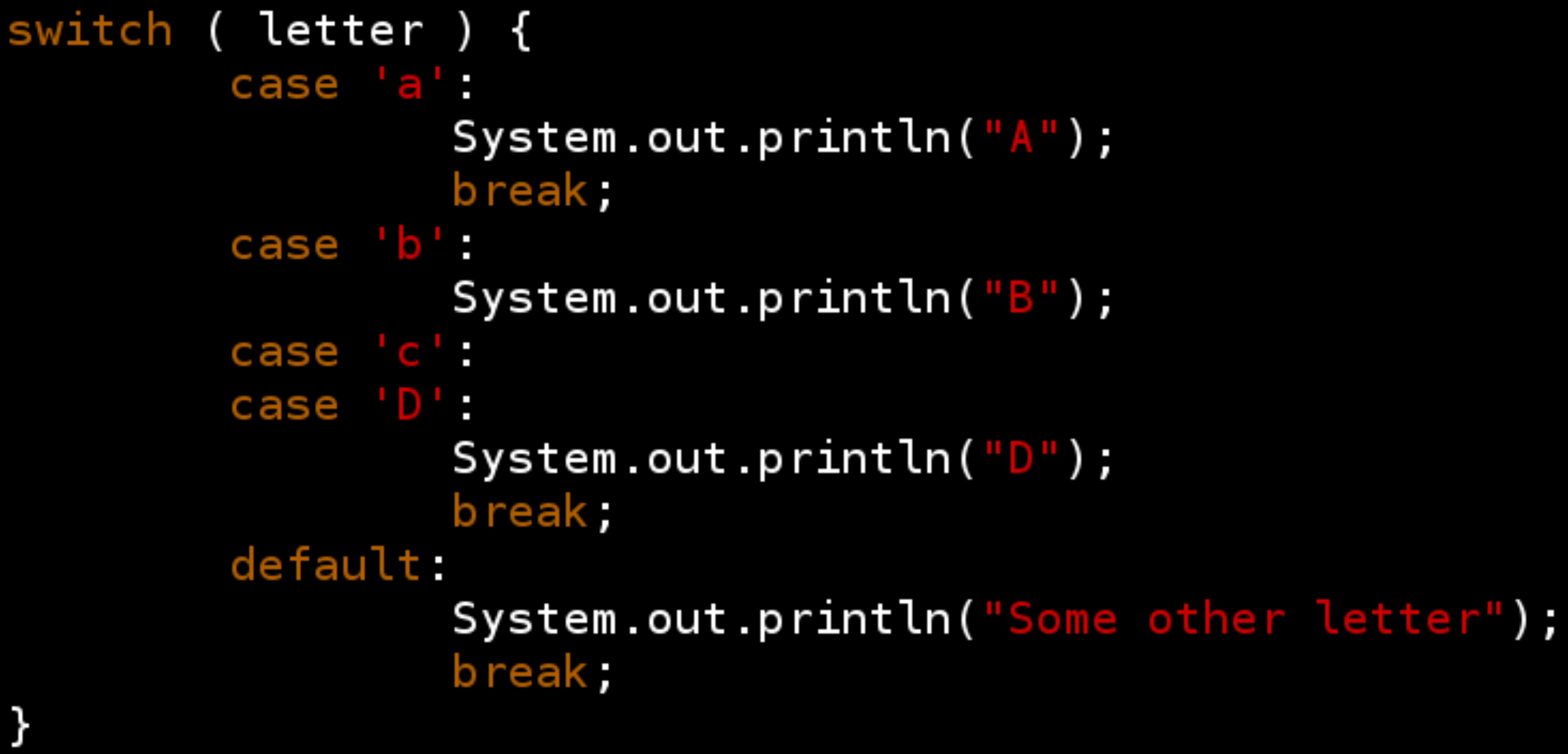
```
while ( CONDITION ) {
        // Do some cool stuff
}


 do {
        // Operations on things
 } while ( CONDITION );
```

For Loops



```
for(int i = 0; i < 10; i++) {
        System.out.println("Number: " + i);
}
```

Switch Statements



```
switch ( letter ) {
        case 'a':
                System.out.println("A");
                break;
        case 'b':
                System.out.println("B");
        case 'c':
        case 'D':
                System.out.println("D");
                break;
        default:
                System.out.println("Some other letter");
                break;
}
```

Classes and Objects

- Classes describe the same kind of object
- All methods (functions) are written inside classes
- Classes can contain variables that are class or object specific

Classes and Objects

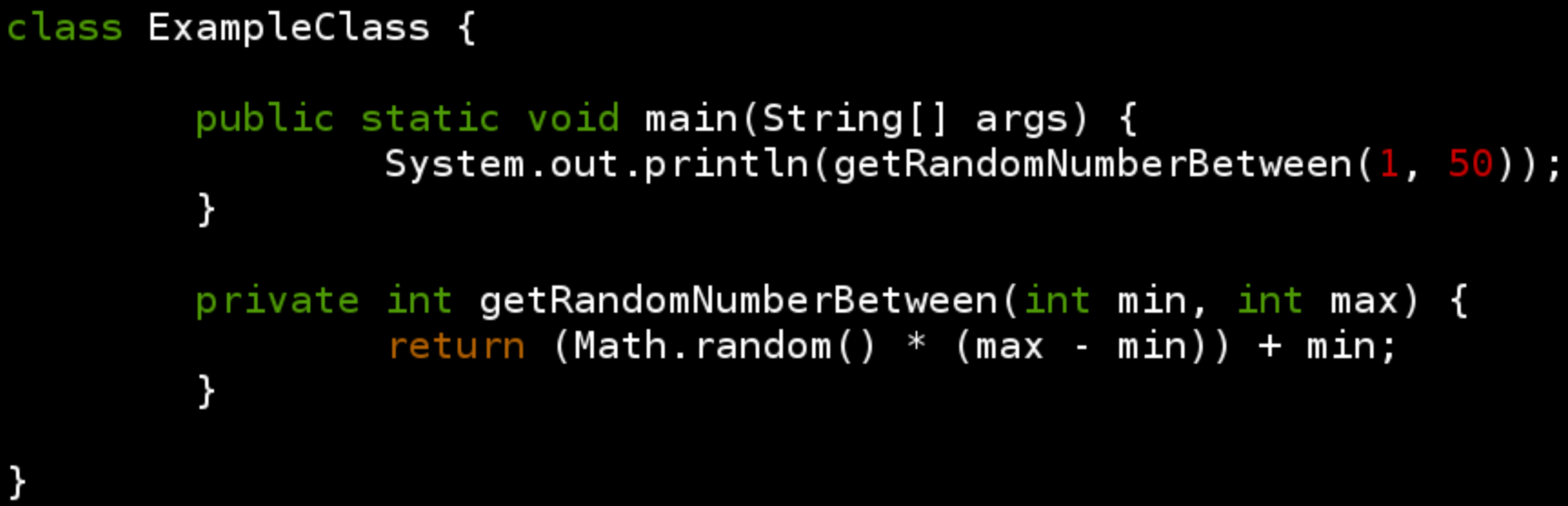
- Objects are instances of a Class
    - Truck can be a Class
    - Victor’s Truck is a specific instance of the Truck Class

- Truck victorsTruck = new Truck( );

Methods (functions)

- Used to…
  - Encapsulate code is used many times
  - Expose functionality or variables in Classes / Objects

Classes and Methods



```java
class ExampleClass {

        public static void main(String[] args) {
                System.out.println(getRandomNumberBetween(1, 50));
        }

        private int getRandomNumberBetween(int min, int max) {
                return (Math.random() * (max - min)) + min;
        }

}
```

Public vs Private

- Allow you to show or hide variables and methods in classes
  - *public* : accessible inside and outside the Class or Object
  - *private* : only visible inside

Getters / Setters

- Public methods that *get* or *set* private variables

- Note: AP likes to also refer to these as Accessors / Modifiers