Data Structures and Algorithms in Java

Procedural Programming: Your First Programs

Outline

1. Java
2. Programming in Java
3. Application Programming Interface (API)
4. Input and Output
5. Errors in a Program

Java

Java

General-purpose, high-level, object-oriented programming language

Java

General-purpose, high-level, object-oriented programming language

Key features

- Write once, run anywhere
- Relatively fast
- Robust
- Secure

Programming in Java

Step 1: Create/edit the program (eg, `Program.java`)

Programming in Java

Step 1: Create/edit the program (eg, `Program.java`)

Step 2: Compile the program

```
× ~/workspace/dsaj
$ _
```

Programming in Java

Step 1: Create/edit the program (eg, `Program.java`)

Step 2: Compile the program

× ~/workspace/dsaj

```
$ javac -d out src/Program.java
```

Programming in Java

Step 1: Create/edit the program (eg, `Program.java`)

Step 2: Compile the program

```
× ~/workspace/dsaj
1 $ javac -d out src/Program.java
2 $ _
```

Programming in Java

Step 1: Create/edit the program (eg, `Program.java`)

Step 2: Compile the program

```
× ~/workspace/dsaj
$ javac -d out src/Program.java
$ _
```

Step 3: Run the program

```
× ~/workspace/dsaj
$ _
```

Programming in Java

Step 1: Create/edit the program (eg, `Program.java`)

Step 2: Compile the program

```
× ~/workspace/dsaj
$ javac -d out src/Program.java
$ _
```

Step 3: Run the program

```
× ~/workspace/dsaj
$ java Program
```

Programming in Java

Step 1: Create/edit the program (eg, `Program.java`)

Step 2: Compile the program

```
× ~/workspace/dsaj
$ javac -d out src/Program.java
$ _
```

Step 3: Run the program

```
× ~/workspace/dsaj
$ java Program
<program output>
$ _
```

Programming in Java

Step 1: Create/edit the program (eg, `Program.java`)

Step 2: Compile the program

```
× ~/workspace/dsaj
$ javac -d out src/Program.java
$ _
```

Step 3: Run the program

```
× ~/workspace/dsaj
$ java Program
<program output>
$ _
```

Repeat steps 1 – 3 until program output matches expected

Programming in Java

× Program.java 1/2

```
// Package statement.
[package dsa;]

// Import statements.
...

// Outer class definition.
public class Program [implements <name>] {
    // Class/instance variable declarations.
    ...

    // Constructor definitions.
    ...

    // Method definitions.
    ...

    // Inner class definitions.
    ...

```

Programming in Java

× Program.java 2/2

```
21      // Function definitions.
22      ...
23  }
```

Programming in Java

`HelloWorld.java`

- Standard output: the message “Hello, World”

`HelloWorld.java`

- Standard output: the message “Hello, World”

```
× ~/workspace/dsaj
$ _
```

`HelloWorld.java`

- Standard output: the message “Hello, World”

```
× ~/workspace/dsaj
$ javac -d out src/HelloWorld.java
```

`HelloWorld.java`

- Standard output: the message “Hello, World”

× ~/workspace/dsaj

```
$ javac -d out src/HelloWorld.java
$ _
```

`HelloWorld.java`

- Standard output: the message “Hello, World”

× ~/workspace/dsaj

```
$ javac -d out src/HelloWorld.java
$ java HelloWorld
```

`HelloWorld.java`

- Standard output: the message “Hello, World”

× ~/workspace/dsaj

```
$ javac -d out src/HelloWorld.java
$ java HelloWorld
Hello, World
$ _
```

Programming in Java

Programming in Java

× HelloWorld.java

```java
// Writes the message "Hello, World" as standard output.

import stdlib.StdOut;

public class HelloWorld {
    // Entry point.
    public static void main(String[] args) {
        StdOut.println("Hello, World");
    }
}
```

Application Programming Interface (API)

Application Programming Interface (API)

API is a set of rules or protocols that allow different software applications to communicate with each other

Application Programming Interface (API)

API is a set of rules or protocols that allow different software applications to communicate with each other

The API for a library describes the behavior of its functions

Application Programming Interface (API)

API is a set of rules or protocols that allow different software applications to communicate with each other

The API for a library describes the behavior of its functions

Example (`stdlib.StdOut`)

`static void println(Object x)`	prints an object and a newline to standard output
`static void print(Object x)`	prints an object to standard output

Input and Output

Input and Output

Input and Output

Input types

- Command-line input
- Standard input
- File input

Input and Output

Input types

- Command-line input
- Standard input
- File input

Output types

- Standard output
- File output

Input and Output

Input and Output

Command-line inputs (aka arguments) are strings listed next to the program name during execution

× ~/workspace/dsaj

```
$ java Program input1 input2 input3 ...
```

Input and Output

Command-line inputs (aka arguments) are strings listed next to the program name during execution

× ~/workspace/dsaj

```
$ java Program input1 input2 input3 ...
```

The inputs are accessed within the entry-point function as `args[0]`, `args[1]`, `args[2]`, ...

Input and Output

Command-line inputs (aka arguments) are strings listed next to the program name during execution

× ~/workspace/dsaj

```
$ java Program input1 input2 input3 ...
```

The inputs are accessed within the entry-point function as `args[0]`, `args[1]`, `args[2]`, ...

Example

× ~/workspace/dsaj

```
$ java Program Galileo "Isaac Newton" Einstein
```

args[0]	args[1]	args[2]
"Galileo"	"Isaac Newton"	"Einstein"

Input and Output

Input and Output

`UseArgument.java`

- Command-line input: a name
- Standard output: a message containing the name

Input and Output

`UseArgument.java`

- Command-line input: a name
- Standard output: a message containing the name

```
× ~/workspace/dsaj
$ _
```

Input and Output

`UseArgument.java`

- Command-line input: a name
- Standard output: a message containing the name

× ~/workspace/dsaj

```
$ javac -d out src/UseArgument.java
```

Input and Output

`UseArgument.java`

- Command-line input: a name
- Standard output: a message containing the name

~/workspace/dsaj

```
$ javac -d out src/UseArgument.java
$ _
```

Input and Output

`UseArgument.java`

- Command-line input: a name
- Standard output: a message containing the name

× ~/workspace/dsaj

```
$ javac -d out src/UseArgument.java
$ java UseArgument Alice
```

Input and Output

`UseArgument.java`

- Command-line input: a name
- Standard output: a message containing the name

~/workspace/dsaj

```
$ javac -d out src/UseArgument.java
$ java UseArgument Alice
Hi, Alice. How are you?
$ _
```

Input and Output

UseArgument.java

- Command-line input: a name
- Standard output: a message containing the name

× ~/workspace/dsaj

```
$ javac -d out src/UseArgument.java
$ java UseArgument Alice
Hi, Alice. How are you?
$ java UseArgument Bob
```

Input and Output

`UseArgument.java`

- Command-line input: a name
- Standard output: a message containing the name

× ~/workspace/dsaj

```
$ javac -d out src/UseArgument.java
$ java UseArgument Alice
Hi, Alice. How are you?
$ java UseArgument Bob
Hi, Bob. How are you?
$ _
```

Input and Output

UseArgument.java

- Command-line input: a name
- Standard output: a message containing the name

× ~/workspace/dsaj

```
$ javac -d out src/UseArgument.java
$ java UseArgument Alice
Hi, Alice. How are you?
$ java UseArgument Bob
Hi, Bob. How are you?
$ java UseArgument Carol
```

Input and Output

`UseArgument.java`

- Command-line input: a name
- Standard output: a message containing the name

× ~/workspace/dsaj

```
$ javac -d out src/UseArgument.java
$ java UseArgument Alice
Hi, Alice. How are you?
$ java UseArgument Bob
Hi, Bob. How are you?
$ java UseArgument Carol
Hi, Carol. How are you?
$ _
```

Input and Output

Input and Output

× UseArgument.java

```java
// Receives a name as command-line input; and writes a message containing that
// name as standard output.

import stdlib.StdOut;

public class UseArgument {
    // Entry point.
    public static void main(String[] args) {
        StdOut.print("Hi, ");
        StdOut.print(args[0]);
        StdOut.println(". How are you?");
    }
}
```

Errors in a Program

Errors in a Program

Compile-time errors are identified and reported by `javac` when it compiles a program

Errors in a Program

Compile-time errors are identified and reported by javac when it compiles a program

Example

× UseArgument.java

```java
// Receives a name as command-line input; and writes a message containing that
// name as standard output.

import stdlib.StdOut;

public class UseArgument {
    // Entry point.
    public static void main(String[] args) {
        StdOut.print("Hi, ");
        StdOut.print(args[0]);
        StdOut.println(". How are you?");
    }
}
```

Errors in a Program

Errors in a Program

```
× ~/workspace/dsaj
$ _
```

Errors in a Program

```
× ~/workspace/dsaj
$ javac -d out src/UseArgument.java
```

Errors in a Program

```
× ~/workspace/dsaj
$ javac -d out src/UseArgument.java
UseArgument.java:10: error: ']' expected
        StdOut.print(args[0);
                            ^
1 error
$ _
```

Errors in a Program

Errors in a Program

Run-time errors are identified and reported by `java` when it runs a program

Errors in a Program

Run-time errors are identified and reported by `java` when it runs a program

Example

× UseArgument.java

```
// Receives a name as command-line input; and writes a message containing that
// name as standard output.

import stdlib.StdOut;

public class UseArgument {
    // Entry point.
    public static void main(String[] args) {
        StdOut.print("Hi, ");
        StdOut.print(args[0]);
        StdOut.println(". How are you?");
    }
}
```

Errors in a Program

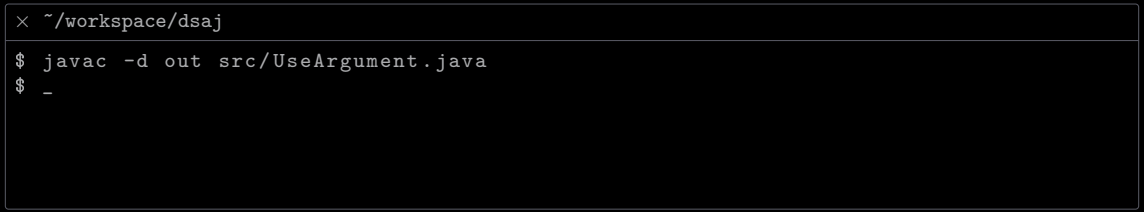
Errors in a Program

```
× ~/workspace/dsaj
$ _
```

Errors in a Program

```
× ~/workspace/dsaj
$ javac -d out src/UseArgument.java
```

Errors in a Program



Errors in a Program

× ~/workspace/dsaj

```
$ javac -d out src/UseArgument.java
$ java UseArgument
```

Errors in a Program

```
× ~/workspace/dsaj
$ javac -d out src/UseArgument.java
$ java UseArgument
Hi, Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 0
out of bounds for length 0
      at UseArgument.main(UseArgument.java:10)
$ _
```

Errors in a Program

Errors in a Program

Logic errors are neither identified nor reported by `java`, but produce unintended output

Errors in a Program

Logic errors are neither identified nor reported by java, but produce unintended output

Example

× UseArgument.java

```java
// Receives a name as command-line input; and writes a message containing that
// name as standard output.

import stdlib.StdOut;

public class UseArgument {
    // Entry point.
    public static void main(String[] args) {
        StdOut.print("Hi, ");
        StdOut.print(args[0]);
        StdOut.print(". How are you?");
    }
}
```

Errors in a Program

Errors in a Program

```
× ~/workspace/dsaj
$ _
```

Errors in a Program

× ~/workspace/dsaj

```
$ javac -d out src/UseArgument.java
```

Errors in a Program

× ~/workspace/dsaj

```
$ javac -d out src/UseArgument.java
$ _
```

Errors in a Program

× ~/workspace/dsaj

```
$ javac -d out src/UseArgument.java
$ java UseArgument Alice

```

Errors in a Program