CS 112 – Introduction to Computing II

Wayne Snyder
Computer Science Department
Boston University

Today:

Java Statements:

- conditionals: if and if else
- loops: while and for

Java Data Structures:

- arrays

Next Time: Program structure, methods, and scope of identifiers

Reading assignments are posted on the web site!

Computer Science

Java Statements

Computer Science

A Java program is a sequence of statements followed by semicolons; each statement is executed in sequence, and each has some effect on the state of the computer:

```
1  /* File: SampleProgram.java
2   * Author: Wayne Snyder (snyder@bu.edu)
3   * Date: 2/25/16
4   * Purpose: This is a sample problem for lecture 3 in CS 112.
5   */
6
7  public class SampleProgram {
8
9      public static void main(String[] args) {
10         int x;                  // declare x to be int
11         x = 3;                  // assign value to x
12         System.out.println(x);
13         double y = 3.4;         // combine declaration and assignment
14         double z = x + y;
15         System.out.println(z);
16     }
17 }
```

Flow of control: conditionals if, if/then, if/then/else

Computer Science

Conditional statements alter the flow of execution by testing some boolean condition and branching one way or the other; you just have to translate what you already know from Python into Java syntax:

Simple if statement:

Python:

```
x = 4

print("testing x....")

if( (x % 2) == 0):
    print("x is even")

print("done")
```

Java:

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

    int x;
    System.out.println("testing x....");

    if( (x % 2) == 0 )
        System.out.println("x is even");

    System.out.println("done");

}
```

Flow of control: conditionals if, if/then, if/then/else

Computer Science

Conditional statements alter the flow of execution by testing some boolean condition and branching one way or the other; you just have to translate what you already know from Python into Java syntax:

Compound if/then statement:

Python:

```
print("testing x....")

if( (x % 2) == 0):
    print("x is even")
else:
    print("x is odd")

print("done")
```

Java:

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

    int x = 4;
    System.out.println("testing x....");

    if( (x % 2) == 0 )
        System.out.println("x is even");
    else
        System.out.println("x is odd");

    System.out.println("done");
}
```

Flow of control: conditionals if, if/then, if/then/else

Conditional statements alter the flow of execution by testing some boolean condition and branching one way or the other; you just have to translate what you already know from Python into Java syntax:

Compound if/then statement:

Python:

```
print("testing x....")
if(x < 0):
    print("x is negative")
elif( x < 10):
    print("x is positive but less than 10")
elif(x < 100):
    print("x is positive but less than 100")
else:
    print("x is greater or equal to 100")

print("done")
```

Java:

```
int x = 4;
System.out.println("testing x....");

if( x < 0 )
    System.out.println("x is negative");
else if( x < 10 )
    System.out.println("x is positive but less than
else if( x < 100 )
    System.out.println("x is positive but less than
else
    System.out.println("x is greater or equal to 10
System.out.println("done");
```

5

Flow of control: conditionals if, if/then, if/then/else

Compound statements

Any branch of a conditional can have multiple statements (called "compound statements" in Java); instead of indentation, we use curly braces to indicate that all these statements should be executed in sequence:

Python:

```
x = 4

print("testing x....")

if( (x % 2) == 0):
    print("x is even")
    print("proceeding to divide x by 2")
    x = x / 2;

print("done")
```

Java:

```
int x = 4;
System.out.println("testing x....");

if( (x % 2) == 0 ) {
    System.out.println("x is even");
    System.out.println("proceeding to divide
    x = x / 2;
}

System.out.println("done");
```

6

Flow of control: compound statements

Compound statements

Any branch of a conditional can have multiple statements (called "compound statements" in Java); instead of indentation, we use curly braces to indicate that all these statements should be executed in sequence:

Python:

```
print("testing x....")

if( (x % 2) == 0):
    print("x is even")
    print("proceeding to divide x by 2")
    x = x / 2
else:
    print("x is odd")
    print("proceeding to add 1 to x")
    x += 1

print("done")
```

Java:

```
System.out.println("testing x....");

if( (x % 2) == 0 ) {
    System.out.println("x is even");
    System.out.println("proceeding to divide x by 2");
    x = x / 2;
}
else {
    System.out.println("x is odd");
    System.out.println("proceeding to add 1 to x");
    x += 1;
}

System.out.println("done");
```

Flow of control: loops: while and for

Loop: while statement

Python:

```
x = 6
while(x < 10):
    print(x)
    x += 1
```

Java:

```
x = 6;
while( x < 10 ) {
    System.out.println( x );
    x += 1;
}
```

Loop: for statement

Python:

```
for y in range(6,10):
    print(y)
```

Java:

```
for(int y = 6; y < 10; ++y)
    System.out.println(y);
```

8

Flow of control: loops: break, continue

break and continue work exactly the same as in Python

Python:

```
x = 0
while(x < 10):
    print(x)
    if(x == 5):
        break
    x += 1
```

```
x = 0
while(x < 10):
    print(x)
    if(x == 5):
        x = 7
        continue
    x += 1
```

Java:

```
x = 0;
while( x < 10 ) {
    System.out.println( x );
    if( x == 5 )
        break;
    x += 1;
}
```

```
x = 0;
while( x < 10 ) {
    System.out.println( x );
    if( x == 5 ) {
        x = 7;
        continue;
    }
    x += 1;
}
```

9

Java Data Types: Array

The fundamental data type in Python is a list, which stores a list of values:

Python:

```
In [1]: A = [2, 3, 4, 6, 7]

In [2]: S = ["hi", "there", "folks!"]

In [3]: X = [3.14, 3.1415, 3.141592]

In [4]: A[0]
Out[4]: 2

In [5]: S[3]
Traceback (most recent call last):

  File "<ipython-input-5-2cf1e01d48e3>", line 1, in <module>
    S[3]

IndexError: list index out of range


In [6]: for i in range(len(X)):
   ...:     print(X[i])
   ...:
3.14
3.1415
3.141592

In [7]:
```

Java:

10

Java Data Types: Array

Computer Science

In Java, the most common way to store a sequence of values is in an array:

Python:

```
In [1]: A = [2, 3, 4, 6, 7]

In [2]: S = ["hi", "there", "folks!"]

In [3]: X = [3.14, 3.1415, 3.141592]

In [4]: A[0]
Out[4]: 2

In [5]: S[3]
Traceback (most recent call last):

  File "<ipython-input-5-2cf1e01d48e3>",
    S[3]

IndexError: list index out of range


In [6]: for i in range(len(X)):
   ...:     print(X[i])
   ...:
3.14
3.1415
3.141592

In [7]:
```

Java:

```
int[] A = { 2, 3, 4, 6, 7 };

String[] S = { "hi", "there", "folks" };

double[] X = { 3.14, 3.1415, 3.141592 };

System.out.println( A[0] );

System.out.println( S[3] );

for( int i = 0; i < X.length; ++i )
    System.out.println( X[i] );
```

11

Java Data Types: Array

Computer Science

In Java, the most common way to store a sequence of values is in an array:

Python:

```
In [1]: A = [2, 3, 4, 6, 7]

In [2]: S = ["hi", "there", "folks!"]

In [3]: X = [3.14, 3.1415, 3.141592]

In [4]: A[0]
Out[4]: 2

In [5]: S[3]
Traceback (most recent call last):

  File "<ipython-input-5-2cf1e01d48e3>",
    S[3]

IndexError: list index out of range


In [6]: for i in range(len(X)):
   ...:     print(X[i])
   ...:
3.14
3.1415
3.141592

In [7]:
```

Java:

```
18
19        System.out.println( S[3] );        // run tim
20
```

Interactions | Con

```
Welcome to DrJava.  Working directory is /Users/waynesnyder/D
> run SampleProgram
2
java.lang.ArrayIndexOutOfBoundsException: 3
        at SampleProgram.main(SampleProgram.java:19)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeM
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(Del
        at java.lang.reflect.Method.invoke(Method.java:597)
        at edu.rice.cs.drjava.model.compiler.JavacCompiler.run
>
```

12

Java Data Types: Array

Computer Science

In Java, the most common way to store a sequence of values is in an array:

Python:

```
In [1]: A = [2, 3, 4, 6, 7]

In [2]: S = ["hi", "there", "folks!"]

In [3]: X = [3.14, 3.1415, 3.141592]

In [4]: A[0]
Out[4]: 2

In [5]: S[3]
Traceback (most recent call last):

  File "<ipython-input-5-2cf1e01d48e3>
    S[3]

IndexError: list index out of range


In [6]: for i in range(len(X)):
   ...:     print(X[i])
   ...:
3.14
3.1415
3.141592

In [7]:
```

Java:

```
11        int[] A = { 2, 3, 4, 6, 7 };
12
13        String[] S = { "hi", "there", "folks" };
14
15        double[] X = { 3.14, 3.1415, 3.141592 };
16
17        System.out.println( A[0] );
18
19    //    System.out.println( S[3] );        // run time error
20
21        for( int i = 0; i < X.length; ++i )
22            System.out.println( X[i] );
23
24    }
25 }
```

```
Welcome to DrJava.  Working directory is /Users/waynesnyder/Dropbox/Doc
> run SampleProgram
2
3.14
3.1415
3.141592
>
```

Java Data Types: Array

Computer Science

As in Python, in Java we can change the value in a particular location in the sequence:

Python:

```
In [7]: A = [ 2, 3, 4, 6, 7]

In [8]: A
Out[8]: [2, 3, 4, 6, 7]

In [9]: A[2] = 12

In [10]: A
Out[10]: [2, 3, 12, 6, 7]
```

Java:

```
int[] A = { 2, 3, 4, 6, 7 };

System.out.print("\n[ ");
for( int i = 0; i < A.length-1; ++i )
    System.out.print( A[i] + ", ");
System.out.println( A[ A.length - 1 ] + "]" );

A[2] = 12;

System.out.print("\n[ ");
for( int i = 0; i < A.length-1; ++i )
    System.out.print( A[i] + ", ");
System.out.println( A[ A.length - 1 ] + "]" );
```

```
> run SampleProgram

[ 2, 3, 4, 6, 7]

[ 2, 3, 12, 6, 7]
>
```

14

Java Data Types: Array

Computer Science

But we can NOT change the size of the sequence at run time, as we can in Python:

Python:

```
In [10]: A
Out[10]: [2, 3, 12, 6, 7]

In [11]: A.append(23)

In [12]: A
Out[12]: [2, 3, 12, 6, 7, 23]
```

Java:

NOT POSSIBLE!

15

Java Data Types: Array

Computer Science

The reason has to do with strong typing: in Java, we have to allocate memory for the array when we create it. We can't change the size once this is done. To add to the array, we would have to redo the whole process. Let's look at it from the beginning…..

```
int[] A = new int[ 5 ];    // create a new array of size 5

A[0] = 2;                  // put values in at run time
A[1] = 3;
A[2] = A[0] + 2;
A[3] = A[0] * A[1];
A[4] = A[1] + A[2];

A[2] = 12;

System.out.print("\n[ ");
for( int i = 0; i < A.length-1; ++i )
    System.out.print( A[i] + ", ");
System.out.println( A[ A.length - 1 ] + " ]" );
```

```
> run SampleProgram

[ 2, 3, 12, 6, 15 ]
>
```

Java Data Types: Array

Computer Science

Warning: You can NOT use an array literal to assign to an array, EXCEPT when you declare it; this is why an array literal is called an "array initializer":

```
17
18          int[] A = { 2, 3, 12, 6, 15 };   // OK!
19
20          System.out.print("\n[ ");
21          for( int i = 0; i < A.length-1; ++i )
22              System.out.print( A[i] + ", ");
23          System.out.println( A[ A.length - 1 ] + " ]" );
24
25          A = { 2, 3, 12, 6, 15, 23 };     // NOT OK!
26
27
28
29
30      }
31 }
```

Interactions | Console | Compiler Output

```
4 errors found:
File:
/Users/waynesnyder/Dropbox/Documents/Teaching/cs112/Web/CodeExamples/SampleProgram.java
[line: 25]
Error:
/Users/waynesnyder/Dropbox/Documents/Teaching/cs112/Web/CodeExamples/SampleProgram.java:25:
 illegal start of expression
File:
```

7