CS 112 – Introduction to Computing II

Wayne Snyder
Computer Science Department
Boston University

Today:

- Arrays (1D and 2D)
- Methods
- Program structure
- Fields vs local variables

Next time: Program structure continued: Classes and objects; public vs private; static vs non-static objects;

Reading assignments will be posted on the web site!

Computer Science

Java Data Types: Array

Computer Science

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]:
```

2

Java Data Types: Array

Computer Science

The fundamental data type in Python is a list, which stores a list of values; in Java, a sequence of values is stored 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] );
```

Same: Access elements using […]

3

Java Data Types: Array

Computer Science

The fundamental data type in Python is a list, which stores a list of values; in Java, a sequence of values is stored 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] );
```

Same: Access elements using […]

Indices range from 0 … length – 1

For loop used to run through entire array

4

Java Data Types: Array

Computer Science

The fundamental data type in Python is a list, which stores a list of values; in Java, a sequence of values is stored 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] );
```

```
java.lang.ArrayIndexOutOfBoundsException: 3
        at SampleProgram.main(SampleProgram.java:19)
```

Same: Access elements using […]
Indices range from 0 … length – 1
For loop used to run through entire array
Error if index is not in range.

5

Java Data Types: Array

Computer Science

The fundamental data type in Python is a list, which stores a list of values; in Java, a sequence of values is stored 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] );
```

Not same: Initialize array using { … } instead of […]

6

Java Data Types: Array

Computer Science

The fundamental data type in Python is a list, which stores a list of values; in Java, a sequence of values is stored 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]:  B = [ 4, "hi there", 2.3, True ]
```

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] );
```

Not same: Initialize array using { … } instead of […]
Must declare with array type with [ ] and all elements must be of same type.

7

Java Data Types: Array

Computer Science

The fundamental data type in Python is a list, which stores a list of values; in Java, a sequence of values is stored 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]:  B = [ 4, "hi there", 2.3, True ]
```

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] );
```

Not same: Initialize array using { … } instead of […]
Must declare with array type with [ ] and all elements must be of same type.
Length of array is *name-of-array*.length

Java Data Types: Array

Computer Science

Same: 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]
>
```

9

Java Data Types: Array

Computer Science

Not same: 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:

SIMPLY NOT POSSIBLE!

10

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.

The computer's main memory is essentially a very big array holding 32- or 64-bit numbers which are interpreted as the various Java data types; each slot in the array has an index, called its "address."

int[] A = { 1, 1, 2, 3, 5, 8 };

The array indices are translated into memory addresses: 0 -> 103
1 -> 104, etc.

100:	3.14	other data
101:	-436	
102:	23	
103:	1	The array A
104:	1	
105:	2	
106:	3	
107:	5	
108:	8	
110:	'a'	other data
111:	true	

11

Java Data Types: Array

Computer Science

Programming with Arrays in Java

We can create an array using an array literal when the array is declared:

```
// Some Fibonnaci numbers

int[] A = { 1, 1, 2, 3, 5, 8 };

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

```
Welcome to DrJava.  Wor
> run SampleProgram
1 1 2 3 5 8
>
```

The array A:

0:	1
1:	1
2:	2
3:	3
4:	5
5:	8

12

Java Data Types: Array

Computer Science

Programming with Arrays in Java

We can create an array using an array literal when the array is declared:

```
// Some Fibonnaci numbers

int[] A = { 1, 1, 2, 3, 5, 8 };

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

Or we can declare a "default array" which is initialized with the default types (int: 0, double: 0.0, boolean: false):

```
int[] A = new int[6];
```

The array A:

0:	0
1:	0
2:	0
3:	0
4:	0
5:	0

13

Java Data Types: Array

Computer Science

Programming with Arrays in Java

We can create an array using an array literal when the array is declared:

```
// Some Fibonnaci numbers

int[] A = { 1, 1, 2, 3, 5, 8 };

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

We can then put values into this "default array" using assignments:

```
int[] A = new int[6];

A[0] = A[1] = 1;
A[2] = 2;
A[3] = 3;
A[4] = 5;
A[5] = 8;
```

The array A:

0:	1
1:	1
2:	2
3:	3
4:	5
5:	8

14

Java Data Types: Array

Computer Science

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

```
// Some Fibonnaci numbers

int[] A = new int[6];

A = { 1, 1, 2, 3, 5, 8 };
```

```
File: /Users/waynesnyder/Dropbox/Documents/Teaching/CS 112/Lectures & Course
Materials/SampleProgram.java  [line: 10]
Error: /Users/waynesnyder/Dropbox/Documents/Teaching/CS 112/Lectures & Course
Materials/SampleProgram.java:10: illegal start of expression
```

15

Java Data Types: Array

Computer Science

Two dimensional arrays are implemented as "array of arrays" and can be visualized of as a matrix of values; they are declared using an array initializer:

```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
```

```
// Part of Pascal's Triangle

int[][] A = { { 1, 1, 1 }, { 1, 2, 3 }, { 1, 3, 6 }, { 1, 4, 10 } };

// print out the matrix by rows and columns

for(int row = 0; row < A.length; ++row) {
    for(int col = 0; col < A[0].length; ++col) {
        System.out.print(A[row][col] + " ");
    }
    System.out.println();
}
```

```
> run SampleProgram
1 1 1
1 2 3
1 3 6
1 4 10
```

col:

row:

	0	1	2
0:	1	1	1
1:	1	2	3
2:	1	3	6
3:	1	4	10

16

Java Data Types: Array

Computer Science

Or they can be created using explicit assignments:

```
int[][] A = new int[4][3];
```

number of rows

number of columns

col:

row:

	0	1	2
0:	0	0	0
1:	0	0	0
2:	0	0	0
3:	0	0	0

When the rows are given first, this is called **row-major order**.

17

Java Data Types: Array

Computer Science

Or they can be created using explicit assignments:

col:

row:

	0	1	2
0:	1	1	1
1:	1	2	3
2:	1	3	6
3:	1	4	10

```
// Part of Pascal's Triangle

int[][] A = new int[4][3];

for(int col = 0; col < A[0].length; ++col)
    A[0][col] = 1;

for(int row = 1; row < A.length; ++row) {
    A[row][0] = 1;
    for(int col = 1; col < A[0].length; ++col)
        A[row][col] = A[row-1][col] + A[row][col-1];
}
```

18

Java Data Types: Methods

Computer Science

One way to write a program is to put ALL code into the main method:

```
public class PascalsTriangle {

    public static void main(String[] args) {

        // initialize empty array
        int[][] A = new int[4][3];

        // fill in first row with all 1's
        for(int col = 0; col < A[0].length; ++col)
            A[0][col] = 1;

        // fill in rest of rows; first column 0, rest add from left and above
        for(int row = 1; row < A.length; ++row) {
            A[row][0] = 1;
            for(int col = 1; col < A[0].length; ++col)
                A[row][col] = A[row-1][col] + A[row][col-1];
        }

        // print out the matrix by rows and columns
        for(int row = 0; row < A.length; ++row) {
            for(int col = 0; col < A[0].length; ++col) {
                System.out.print(A[row][col] + " ");
            }
            System.out.println();
        }
    }
}
```

19

Java Data Types: Methods

Computer Science

One way to write a program is to put ALL code into the main method:

```
public class PascalsTriangle {

    public static void main(String[] args) {

        // initialize empty array
        int[][] A = new int[4][3];

        // fill in first row with all 1's
        for(int col = 0; col < A[0].length; ++col)
            A[0][col] = 1;

        // fill in rest of rows; first column 0, rest add from left and above
        for(int row = 1; row < A.length; ++row) {
            A[row][0] = 1;
            for(int col = 1; col < A[0].length; ++col)
                A[row][col] = A[row-1][col] + A[row][col-1];
        }

        // print out the matrix by rows and columns
        for(int row = 0; row < A.length; ++row) {
            for(int col = 0; col < A[0].length; ++col) {
                System.out.print(A[row][col] + " ");
            }
            System.out.println();
        }
    }
}
```

20

Java Data Types: Methods

Computer Science

A better way is to use "Procedural Decomposition" and break the code into methods which are called by main (and possibly each other):

```
public class PascalsTriangle {

    static int[][] getTriangle(int n) {

        // initialize empty array
        int[][] A = new int[n][n];

        // fill in first row with all 1's
        for(int col = 0; col < A[0].length; ++col)
            A[0][col] = 1;

        // fill in rest of rows; first column 0, rest add from left and above
        for(int row = 1; row < A.length; ++row) {
            A[row][0] = 1;
            for(int col = 1; col < A[row].length; ++col)
                A[row][col] = A[row-1][col] + A[row][col-1];
        }
        return A;
    }
    static void printTriangle(int[][] A) {

        // print out the matrix by rows and columns
        for(int row = 0; row < A.length; ++row) {
            for(int col = 0; col < A[0].length; ++col)
                System.out.print(A[row][col] + " ");
            }
            System.out.println();
        }
    }

    public static void main(String[] args) {

        for( int n = 3; n < 10; ++n) {
            printTriangle( getTriangle(n) );
            System.out.println();
        }
    }
```

Advantages:

(1) Avoid duplicating code by cutting and pasting – and avoid the "multiple update problem" (when you fix one instance, you have to find all the other instances.

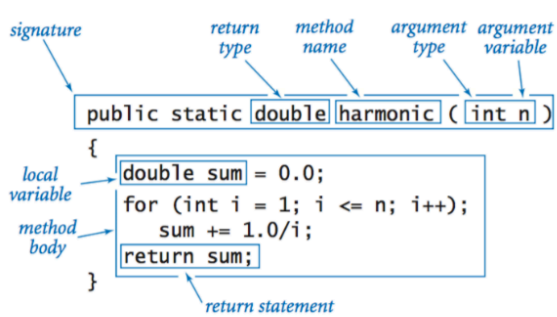
(2) Better organization: you have labelled various parts of your code, so you can understand the whole thing better – you can just focus on coding one method at a time!

21

Java Data Types: Methods

Computer Science

Anatomy of a method definition:



If a method does not return a value, then the return type is void, and the return statement (without an argument) is optional.

22

Java Data Types: Methods

Computer Science

Methods, like operators, can be overloaded:

```
public class OverloadTest {

    static int sum(int n, int m) {
        System.out.println("Calling sum(int n, int m)...");
        return (n+m);
    }

    static double sum(double x, double y) {
        System.out.println("Calling sum(double x, double y)...");
        return (x+y);
    }

    public static void main(String[] args) {

        System.out.println("\nTry sum(2,3)...");
       int n = sum(2, 3);
       System.out.println("Returns " + n);

       System.out.println("\nTry sum(2.3, 3.1)...");
       double x = sum(2.3, 3.1);
       System.out.println("Returns " + x);

       System.out.println("\nTry sum(2, 3.1)...");
       double y = sum(2, 3.1);
       System.out.println("Returns " + y);
    }
}
```

```
Try sum(2,3)...
Calling sum(int n, int m)...
Returns 5

Try sum(2.3, 3.1)...
Calling sum(double x, double y)...
Returns 5.4

Try sum(2, 3.1)...
Calling sum(double x, double y)...
Returns 5.1
```

sum(2, 3.1)

↓ Widening Conversion

sum(2.0, 3.1)

23