CSE 20
Intro to Computing I

Lecture 4 – Type Conversion (cont.)
Intro to Strings
Error Types

Announcements

- Today: Type Conversion (cont.), Intro to Strings, Error types
- Labs
  - Lab 3 due this week (9/29 – 10/5) with and additional 3 days grace period
  - Lab 4 (Data Types) assigned this week
    - Due in one week (plus additional **3 days** grace period)
    - Make sure to demo your work to a TA (or me) after submission
    - Demo is REQUIRED to receive credit for assignment
- Reading Assignments
  - Reading 02 (2.6 – 2.18, 2.20) due Oct 7
    - Complete Participation Activities in each section to receive grade towards Participation
    - IMPORTANT: Make sure to **submit score to CatCourses** by using link provided on CatCourses

Type Casting - Up Conversion (review)

```
double first;                // Use "higher" type
first = 0;                   // 0 is also a valid double (0.0)
first = 1;                   // 1.0
double second = 0.5;
double result = first - second;
```

Up Conversion -> no information loss

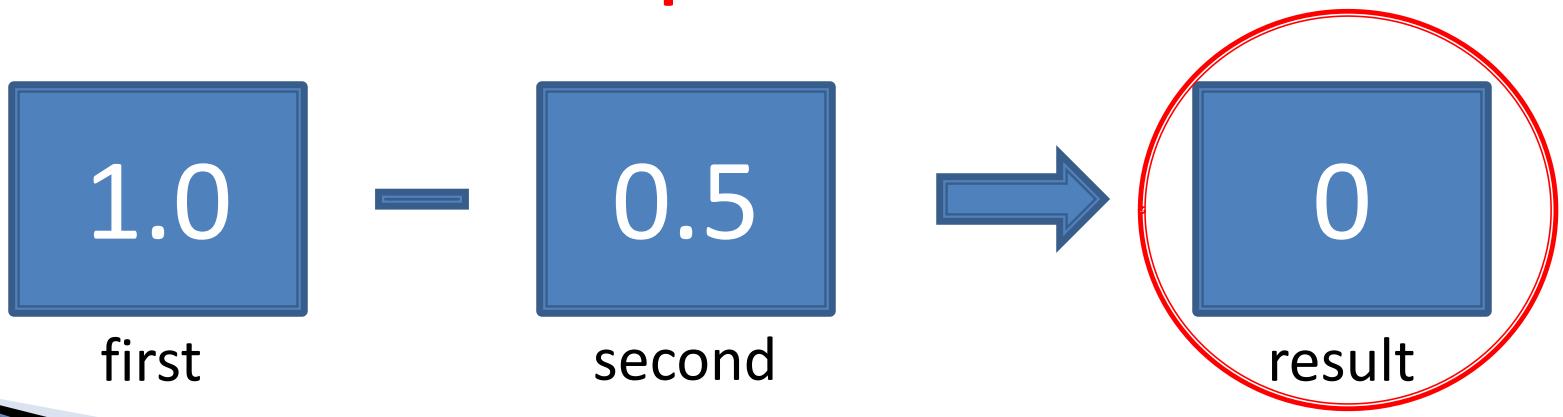
Type Casting - Down Conversion (review)

```
double first;
first = 0;
first = 1;
double second = 0.5;
int result = (int)(first – second);  // Using "lower" type needs
                                     // explicit cast
```

Forced it to be an **int** (explicitly)

A **double**

Down Conversion -> possible information loss



Type Conversions (review)

- Implicit – Up conversion
  - *double* d = 4; d is 4.0
  - *char* a = '}'; a is '}'
  - *int* i = 'A'; i is 65
  - *float* f = 'A'; f is 65.0
  - *double* e = 'A'; e is 65.0
- Explicit – Down conversion
  - a = (*char*)i; a is A
  - a = (*char*)f; a is A
  - a = (*char*)d; a is EOT
  - i = (*int*)f; i is 65
  - i = (*int*)e; i is 65
  - f = (*float*)e; f is 65.0

There is no loss of information

There may be loss of information

Addition : + (Data Types)

- short + short → int
- short + int → int
- char + char → int
- int + int → int
- int + float → float
- string + boolean → string
- string + (expression) → string
- string + char + char → string + char → string
- char + char + string → int + string → string

Highest data type in the expression

Output Variable

```
double first;
first = 0;
first = 1;
double second = 0.5;
int result = (int)(first – second); // Loose fractional part
System.out.println(result);
```

Console Output: `0`

Output Message

```
double first;
first = 0;
first = 1;
double second = 0.5;
int result = (int)(first – second);
System.out.println("Result is ");
System.out.println(result);
```

Console Output:

```
Result is
0
```

Output Message – Corrected

```
double first;
first = 0;
first = 1;
double second = 0.5;
int result = (int)(first – second);
System.out.print("Result is ");
System.out.println(result);
```

Console Output: `Result is 0`

Output Message using +

```
double first;
first = 0;
first = 1;
double second = 0.5;
int result = (int)(first – second);
System.out.println("Result is " + "result");
```

This is a string literal.

Console Output: `Result is result`

Output Variable using +

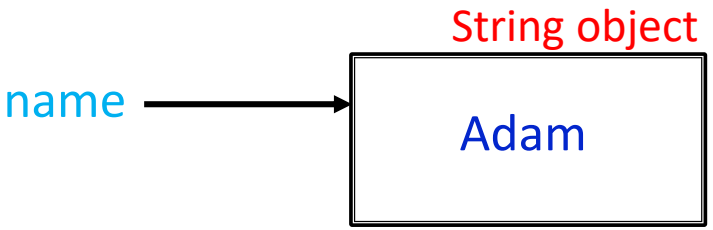
```
double first;
first = 0;
first = 1;
double second = 0.5;
int result = (int)(first – second);
System.out.println("Result is " + result);
```

Console Output: `Result is 0`

String Variable

- String: sequence of characters
- String literal: character sequence surrounded by double quotes
  - "Adam", "Feb 23, 2019", "Today is \n a \t Wednesday"
- String variable: pointer to a String object storing the sequence of characters

```
String name = "Adam";
```



- Whitespace character: used to represent horizontal and vertical spaces in text
  - Spaces, tabs, newline, etc.

String Input

```
Scanner kb = new Scanner(System.in);       // Create Scanner

// Declare String variable (not exactly but okay to think this way for now)
String mystr;

// Getting string without whitespaces
mystr  = kb.next(); // Skip initial whitespace(s), get characters until next
                    // whitespace is seen (leaving that whitespace in the
                    // input)

// Getting string with whitespaces
mystr  = kb.nextLine();  // Get all remaining text on current input line, up
                         // to next \n character (which is removed from
                         // input but not put in mystr).
```

String Variable (1)

```
double first;
first = 0;
first = 1;
double second = 0.5;
int result = (int)(first – second);
String outMessage = "Result is ";
System.out.println(outMessage + result);
```

Console Output: `Result is 0`

String Variable (2)

```
double first;
first = 0;
first = 1;
double second = 0.5;
int result = (int)(first – second);
String outMessage = "Result is " + result;
System.out.println(outMessage);
```

Save output string as outMessage first.
Then print it out.

Console Output: `Result is 0`

Putting it all Together

```
Scanner keyboardInput = new Scanner(System.in); //Create Scanner
System.out.print("What is your name? ");
String myName = keyboardInput.next();
System.out.print("Where do you live " + myName + "? ");
String myCity = keyboardInput.next();

System.out.println("\n" + myName + " lives in " + myCity + ".");
```

Output:

```
What is your name? Santosh
Where do you live Santosh? Atwater

Santosh lives in Atwater.
```

Putting it all Together

```
Scanner input = new Scanner(System.in); //Allow user input
System.out.print("What is your name? ");
String name = input.next();
System.out.print("Where do you live " + name + "? ");
String city = input.next();

System.out.println("\n" + name + " lives \n in " + city + ".");
```

Output:

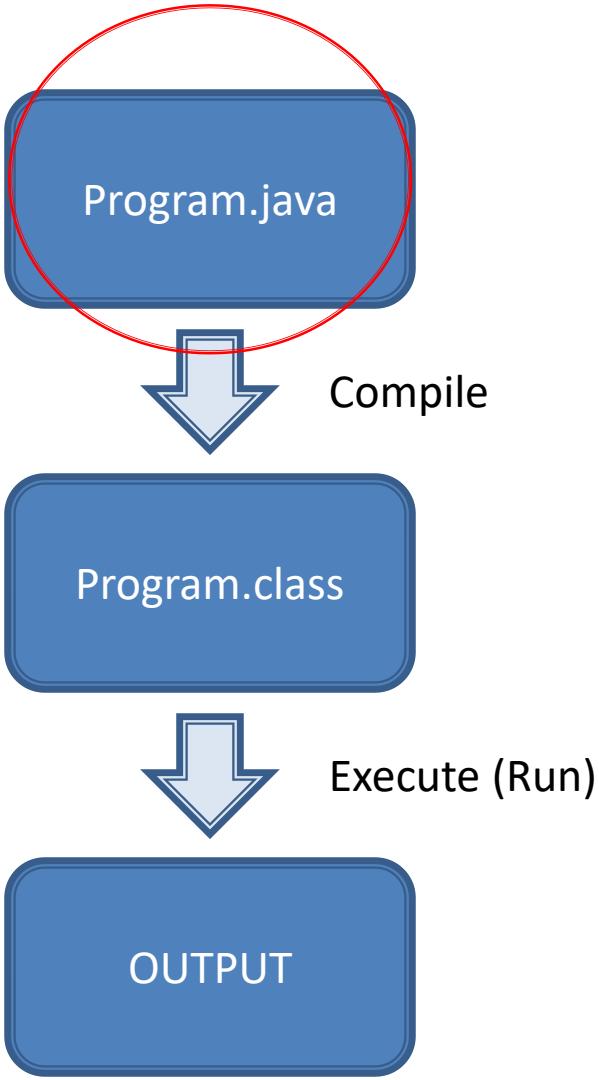
```
What is your name? Santosh
Where do you live Santosh? Atwater

Santosh lives
in Atwater.
```

How Java Programing Works?

- Java Execution Model
  - Capitalized program name

Submit this file!



Types of Errors in Programing

- **Compile-time errors:** Errors found when the program is being compiled.
  - Example: Syntax errors
  - Depending on your setup, Eclipse may catch some or all of these as you type.
- **Run-time errors:** The program compiles correctly, but an error results when run.
  - Example: Errors in utilizing memory
- **Logical errors:** The program compiles OR runs, but behaves unexpectedly.
  - Example: you intended the program to print "***Hello world!***" but it prints "***Goodbye cruel world***"