Topic 1:
Java Data Types

Learning Goals (Week 1):

- **Identify data types based on value**
- **Map variables to the current values**
- Perform basic operations on variables
- Create and use Java and user-defined methods
- Format Printed Output
- Obtain and process user input from the console
- Use booleans, conditionals, and compound conditionals correctly
- Select and implement different types of loops depending on scenario
- Use special String and Math operations
- Successfully implement and manipulate java arrays

Data Types

Java is **strongly typed:** every value has a type associated with it

Primitive Data Types

String
int
float
double
byte
short
char
boolean

Data Types

Java is **strongly typed:** every value has a type associated with it

Primitive Data Types

String
int
float
double
byte
short
char
boolean

String

- More on strings laters
- for now:
  - double quotes
  - + to concatenate
  - can't be broken up over lines

Examples:

- "Hello"
- "Hi" + " " + "There"
- "You can't do
  this"

Data Types

Java is **strongly typed:** every value has a type associated with it

Primitive Data Types

String
int
float
double
byte
short
char
boolean

Integer Types

int ± 2147483647 **<- most common**
long ± 9223372036854775807
short ± 32767
byte ± 127

Examples

- 100 (could be any of them)
- 2000 (all but byte)
- 50000 (int or long)
- 5000000000**L** (long)
- 12345678901234567890**1**L (none: too big!)

Data Types

Java is **strongly typed:** every value has a type associated with it

Primitive Data Types

String
int
float
double
byte
short
char
boolean

Floating-point Types

float: approx. 7 significant digits
double: approx 8 signification digits

Examples

- 1.0 (double) **<- most common**
- -0.34E-5 (double)
- 2.0**d** (double)
- 1.0**f** (float)
- -0.34E-5**f** (float)
- 2.0**f** (float)

Data Types

Java is **strongly typed:** every value has a type associated with it

Primitive Data Types

String
int
float
double
byte
short
char
boolean

Other types

char: a single character in single quotes
boolean: true/false with no quotes (more on these later)

Example

- ‘a’ (char)
- ‘0’ (char)
- ‘ ‘ (char)
- ‘&’ (char)
- true (boolean)
- false (boolean)

Declaring Variables

- A place to store these literal values
- Should always use camelCaseNamingConventionForVariables
- first letter is lower case and every other word is capitalized
- USE GOOD NAMES! (It helps everyone)

Example:

- variable that stores:
  - a person's first and last name: **String fullName;**
  - a person's age: **int age;**
  - a person's salary: **double yearlySalary;**
  - a person's middle initial: **char middleInitial;**
  - if a person is of legal drinking age: **boolean isLegalDrinkingAge;**

Variable Assignment

- **Declaration** and **assignment** can be separate
  - Think like a storybook: *The heroine wore a cape. It was purple.*
  - The cape is declared in one sentence, the colour in a second.

```
String fullName;
fullName = “Janice Feathers”;
```

or it can be declared in the same command.

- storybook example: *The heroine wore a purple cape.*

```
String fullName = “Janice Feathers”;
```

Comments

Could variable names help but sometimes we need more.

- Enter: Comments!

```
// Our heroine’s name is Janice Feather
String fullName = “Janice Feathers”;
```

If we have lots of comments about one code block we use a COMMENT BLOCK

```
/*
 * This is a comment block. We only need /**/ but the
 * extra *’s on these line does make it line up quite nicely
 */
 String fullName = “Janice Feathers”;
```

Printing Variables

Thankfully, this is pretty simple!

We print variables just like literals

```
// Our heroine's name is Janice Feather
String fullName = "Janice Feathers";

System.out.println("Our Heroine's name is:....");
System.out.println(fullName);
```

We can even combine them into one println

```
// Our heroine's name is Janice Feather
String fullName = "Janice Feathers";

System.out.println("Our Heroine's name is: " + fullName);
```

Updating Variables

- **Updating Variables is easy peasy!**
  - Think like a storybook: *The heroine wore a purple cape. She spilled coffee on it, so she changed into her gold cape.*
  - The cape and colour is declared but then something happens that requires as update

```
String capeColour = “purple”;
System.out.println(“Our Heroine’s cape is ” + capeColour);

System.out.println(“Oh no! Coffee Spill! Time to get changed!”);
capeColour = “gold”;
System.out.println(“Now our Heroine’s cape is ” + capeColour);
```

Pause & Practice

Answers on next slide (not shown in recording)

- **Prac. 1**:
  - make a program that has an **age** variable set to 17.5
  - What type of variable is this going to be?
  - print the variable out in a sentence that says “My age is 17.5”
- **Prac. 2:**
  - make a program which has **two chars** with values ‘a’ and ‘z’
  - Print out a sentence that says “The english alphabet goes from a - z”.

Casting Variables

- Consider the above example of someone with an age of 17.5
  - Schools do not care about that decimal value. They want to know you're 17.
  - How can we tell the school you are 17?

```
double age = 17.5;
age = 17; // will actually be stored as 18.0
```

- What can we do? CASTING!

```
double age = 17.5;
int intAgeVersion = (int)age; // becomes 17 – integer truncation

int age = 17;
double doubleAgeVersion = age; // becomes 17.0
```

- No cast needed when going from smaller (int) to bigger (double)
- **Order: double > float > long > int > short > byte**