

CISC124 – Today's Topics		
<ul style="list-style-type: none"> • Quiz 1 • More on accessor and mutator methods • Type casting 		
Winter 2019	F. de la Parra	1

•1

Quiz 1		
<p>Location and Hours:</p> <ul style="list-style-type: none"> • Quiz 1 will be written in Jeffery 155, at the beginning of your lab session • Mon Jan 28, 9:30 am – 10:30 am • Mon Jan 28, 2:30 pm – 3:30 pm • Tue Jan 29, 8:30 am – 9:30 am • Wed Jan 30, 2:30 pm – 3:30 pm <p>Format:</p> <ul style="list-style-type: none"> • One coding question (i.e., write a method) • A set of multiple choice questions 		
Winter 2019	F. de la Parra	2

•2

Quiz 1		
<p>Topics (Everything covered until Jan 25) :</p> <ul style="list-style-type: none"> • Java environment basics (tools, libraries, development cycle) • Basic structure of java programs • Console I/O. Formatted console output • Class declaration and members • Visibility and storage modifiers (public, static, final, etc.) • Primitive and class types • Expressions. Operators • Arrays (1D) of primitive and class types • Branch and loop statements: if-else, switch, while, for • Encapsulation. Accessor and mutator methods • Type casting 		
Winter 2019	F. de la Parra	3

•3

More on Accessors and Mutators		
<ul style="list-style-type: none"> • In general, accessor methods can access one or more attributes in one invocation • An accessor method can do some processing with the attributes accessed and return a derived value(s) <ul style="list-style-type: none"> • Examples: compute an average, upper-case or concatenate string attributes, etc. 		
<ul style="list-style-type: none"> • In general, mutator methods can take one or more arguments in one invocation • A mutator method can do some processing with the arguments passed to it and update one or more attributes <ul style="list-style-type: none"> • Examples: check that arguments are in valid ranges, check that arguments will not make attributes to produce invalid future operations 		
Winter 2019	F. de la Parra	4

•4

Type casting		
<ul style="list-style-type: none"> • Type casting operation converts a value of a given type into a value in a different type • In some situations casting affects the intended value of a variable in a different type. • Implicit → <code>float var1 = 1 * 2 * 3 + 3.1415f;</code> • Explicit → <code>float var2 = 10/float(3);</code> • Automatic type coercion → <code>float var3 = 10;</code> • Illegal attempt of type coercion → <code>int var4 = 4.5;</code> (Needs explicit casting → <code>int var4 = (int)4.5;</code> var4 is assigned a 4) • Cannot cast class types to primitive types 		
Winter 2019	F. de la Parra	5

•5