CECS 174 – Lecture 4 – Type Casting

Operators and Data Types

We may need to mix data types when adding values together and assigning them to variables. The compiler will handle some combinations, but others will need to be type casted otherwise the compiler will return errors.

Combination:	Returns:	Assignment:
int + int;	int	int = int + int
double + double;	double	double = double + double
int + double;	double	int = double + int // error
int + double;	double	double = double + int // ok

Type Casting

In Java, it is acceptable to assign an integer value to a double variable. But when the value is stored in memory, the integer will be converted into a double by adding on a decimal point and a zero.

```
double price = 5;  // stored as 5.0
```

However, if you try to assign a double value to an integer, the compiler will return an error.

```
int value = 4.32;  // error
```

There is a way around this though, which is by type casting the value.

```
int value = (int) 4.32;  // stored in value as 4
```

This is equivalent to telling the compiler that you understand that you are potentially losing information in this operation. This will simply truncate off the decimal portion and assign the remaining integer to the variable. It does **NOT** round the number to the nearest integer.

Other types may be used in a similar manner.

```
double value = (double) 5 / 2;  // stored as 2.5

char letter = (char) 76;  // stored as 'L'
```

Example: What value is assigned? (If an error is produced, write error).

___________ `int value = 9.66;`

___________ `double price = 7;`

___________ `int num1 = (int) 1.99;`

___________ `double pct = 3/4*100;`

___________ `double pct = (double)(3/4)*100;`

___________ `double pct = ((double)3/4)*100;`