Practice with variables and types

1. Types. For each literal or expression, state its type (`String`, `int`, `double`, or `boolean`).

Expression	Type	Expression	Type
`387`		`"pancakes"`	
`true`		`45.0`	
`"14"`		`87.98515`	
`"false"`		`15 >= 71`	
`31.6 + 7`		`(double)(int)93.2`	

2. Declaring and using variables. Only one of the following code snippets is valid (i.e., will compile without errors). Which is it, and what's wrong with each of the others?

Code snippet A:

```
int x = 3;
int y = 17;
int x = x + y;
```

Code snippet B:

```
int num = 42;
double anotherNum = 81;
num = anotherNum - num;
```

Code snippet C:

```
int years = 18;
int months = 7;
double totalAge = years + months/12.0;
```

Code snippet D:

```
int p = 5;
int q = 43.7;
p = q;
```

3. Casting. For each of the following, add a cast to fix the type error.

```
int i = 5;
double j = 21.3;
i = i + j;

int totalLabScore = 84;
int numLabs = 10;
double averageScore = totalLabScore/numLabs;
```

4. Using variables. Write a piece of code that asks the user to enter their height (as a number of feet and a number of inches, i.e., 5 7) and tells them their height in meters (i.e., 1.7018). (Note: there are 12 inches in a foot, and there are 3.28 feet in a meter.)

Practice with `if` statements

1. Are they equivalent? Which of the following snippets of code do the same thing? That is, which print the same message(s) on every single input value for `num`?

Code snippet A:

```
int num = keyboard.nextInt();
if(num > 54) {
   if(num > 82) {
      System.out.println("one");
   }
   else {
      System.out.println("two");
   }
}
else {
   System.out.println("three");
}
```

Code snippet B:

```
int num = keyboard.nextInt();
if(num > 82) {
   System.out.println("one");
}
else if(num > 54) {
   System.out.println("two");
}
else {
   System.out.println("three");
}
```

Code snippet C:

```
int num = keyboard.nextInt();
if(num < 54) {
   System.out.println("three");
}
if(num > 82) {
   System.out.println("one");
}
else {
   System.out.println("two");
}
```

Code snippet D:

```
int num = keyboard.nextInt();
if(num > 54) {
   if(num < 82) {
      System.out.println("two");
   }
}
else if(num > 82) {
   System.out.println("one");
}
else {
   System.out.println("three");
}
```

2. Old enough? Write some code that asks the user for their age and then prints out whether they are old enough to:

1. Vote (age 18)
2. Get a driver's license in MA (age 16)
3. Rent a car (age 25)
4. Drink legally (age 21)

3. Scope. Determine whether each of the following code snippets will compile successfully. If not, correct the error. Then determine what prints.

Code snippet A:

```
int i = 5;
if(i > 2) {
   i = i * 7;
}
System.out.println(i);
```

Code snippet B:

```
int i = 8;
if(i % 2 == 0) {
   int j = 4;
}
System.out.println(i + j);
```

Code snippet C:

```
int x = -3;
int y = -2;
if(x * y > 0) {
   int z = x + y;
   y = z * 2;
}
System.out.println(x + " " + y);
```

Code snippet D:

```
int num1 = 42;
int num2;
if(num1 < 10) {
   num2 = 3;
}
System.out.println(num2);
```

4. Seasons. Write some code that asks the user to enter the current month (as an `int`, 1=January and 12=December) and then prints the season (Winter for Dec-Feb, Spring for Mar-May, Summer for June-Aug, Fall for Sep-Nov).

Practice with boolean expressions and order of operations

1. `true` or `false`? Evaluate each of the following boolean expressions when `int x = 4` and `int y = 6`.

```
x <= 5 || y + x > 12 && !(x % 3 == 1)

y/x > 1 && x != 17

!(y % 4 % 2 == 0 || !((x + y / 3) >= y))
```

2. What prints? What prints when each of the following pieces of code runs?

```
int month = 2;
int day = 20;
System.out.println("Tomorrow is " + month/day);

int month = 2;
int day = 20;
System.out.println("Tomorrow is " + month + "/" + day);

int age = 19;
System.out.println("In three years your age will be: " + age + 3);
System.out.println("Your age in three years is: " + (age + 3));
System.out.println(age + 3 + " is your age in three years");
```

3. Broken code. Assume that the declaration and initialization `int x = 7;` appears somewhere earlier in the code. None of the following pieces of code will compile without error. Make a small change to fix the error without changing the intended meaning of the code.

```
if(!x < 17) {
   System.out.println("yes");
}

int y = 4;
if(x < -1 || < y) {
   System.out.println("yes");
}

if(10 >= x > 2) {
   System.out.println("yes");
}
```

Practice with loops

1. What prints? Consider the following `while` loop. What is the output?

```
int i = 0;
while(i < 5) {
   int j = 0;
   while(j < 3) {
      System.out.print(i + j);
      j++;
   }
   System.out.println();
   i++;
}
```

2. `while` and `for`. Translate the following `while` loop into a `for` loop that does the same thing.

```
int i = 0;
while(i < 100) {
   System.out.println(i * 7);
   i++;
}
```

3. Pretty patterns. Write some nested `while` loops that print the following pattern. Then do the same thing with `for` loops.

```
******
*    *
*    *
*    *
*    *
******
```

4. Comparing code. Do the following two pieces of code do the same thing? If so, what do they both do? If not, change the second in some small way so that they do the same thing.

Code snippet A:

```
for(int i = 1; i <= 10; i++) {
   System.out.println(i);
}
```

Code snippet B:

```
for(int i = 10; i > 0; i--) {
   System.out.println(10 - i);
}
```

5. Improving code that already works. What is stylistically not so great about the following piece of code? Fix it to improve the code style without changing what it does.

```
int i = 0;
while(i < 1) {
   System.out.print(i + " ");
   i++;
}
System.out.println();
i = 0;
while(i < 2) {
   System.out.print(i + " ");
   i++;
}
System.out.println();
i = 0;
while(i < 3) {
   System.out.print(i + " ");
   i++;
}
System.out.println();
i = 0;
while(i < 4) {
   System.out.print(i + " ");
   i++;
}
System.out.println();
```

6. Finding factors. Write some code to print out all of the factors of all numbers from 1 to 100. For example, the first few lines of your code's output should be:

```
1: 1
2: 1 2
3: 1 3
4: 1 2 4
5: 1 5
```

Do this twice, once with `while` loops and once with `for` loops.

Practice with methods

1. Method headers. Write the *header* for each of the following methods:

- A method that, given a length and a width, prints out a rectangle of $\star$s of the specified size.
- A method that, given an integer n, computes and returns $n!$ ($n!$, read as "n factorial," is the product of all integers from 1 to n).
- A method that, given an integer x, returns whether or not x is prime.
- A method that, given an array of doubles, finds and returns the largest number stored in that array.
- A method that, given a string and a character (stored in a variable of type `char`, which we haven't talked about yet—a `char` is a single letter, number, or other symbol), returns a count of the number of times that character appears in the string.

2. Method bodies. Now write the *body* of the first four methods listed above. (The fifth method, which uses arrays, is something that you should be able to write at this point but is more involved than what I expect you to know about arrays on this exam. The sixth method, which involves manipulating Strings, is not something that I expect you to know how to write at this point.)

3. Programs that use methods. Write a program that asks the user to enter two prime numbers, then prints a rectangle of the dimensions specified by the user. Your program should ensure that the user behaves well (i.e., ask the user to try again until they've actually entered positive prime numbers). You can assume that the user will only enter integers. Your program should call the methods you wrote above, and you can feel free to write new methods if you'd like.

Practice with arrays

1. Creating arrays. Write a piece of code to declare and allocate an array of size 10.

2. Storing values in arrays. Write a piece of code to store multiples of 2 in an array.

3. Printing arrays. Write a method that prints the contents of an array of ints.