Sunday, December 30, 2018 1:52 PM

3Parameters

```
public static void power(int base, int exp) {
   result = 1;
   for (int i = 1; i <= exp; i++) {
      result *= base;
   }
   System.out.println("base to the " + exp + " = " + result);
}

for (int base = 0; base <= 16; base++) {
   power(2,base);
}
```

Return

```
public static int power(int base, int exp) {
   result = 1;
   for (int i = 1; i <= exp; i++) {
      result *= base;
   }
   return result;
}
```

Java's Math Class

Essential

method headerexamplesummary

< any > abs(< any > x)int x = Math.abs(-2);returns the absolute value of x

double pow(double base, double exp)double x = Math.pow(10,2);returns the result of base to the power exp

double sqrt(double x)double x = Math.sqrt(25.0);returns the square root of x

double random()double x = Math.random();returns a randomly generated number between 0 and 1

Very Useful

Math.PI = 3.14159…
Math.E = 2.71828
method headerexamplesummary
int round(double x)int x = Math.round(0.66);if x's decimal is .5 or more round up to the next int otherwise round down
< any > max(double x, double y)int x = Math.max(5,10);returns the larger of 2 parameters
< any > min(double x, double y)int x = Math.min(-10, 10);returns the smaller of 2 parameters
double sin(double radians)double x = Math.sin(Math.PI / 2.0);sine of angle in radians
double cos(double radians)double x = Math.cos(Math.PI / 3.0);cosine of angle in radians
double tan(double radians)double x = Math.tan(Math.PI / 4.0);tangent of angle in radians
double toDegrees(double radians)double x = Math.toDegrees(Math.PI);convert radians to degrees
double toRadians(double degrees)double x = Math.toRadians(180);convert degrees to radians

Java's String Methods

The individual characters of a string are numbered from 0 to length()-1; this number is called the index of the character in the String.

Expressionreturnsexplanation
"example".substring(0,4)"exam"first 4 chars
"example".substring(1,2)"x"1 char starting at index 1 (second char)
"example".substring(3,3)""0 chars
"example".substring(4,7)"ple"3 chars starting at index 4
"example".substring(4)"ple"index 4 to the end
"example".substring(6)"e"last char (index = length-1)
"example".substring(7)""empty String at end
"example".substring(-1,4)error!-1 is not a valid index
"example".substring(0,8)error! 8 is not a valid index
"example".substring(4,3) error! start index > end index

Expression	returns	explanation
"example".indexOf("e")	0	first character index is 0
"example".indexOf("m")	3	fourth character index is 3
"example".indexOf("q")	-1	-1 means not found

"example".indexOf("E")	-1	uppercase does not match lowercase
"example".indexOf("")	0	empty String returns beginning
"example".indexOf('p')	4	char argument is OK
"example".indexOf("ple")	4	returns start of matching substring
"example".indexOf("plx")	-1	entire substring must match

method	example	example results	summary
boolean equals(String)	"hello".equals("goodbye")	false	true if strings are identical, case sensitive
char charAt(int index)	"hello".charAt(2)	'l'	char at index
int lastIndexOf(String str)	"hello".lastIndexOf("l")	3	position of last occurrence of String
String toLowerCase()	"HELLO".toLowerCase()	"hello"	new string converted to all lowercase
String toUpperCase()	"hello".toUpperCasE()	"HELLO"	new string converted to all uppercase
String replace(char old, char new)	"hello".replace('l', 'x')	"hexxo"	new string with all occurrences of old replaced by new

Recursive Algorithms

```
public static int fibonacci(int n) {
   if (n == 1) {
      return 1;
   } else if (n == 2) {
      return 1;
   } else {
      return fibonacci(n-1) + fibonacci(n-2);
   }
}

public static int factorial(int n) {
   if (n == 1) {
      return 1;
   } else {
      return n * factorial(n-1);
   }
}
```