1. // [comment]	10. Private	64-bit number with decimals.
Single line comment.	Can only be changed by a method.	
2. /* [comment] */		19. float
Multi line comment.	11. int	32-bit number with decimals.
	Can store numbers from 2^-31 to 2^31.	
3. public		20. protected
This can be imported publically.	12. fields are attributes	Can only be accessed by other code in the package.
4. import [object].*	13. boolean	24.5
Imports everything in	Can have true or false as the value.	21. Scanner
object.		This lets you get user input.
	14 ()	22[akinst
5. static	14. { } These are used to start and	22. new [object constructor]
Going to be shared by every [object].	end a function, class, etc.	This will let you create a new object.
6. final	15. byte	
Cannot be changed;	These can store from -127 -	23. System.in
common to be defined with all uppercase.	128.	This lets you get data from the keyboard.
	16. long	
7. double	Can store numbers from	24. public [class]()
Integer with numbers that can have decimals.	2^127 to 2^-127.	This will be the constructor, you use it to create new objects.
	17. char	•
8. ;	Just lets you put in one	25. super()
Put after every command.	chracter.	This will create the
	18. double	superclass (the class it's inheriting).
9. String	16. double	crem _b ,

Just a string of characters.

26. extends [class]	35. public static void main(String[] args)	44. <
Makes the object a subclass of [object], [object] must be a superclass.	This is your main function and your project will start in here.	This means less than.
		45. >
27. ++		This means greater than.
Will increment the amount.	36. System.out.print([text])	
	This prints stuff but there is no line break. (/n)	46. >=
28		This means greater than or equal to.
Will decrement the amount.	37. \n	
29. += [amount]	Called a line break; will print a new line.	47. [inputVarHere].hasNextLine ()
Increment by [amount]		This will return if there is a
	38. \t	next line in the input.
30= [amount]	This will print a tab.	
Decrement by [amount]	22 (6/1 1)	48. this
	39. if ([condition])	Refer to the class that you
31. *= [amount]	This will make it so if [condition] is true then it'll	are in.
Multiply by [amount]	keep going.	
		49. [caller].next[datatype]()
32. /= [amount]	40. &&	This will get the [datatype] that you somehow
Divide by [amount]	This means and.	inputted.
33.	41. !	50. Create getters and
System.out.println([text])	This means not.	setters
Will print something to the output console.		This will create the get methods
	42.	for every checked variable.
34. +	This means or.	
Can be used for concatenation. (ex. "6" + [var_here])	43. ==	51. [caller].hasNext[datatype]()
	This means equal to.	

This will return if it has the correct datatype within the input.	This will parse [number] into the [numbertype] with [string].	66. for ([number]; [condition]; [operation]) This will start at [number] and then do [operation]
52. overloading	59. ^	until [condition] is met.
If you have different parameters you can call them whatever way you want.	Return true if there is one true and one false.	67. continue This will just go back to the enclosing loop before
	60. !=	reaching other code.
53. parameters	Not equal too. (NEQ)	
These are the inputs of your		68. while ([condition])
function.	61. ([condition]) ? [amount] : [var]	This will basically do something while [condition]
54. ([datatype])[variable]	This will be like a shortcut way to an if statement.	is true.
This will convert [variable] into [datatype]. Also known		69. void
as casting.	62. switch([variable])	This means no return type.
55. Math.random()	This will do stuff with specific cases. (e.g.	,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
	switch(hi){ case 2: (do	70. return
Generate an extremely percise string of numbers between 0 and 1.	stuff)}) 63. case [value]:	This will return something when you call it to where it was called from .
56. Primitives	This will do stuff if the case	
Just the basic data types	is the case.	71. do { } while ([condition])
which are not objects.	64. break	Guarantees it will execute once even if [condition] isn't met.
57. [x].toString()	Put that when you want to leave the loop/switch;	
Will convert [x] into a string.	should be at end of case.	72. printf("%[type] stuff here bah bla", [variable here])
58. [number].parse[numbertyp e]([string])	65. default [value]:	This will let you use [variable here] with %s being where.
	This will do stuff if none of the cases in the switch statement was made.	

73. System.out.printf([text])

Another way to print? // didn't quite get but ok then

This will get how long something is, text, amount of indexes in array, etc.

74. [type] [returntype] [name]([parameters]) {

This is a way to create a method.

80. Arrays.copyOf([array], indexes);

This will copy the array and how many indexes into another array.

75. [type][[indexes]]

This will create an array with [indexes] amount of indexes; default infinite.

81. Arrays.toString([array])

Convert the whole array into one huge string.

76. int[] something = new int[20];

This will just make an array of ints with 20 ints in it.

82.

Arrays.binarySearch([array], [object])

This will search for [object] in [array].

77. for ([object] [nameOfObject] : [arrayOfObject]) {

This will iterate through all of the arrayOfObject with object in use incrementing by 1 until done.

78. [object][[1]][[2]][[3]]
[name] = {[value] [value]
[value] \n [value] [value]
[value]}

[1] is how many down in array, [2] how many accross in array, [3] how many groups