Outputting text

Python
```
print "Hello"
```

Matlab
```
disp('Hello');
```

Java
```
System.out.println("Hello");
```

Creating Variables

Python
```
someNumber = 0
someDecimal = 2.1567
oneCharacter = 'a'
someText = "abcd"
someTrueOrFalse = True
```

Matlab
```
someNumber = 0;
someDecimal = 2.1567;
oneCharacter = 'a';
someText = 'abcd';
someTrueOrFalse = true;
```

Java
```
int someNumber = 0;
double someDecimal = 2.1567;
char oneCharacter = 'a';
String someText = "abcd";
boolean someTrueOrFalse = true;
```

Note that in Java, it is necessary to declare a variable before use. So the first time you use a variable, you code something like: int someNumber = 0;. Once the variable is declared, you can change its value by saying someNumber = 7;. When you declare a variable in Java, you must specify the variable's type (e.g. int, double, char, etc.).

If Statements

Python
```
if someNumber == 7:
    print "var is seven"
else:
    print "var is not 7"
```

Matlab
```
if someNumber == 7
  disp('var is seven');
else
  disp('var is not 7');
end
```

Java
```
if(someNumber == 7) {
  System.out.println("var is seven");
} else {
  System.out.println("var is not 7");
}
```

We have indented the Matlab and Java examples slightly for clarity, but only in Python does indentation actually matter to the program's meaning.

For Loops

Python
```
for i in range(1,11):
    sum = sum + i
```

Matlab
```
for i = 1:10
  sum = sum + i;
end
```

Java
```
for(int i = 1;i < 11; i++) {
  sum = sum + 1;
}
```

While Loops

Python
```
while(not processDone):
    processDone = doOneMoreStep()
```

Matlab
```
while(~processDone)
  processDone = doOneMoreStep();
end
```

Java
```
while(!processDone) {
  processDone = doOneMoreStep();
}
```

Using Lists/Arrays

Python
```
list = []
list.append("string to add")
firstItem = list[0]
```

Matlab
```
list = {};
list = [list 'string to add'];
firstItem = list(1);
```

Java
```
ArrayList<String> list = new ArrayList<String>();
list.add("stringToAdd");
String firstItem = list.get(0);
```
or if you happen to have a fixed number of elements…
```
String[] array = new String[10];
array[0] = "string to add";
String firstItem = array[0];
```

Getting Third Character from a String

```
myThirdChar = myString[2]
```

```
myThirdChar = myString(3);
```

```
char myThirdChar = myString.charAt(2);
```

Updating a Particular Character from a String

```
listVersion  = list(myString)
listVersion[2] = 'q'
myString = ''.join(listVersion)
```

```
myString(3) = 'q';
```

```
char[] arrayVersion = myString.toCharArray();
arrayVersion[2] = 'q';
myString = new String(arrayVersion);
```

This is harder than you might expect in Java and Python because arrays are designed to be unchangeable. So rather than just updating the existing string we have to create a new string.

Converting a String to an Integer

```
myInt = int("1234")
```

```
myInt = str2num('1234');
```

```
int myInt = Integer.parseInt("1234");
```

Creating a Function

```
def myAdd(n1, n2):
    return n1 + n2
```

```
function result = myAdd(n1, n2)
  result = n1 + n2;
end
```

```
public int myAdd(int n1, int n2) {
  return n1 + n2;
}
```