These notes are going to introduce how to write a function using an idea that may completely mislead the weak minded student. All students must be particularly careful to NOTICE all of the warnings that appear in the lesson.
You are asked to write a Java function which is given an integer array as its parameter and which will compute the sum of the elements of that array and then return that sum.

The BAD IDEA is that we are going to solve ANOTHER PROBLEM INSTEAD

The DANGER is that you will confuse the two problems:

(1) Original problem: WRITE a FUNCTION
(2) Alternate problem: write a user program
Here is the ALTERNATE problem:

Write a program which will input an integer array from the user in brackets notation and will compute the sum of its elements and print out that sum.
Here is a comparison of the two problems - highlighting how they are different and how they are the same:

<table>
<thead>
<tr>
<th>ORIGINAL</th>
<th>ALTERNATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>You are asked to write a Java function which is given an integer array as its parameter and which will compute the sum of the elements of that array and then return that sum.</td>
<td>Write a program which will input an integer array from the user in brackets notation and will compute the sum of its elements and print out that sum.</td>
</tr>
</tbody>
</table>
Here are the essential differences again

<table>
<thead>
<tr>
<th>ORIGINAL</th>
<th>ALTERNATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>function</td>
<td>program</td>
</tr>
<tr>
<td>Given parameter</td>
<td>User input</td>
</tr>
<tr>
<td>return</td>
<td>output</td>
</tr>
</tbody>
</table>
By now (in this course) you should have no problem writing the ALTERNATE program as described (using JavaBlocks) - it is just a for-in loop with a simple summation in it.

```java
static Console cin = new Console();

public static void main(String[] args) {
    int[] data = cin.nextIntArray();
    int total = 0;
    for(int item : data) {
        total = total + item;
    }
    out.println("total="+total);
}
```
Here is the same alternate program broken into parts:

```java
static Console cin = new Console();

public static void main(String[] args) {
    int[] data = cin.nextIntArray();

    int total = 0;
    for(int item : data) {
        total = total + item;
    }

    out.println("total=\"+total+\")
}
```
We will TRANSFORM the user I/O program into a function through a series of moves

First we remove the programming stuff:

```java
int[] data = cin.nextIntArray();

int total = 0;
for(int item : data) {
    total = total + item;
}

out.println("total="+total);
```
Next we replace the User input part with a function heading - also making up a name `addEmUp` for our function.

```java
int addEmUp(int[] data) {
    int total = 0;
    for (int item : data) {
        total = total + item;
    }
    out.println("total="+total);
}
```
Next we replace the user output part by a return command:

```java
int addEmUp(int[] data) {
    int total = 0;
    for (int item : data) {
        total = total + item;
    }
    return total;
}
```
Finally we add the close brace defining the end of the function

```java
int addEmUp(int[] data) {
    int total = 0;
    for(int item : data) {
        total = total + item;
    }
    return total;
}
```
The same thing without all the blank lines showing the structure

```java
int addEmUp(int[] data) {
    int total = 0;
    for(int item : data) {
        total = total + item;
    }
    return total;
}
```
WARNING
A function does NOT imply Input/Output

Here is a program that fails - and would not do input/output in any case.

class BadIdea {
    static int addEmUp(int[] data) {
        int total = 0;  for(int item : data) {
            Total = total + item
        }
        return total
    }
    public static void main(String[] args) {
        int[] data;
        int sum = addEmUp(data);
    }
}
Here is what the original PROGRAM did - but using a function to compute the total.

```java
class OkayIdea {
    static Console cin = new Console();

    static int addEmUp(int[] data) {
        int total = 0; for(int item : data) {
            total = total + item;
        }
        return total
    }

    public static void main(String[] args) {
        int[] data = cin.nextIntArray();
        int sum = addEmUp(data);
        println("total=\"+sum);
    }
}
```

Note - all the function does is move the computational part out of the main routine into a separate place.

The programmer must still put in commands for the user input and output.
So why bother with a function at all - seems like unnecessary work.

BECAUSE:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Breaking up a program into smaller parts is a GOOD THING (trust 100,000 programmers)</td>
</tr>
<tr>
<td>2</td>
<td>We can do a more complicated program without much more work.</td>
</tr>
</tbody>
</table>
Now suppose we want a program to read in a bracketed array of integers and print its sum AND THEN to read in a second bracketed array of integers and print its average. The solution below uses the function that has been designed.

class GoodIdea {
    static Console cin = new Console();
    static int addEmUp(int[] data) {
        int total = 0; for(int item : data) {
            total = total + item;
        }
        return total;
    }
    public static void main(String[] args) {
        int data[] = cin.nextIntArray();
        println("total="+addEmUp(data));
        int more[] = cin.nextIntArray();
        println("avg="+addEmUp(more)/more.length);
    }
}
Another way to do program Two

class BetterIdea {
    static Console cin = new Console();
    static int addEmUp(int[] data) {
        int total = 0; for (int item : data) {
            total = total + item;
        }
        return total;
    }
    static void work(int[] data, String legend, boolean avg) {
        int ans = addEmUp(data);
        if (avg) ans = ans/data.length;
        println(legend+ans);
    }
    public static void main(String[] args) {
        int[] data = cin.nextIntArray();
        work(data,"total=",false);
        int[] more = cin.nextIntArray();
        work(more,"avg=",true);
    }
}

NOTE: Although a function does NOT AUTOMATICALLY do IO, it can do so if desired.
class GoodIdea {
    static Console cin = new Console();
    static int addEmUp(int[] data) {
        int total = 0; for(int item : data) {
            total = total + item;
        }
        return total
    }
    static void allwork(String legend, boolean avg) {
        int[] data = cin.nextintArray(); // NOW LOCAL
        int ans = addEmUp(data);
        if(avg) ans = ans/data.length;
        println(legend+ans);
    }
    public static void main(String[] args) {
        allwork("total=",false); // NO ARRAYS HERE
        allwork("avg=",true);
    }
}

Knowing how to divide up the program's work among a variety of functions is an ART - best acquired by lots of trial and error and practice.