ARM assembly practice

Add numbers

Write a program that adds three numbers of your choosing and saves the result into `R0`. You can move a number (`48`) into a register (`R0`) using the instruction:

```
MOV  R0, #48
```

Challenge: Are there numbers you can't move into a register this way? Note that you *can* move some pretty big numbers (like 131072). What does a compiler do if you set a variable to this number? (Try godbolt.org, perhaps using `armv7-a clang`)

Storing to memory

Write a program that writes the value 5 to memory location `0x1000`. You should be able to see the stored value if you click on the "memory" tab in VisUAL.

Adding to memory

Write a program that adds 2 to the value in the memory location `0x1000`. In C, this might look like

```
int* p = (int*)0x1000;
*p = *p + 2;
```

Conditional execution

Write a program that computes the absolute value of a number stored in `R0`.

Branches

Write code to loop through the numbers 1-10, and compute their sum.

In C, this might look like

```
int sum = 0;
for(int i = 1; i <= 10; i++){
    sum = sum + i;
}
```

Bonus problems

Sum of array

Write an ARM assembly program which computes the sum of 10 numbers stored in an array in memory. You can initialize the memory with code like:

```
DATA DCD 2, 3, 5, 7, 11, 13, 17, 19, 23, 29
```

Then you can use the label `DATA` as an immediate. For example, `MOV R0, #DATA` would put the address of the array into `R0`.

Copying arrays

Write an ARM assembly program that copies one array of 10 elements to another, as in the C code below:

```c
int array1[10];
int array2[10];
for(int i = 0; i < 10; i = i+1){
    array[1] = array[2];
}
```