ASSESSED INDIVIDUAL WORK

COMP1006
LAB EXERCISE #1

Steven R. Bagley

Submission Deadline:
17/10/2022 1200

Version: 1.00

These exercises are designed to be straightforward and are primarily to get you used to using Komodo on the school's Linux system and writing ARM assembly language. These exercises are all based around and expand on the simple ARM 'Hello World' program we saw previously, and expects you to be familiar with both loading and 'compiling' (actually, the **Compile** button should read assemble) source code in Komodo. If you are not yet familiar with Komodo I suggest you refer to the video on Moodle outlining how to use Komodo.

Getting Started

You will need to fork and clone the relevant project via `git` in the usual fashion (i.e. as you have previously done for COMP1005). In this case, the project to fork can be found at:

`https://projects.cs.nott.ac.uk/2022-COMP1006/2022-COMP1006-LabExercise01`

Once cloned, you will find skeleton `.s` files for each exercise. It is **strongly recommended** that you start from these skeletons otherwise your programs may not work against the marking test scripts.

Note: For all these programs you may well find it useful to start by generating a C version first to ensure the logic of the program is sorted before you start transliterating it into ARM assembler.

Exercise 1 — Hello World

The file `01-hello.s` contains the simple ARM assembly version of the classic 'Hello World' program, we've already seen. This exercise requires you to modify this program to add extra functionality. Currently when run the program prints out the messages:

```
Hello World
Goodbye Universe
```

You are to modify the source code of the program to make it also print out a third message in-between 'Hello World' and 'Goodbye Universe'. The new message should print out your name as shown below, and all three messages should be on separate lines. As an example output for the modified program you could aim for (but, I suspect, you are not called Steven Bagley so that line will need modifying, obviously…):

```
Hello World
My name is Steven Bagley
Goodbye Universe
```

Hint: Cheekily modifying the text string of one of the messages is not allowed, you will need to define a new string and provide the ARM code necessary to get Komodo to print it out (remember, the program is already printing things out…)

Exercise 2 — Ten green bottles

This exercise is based around a simple ARM program that should print out a verse from the song 'Ten green bottles' when run. Ideally, the output (in the features window) should be as follows:

```
10 green bottles hanging on the wall,
10 green bottles hanging on the wall,
And if one green bottle should accidentally fall,
There will be 9 green bottles hanging on the wall.
```

Unfortunately, the source code for this program has got somewhat jumbled up and the odd line deleted. As a result the program no longer produces the correct output. For this exercise, you should work out how the lines should be rearranged to produce the correct output. The corrupted source can be found in the cloned git repository as `02-bottle.s`.

Hint: This is a very simply debugging exercise in that it expects you to work out why the program is not working and then correct it. My advice is to use the *single-step* feature of Komodo a lot, and compare what is being printed with what you expect to be printed. Then adjust the order of the instructions until it begins to print the text above.

Another Hint: There is no need to alter any of the string definitions provided — it is possible to print the verse out using the strings provided.

Exercise 3 — The Mowing Song

This exercise gets you to write a simple ARM program that prints out a single verse from the 'The Mowing Song' (the expected output of the program can be found in Appendix B), similar to the 'Ten Green Bottles' program from the previous exercise. A skeleton program is provided in the file `03-mowing.s` (part of the git repository you cloned) which you should use to enter your program.

This will program will work in a similar way to the program above, with a few differences. Firstly, you **must** print out the numbers using `SWI 4` to print integer values. This is in preparation for next week's exercise where you'll modify this program to generate all verses of the song (where each verse decreases the number of men printed, until you are left with a single, solitary meadow mower). Secondly, the verse contains some repeated phrases and the marking pipeline will check that your program doesn't have these defined multiple times.

Some hints for completing this exercise:

- You will need to define strings to contain the phrases needed to printout the verse. Remember you can print strings out more than once if you need to.
- Remember each string requires a label so you can refer to it
- Integer numbers (4, 3, 2 etc.) must be printed out (in decimal) using `SWI 4`.
- Remember that `SWI 3` and `SWI 4` both expect a value in `R0`. `SWI 3` expects `R0` to contain the address of the string to print, while `SWI 4` expects `R0` to containing the number to print in decimal. You will need to ensure that `R0` contains the expected value.
- Don't forget a `SWI 2` to stop your program at the end…

APPENDIX A

The original 'Hello World' program:

```
            B main

hello       DEFB  "Hello World\n",0
goodbye     DEFB  "Goodbye Universe\n",0

            ALIGN

main        ADR   R0, hello        ; put address of hello string in R0
            SWI   3                ; print it out
            ADR   R0, goodbye      ; put address of goodbye string in R0
            SWI   3
            SWI   2                ; stop
```

APPENDIX B

The Mowing Song:

```
4 men went to mow
Went to mow a meadow
4 men, 3 men, 2 men, 1 man and his dog, Spot
Went to mow a meadow
```