CPEN 211 Computer Systems I, 2024
Lab Proficiency Test #2

Question 1 [3 marks, part marks possible]: Create a file named "**q1.s**" and inside it write ARM assembly to implement the C code for function `func` below. Assume that `i`, `j`, and `k` are 32-bit signed integers stored in registers `R1`, `R2`, and `R3` respectively, and that the base of array A is in `R0`.

```
int func(int i, int j, int k, int *A) {
  int result = A[0] + A[1];
  A[0] = 42;
  if ( i & 1 )
    A[0] = 1;
  if ( j > k )
    A[1] = 2;
  else {
    A[2] = 3;
    if ((i < 0) && (k > 10))
      A[3] = -j;
  }
  return result;
}
```

The autograder requires the result returned by `func` be in `R0`. Your q1.s must contain the ARM code below where you must replace the comment "**`// ADD YOUR CODE HERE`**" with ARM code for `func`. Ensure the ARM code you add does not modify `R13` or `R14`. Your ARM code should work with any values of `i`, `j`, `k` and any array `A`, not just the values used in the ARM code below. To test your code, you may change the inputs to `func` by modifying the values in `R0` to `R3` by changing the lines before "`BL func`" and/or changing the array "`data`" in `q1.s`. Your solution for Question 1 will get zero if any of the following are true: (1) Your **last** "Lab Proficiency Test #2" attempt does not include "`q1.s`"; (2) Your "`q1.s`" file does not compile with the Monitor Program configured to use the DE1-SoC Computer or the online simulator: https://cpulator.01xz.net/?sys=arm-de1soc Ignore warnings about "Function clobbered registers(s)" in the online simulator. You may use a compiler such as Microsoft Visual Studio or GCC/GDB to compile and single step through the C code to understand its behavior, but you are NOT allowed use a compiler to generate ARM code or use a debugger to disassemble C code compiled for ARM.

```
.global _start
_start:
  MOV R0, #1    // i=1
  MOV R1, #1    // j=1
  MOV R2, #1    // k=1
  LDR R3, =data // set base of A = first address of array "data"
  BL func
END: B END  // infinite loop; R0 should contain return value of func

.global func
func:
  // ADD YOUR CODE HERE
  MOV PC, LR

data:
  .word 0, 0, 0, 0
```

CPEN 211 Computer Systems I, 2024
Lab Proficiency Test #2

Question 2 [2 marks, part marks possible]: Create a file named “**q2.s**” and inside it write ARM assembly to implement the C code for function `clunky` below. Assume that the base of array A is in `R0`.

```
int clunky(int *A) {
  int result = 0, next = 0;
  while (A[next] != -1) {
    int tmp = A[next];
    result += A[next+1];
    if ( (tmp/2) & 1 ) {
      A[next] = -1;
    } else {
      A[next] = -2;
    }
    next = tmp;
  }
  return result;
}
```

The autograder requires the result returned by `clunky` be in `R0` after your code executes. Your `q2.s` must contain the ARM code below. Replace the comment “**`// ADD YOUR CODE HERE`**” with your ARM code for `clunky`. Ensure the ARM code you add does not modify `R13` or `R14`. With the input arrays `A` given in the code below, after your code runs `R0` should contain 6, `A[0]` should contain -2, `A[1]` should contain 1, `A[2]` should contain -1, `A[3]` should contain 2, `A[4]` should contain -1, `A[5]` should contain 3, `A[6]` should contain -1, and `A[7]` should contain 4. If you wish to try additional tests you will first need to carefully study the C code above to figure out what it does before modifying the contents of array `A`. You may use a compiler such as Microsoft Visual Studio or GCC/GDB to compile and single step through the C code to understand its behavior, but you are NOT allowed use a compiler to generate ARM code or use a debugger to disassemble C code compiled for ARM. You may ignore “Function clobbered register(s)” warnings in the online simulator. Your **last** “Lab Proficiency Test #2” attempt must include both “`q1.s`” and “`q2.s`”.

```
.global clunky
clunky:
  // ADD YOUR CODE HERE
  MOV PC, LR

.global _start
_start:
  LDR R0,=A
  BL clunky
end: B end // infinite loop; R0 should contain return value of clunky

A: .word 4,1,6,2,2,3,-1,4
```