CS1021 Tutorial #5 Solution
More Pseudo-code and Flow Control

1 Translating Pseudo-code into ARM Assembly Language

Translate each of the following pseudo-code programs into ARM Assembly Language.

(a) ARM Assembly

```
    CMP R8, #100
    BLT endif
    CMP R9, #10
    BLT endif
    ADD R8, R8, R9
endif
```

(b) ARM Assembly

```
    CMP R2, #5
    BEQ if
    CMP R3, #15
    BNE endif
if  ADD R3, R3, #1
endif
```

(c) ARM Assembly

```
    CMP R3, #'a'
    BEQ ifvowel
    CMP R3, #'e'
    BEQ ifvowel
    CMP R3, #'i'
    BEQ ifvowel
    CMP R3, #'o'
    BEQ ifvowel
    CMP R3, 'u'
    BNE eifvowel
ifvowel
    ADD R1, R1,#1
eifvowel
```

(d) ARM Assembly

```
    CMP R0, #'a'
    BLO notlc
    CMP R0, #'z'
    BLS ifalpha
notlc
    CMP R0, #'A'
    BLO eifalpha
    CMP R0, #'Z'
    BHI eifalpha
ifalpha
    ADD R1, R1, #1
eifalpha
```

(e) ARM Assembly

```
    CMP R6, #'+'
    BNE elsifmns
    ADD R0, R7, R8
    B   endifop
elsifmns
    CMP R6, #'-'
    BNE elsifmul
    SUB R0, R7, R8
    B   endifop
elsifmul
    CMP R6, #'*'
    BNE elsop
    MUL R0, R7, R8
    B   endifop
elsop
    MOV R0, #0
endifop
```

2 Reading User Input

(a) Pseudo-code

```
value = 0;
ch = getchar();

// process character and read next character
// until RETURN key is pressed
while (ch != 0x0D)
{
    sendchar(ch);

    // splitting this deliberately for clarity!
    value = value * 10;
    value = value + (ch - '0');

    // this could be moved to the top of the
    // loop, avoiding the need for the first
    // getchar(), bit again I'm leaving it here
```

```
    // for clarity!
    ch = getchar();
}
```

(b) ARM Assembly

```
1     MOV R10, #10     ; multiplier = 10;
2     MOV R4, #0   ; value = 0
3     BL  getchar
4  whdigits            ; while (ch != 0x0D)
5     CMP R0, #0x0D    ; {
6     BEQ ewhdigits    ;
7     BL  sendchar     ;   sendchar(ch);
8     MUL R4, R10, R4  ;   value = value * 10;
9     SUB R0, R0, #'0'     ; ch = ch - '0';
10    ADD R4, R4, R0   ;   value = value + ch;
11    BL  getchar      ;   ch = getchar();
12    B   whdigits     ; }
13 ewhdigits
```