Name _***Answers***_

CE-1921-11 – Dr. Durant – Quiz 3
Spring 2017, Week 3

1. (7 points) Translate the following C/Java-like function into ARMv4 assembly. Use the standard ARM registers for arguments and return value

```
int sel(int x, int y) {
      x = x – y;
      if (x==0)
            return y;
      else
            return x;
}

sel:  subs r0,r0,r1     ; standard: x in r0, y in r1
                        ; s updates NVCZ, but could separately cmp/tst/etc.
      moveq r0,r1       ; == case, != case already has x in r0 for return
      mov pc,lr
```

2. (3 points) Write a main routine that calls your function with the arguments 17 and 33 and then hangs/spins on one instruction forever.

```
main: mov r0,#17
      mov r1,#33
      bl sel
end:  b end
```

Name _***Answers***_

CE-1921-21 – Dr. Durant – Quiz 3
Spring 2017, Week 3

1. (7 points) Translate the following C/Java-like function into ARMv4 assembly. Use the standard ARM registers for arguments and return value

```
int lpf(unsigned int x) {
      unsigned int result = 0;
      for (unsigned int i = 1; i <= 5; ++i) {
            result = result + i * (x+i);
      }
      return result;
}

lpf:  mov r1,#0   ; result
      mov r2,#1   ; i = 1
next: cmp r2,#5
      bhi done    ; hi is unsigned >
                  ; getting here means <= was true
      add r0,r0,#1; x+=1; happens i times; could also use separate
                  ; register, e.g., add r3,r0,r2 ; r3 is (x+i)
      mla r1,r2,r0,r1   ; result = i * (x+i) + result;
      add r2,r2,#1; ++i
      b next
done: mov r0,r1
      mov pc,lr

; below we take advantage of an algebraic simplification of the
; implemented formula
lpf2: rsb r0,r0,r0,asl #4     ; 15x = -x + 16x
      add r0,r0,#55           ; 15x+55 = unrolled and simplified version
      mov pc,lr
```

2. (3 points) Write a main routine that calls your function with the argument 17 and then hangs/spins on one instruction forever.

```
main: mov r0,#17
      bl lpf
end:  b end
```