a. (2 pts) Convert -45 to an 8-bit two’s complement number.

\[ +45 = 0 \times 2D \quad (2 \times 16 + 13 = 45) \]
\[ -45 = 0 - (+45) = 0x00 - 0x2D = 0xD3 \]

b. (2 pts) The value 0xEA is an 8-bit two’s complement number, give its decimal value.

MSbit of 0xEA is ‘1’, so this is a negative number, so we know the sign.
To get magnitude, we know that \( +N = 0 - (-N) \), so
\[ +N = 0x00 - 0xEA = 0x16 = +22, \] so this number 0xEA is –22 !

c. (3 pts) Write PIC18 assembly code to implement the following C code fragment.
Please note that \( k \) is a SIGNED INT !

```assembly
signed int k;
b = k >> 1;
bcf STATUS, C       ; assume we shift in a ‘0’
btfsc k+1, 7        ; test MSbit of MSbyte to check sign of k
bsf STATUS, C       ; negative, so shift in a ‘1’
rrcf k+1,f           ; right shift MSByte first
rrcf k,f             ; right shift LSByte next
```

d. (3 pts) Fill in the blanks below in converting the C code to assembly language.
Please note that \( j, k \) are SIGNED CHAR !!!!

```assembly
signed char j, k;
do{
    operation 1...
    operation 2...
}while(k >= j)
```

For \( k \geq j \), do \( k - j \).

If this is true, will get a positive number \((N=0, V=0)\) as subtracting a smaller number from larger number. If overflow occurs, will get a negative number \((N=1, V=1)\). Write flag branches to check both cases.