Xavier Kuehn 1, 2, 5 a-c, 6, 7

1. Write the following C function in assembly language.

float CircleArea(float radius);

```
CircleArea:
     VMUL.F32   S1,S0,S0
     VMUL.F32   S0,S1,S0
     BX         LR
```

2. Write the following C function in assembly language.

float DotProduct(float vec1[], float vec2[], int32_t len);

```
DotProduct:
     PUSH       {R4-R5,LR}           // reserve R4-R5
     LDR        R3,=0                // count variable
     VSUB.F32   S0,S0,S0             // set result to zero

     // beginning of loop
Top: CMP        R3,R2                // compare length and counter
     BGE        Done                 // if count ≥ length exit loop
     LDR        R4,[R0,R3,LSL 2]          // get current vec1
element
     VMOV       S1,R4                // move to FP register
     LDR        R5,[R0,R3,LSL 2]          // get current vec1
element
     VMOV       S2,R5                // move to FP register

     // S1 <- vec1[i], S2 <- vec2[i]
     VMLA.F32   S0,S1,S1             // S0 += vec1[i] x vec2[i]
     ADD        R3,R3,1              // increment counter
     B          Top                  // check loop condition

     // loop finished
Done:POP        {R4-R5,PC}           // return
```

5. Write assembly functions for the Taylor series in problem 4.

ReciprocalX(float x, float coef[], int32_t len);

```
ReciprocalX:    B    Polynomial
                BX   LR
```

SineX(float x, float coef[], int32_t len);

```
SineX:     B    Polynomial
           BX   LR
```

EtoX(float x, float coef[], int32_t len);

```
EtoX:B    Polynomial
     BX   LR
```

6. Write an assembly function to compute the arithmetic mean of floating point values.

float Mean(float x[], uint32_t n);

```
Mean:MOV            R2,0                // counter = 0
     VSUB.F32       S0,S0,S0            // result register = 0

Top: CMP            R2,R1               // counter < n ?
     BHS            Done                // exit loop if counter ≥ 
n
     LDR            R3,[R0,R2,LSL 2]         // get value
     VMOV           S1,R3               // move to FP register
     VADD.F32       S0,S0,S1            // S0 += S1
     ADD            R2,R2,1             // counter += 1
     B              Top                 // go to top of loop

Done:VMOV           S1,R1               // n -> FP register
     VCVT.F32.U32   S2,S1               // convert uint -> float
     VDIV.F32       S0,S0,S2            // sum / n = mean
     BX             LR
```

7. Write an assembly function to compute the variance of floating point values.

float Variance(float x[], uint32_t n, float mean);

```
Variance: MOV            R2,0           // counter = 0
          VSUB.F32       S1,S1,S1       // result register = 0

Top: CMP            R2,R1               // counter < n ?
     BHS            Done                // exit loop if counter ≥ n
     LDR            R3,[R0,R2,LSL 2]         // get value
     VMOV           S2,R3               // move to FP register
     VSUB.F32       S2,S2,S0            // (x - mean)
     VMUL.F32       S2,S2,S2            // (x - mean)^2
     VADD.F32       S1,S1,S2            // S1 += (x - mean)^2
     ADD            R2,R2,1             // counter += 1
     B              Top

Done:VMOV           S3,R1               // n -> FP register
     VCVT.F32.U32   S4,S3               // convert uint -> float
     VDIV.F32       S0,S1,S4            // sum / n = mean
     BX             LR
```