For each of the following problems, use a C #define or typedef to create a data type called Q32 capable of holding a fixed-point Q32.32 real.

```
typedef int64_t Q32 ;
```

1. Write a C function that prints a Q32 number as a real number without using any floating-point operations. Write a C program to test your function. The function prototype is:

   ```
   void PrintQ32(Q32 x) ;
   ```

   ```
   #define Upper32Bits(x) ((uint32_t *) &x)[1]

   void PrintQ32(Q32 x)
      {
      int k ;

      if (x < 0) { putchar('-') ; x = -x ; }
      printf("%u", Upper32Bits(x)) ;
      putchar('.') ;
      for (k = 0; k < 5; k++)
            {
            Upper32Bits(x) = 0 ;
            x = 10 * x ;
            printf(“%d”, Upper32Bits(x)) ;
            }
      }
   ```

   Copied from slide #32 of chapter 10 slides.

2. Create two Q32 values, 12.34 and -56.78 and write a C program to compute and display their sum and difference.

   ```
   // Functions defined elsewhere:
   Q32 Q32Ratio(int32_t top, int32_t btm) ;    // See slide 31
   void PrintQ32(Q32 x) ;                      // See slide 32

   int main()
      {
      Q32 x = Q32Ratio(1234, 100) ;
      Q32 y = Q32Ratio(-5678, 100) ;

      printf(“x + y = “) ; PrintQ32(x + y) ; printf(“\n”) ;
      printf(“x - y = “) ; PrintQ32(x - y) ; printf(“\n”) ;

      return 0 ;
      }
   ```

3. Write a C function to calculate the area of a circle using Q32 fixed-point reals and the multiply routine of Listing 10-1. Write a C program to test your function. The function prototype is:

```
Q32 CircleArea(Q32 radius) ;
```

Copied from slide #45 of chapter 10 slides.

```
// Functions defined elsewhere:
Q32 Q32Ratio(int32_t top, int32_t btm) ;    // See slide 31
Q32 Q32Product(Q32 a, Q32 b) ;              // See slide 42

Q32 CircleArea(Q32 radius)
      {
      Q32 pi = Q32Ratio(314159, 100000) ;
      Q32 rSquared = Q32Product(radius, radius) ;
      return Q32Product(pi, rSquared) ;
      }
```

5. Write a C function to evaluate a polynomial using Q32 fixed-point reals and the multiply routine of Listing 10-1. Write a C program to test your function. The function prototype is:

```
Q32 Polynomial(Q32 x, Q32 coef[], uint32_t terms) ;
```

```
// Functions defined elsewhere:
Q32 Q32Ratio(int32_t top, int32_t btm) ;    // See slide 31
Q32 Q32Product(Q32 a, Q32 b) ;              // See slide 42

Q32 Polynomial(Q32 x, Q32 coef[], uint32_t terms)
   {
   Q32 sum, power ;
   int k ;

   sum = 0 ;
   power = Q32Ratio(1, 1) ;
   for (k = 0; k < terms; k++)
      {
      sum += Q32Product(coef[k], power) ;
      power *= x ;
      power = Q32Product(power, x) ;
      }

   return sum ;
   }
```

Oops! My bad! The multiplication of "power" by "x" has to use Q32Product since both operands are Q32 fixed-point reals