Xavier Kuehn

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

```
typedef int32_t Q32;

#define FIXED(x) ((Q32)((4294967296 * (x) + 0.5)))
#define FLOAT(x) ((float)(x)/4294967296)

Q32 Q32Sum (Q32 A, Q32 B) {
	return (A + B);
}

int main () {
	// get fixed-point reals
	Q32 FPR1 = FIXED(12.34);
	Q32 FPR2 = FIXED(-56.78);

	// get sums and differences
	Q32 res1 = Q32Sum(FPR1, FPR2); // 12.43 - 56.78 = -44.44
	Q32 res2 = Q32Sum(FPR1, -FPR2); // 12.34 + 56.78 = 69.12
	Q32 res3 = Q32Sum(-FPR1, FPR2); // -12.34 + 56.78 = 44.44
	Q32 res4 = Q32Sum(-FPR1, -FPR2); // -12.34 - 56.78 = -69.12

	// convert results to floats
	f_res1 = FLOAT(res1);
	f_res2 = FLOAT(res2);
	f_res3 = FLOAT(res3);
	f_res4 = FLOAT(res4);
}
```

3. Write a C program to compute the dot product of two Q32 vectors.

```
Q32 DotProduct(Q32 vec1[], Q32 vec2[], uint32_t len) {
	if (len == 0) return 0;

	Q32 dp = 0;
	for (int i = 0; i < len; ++i) {
		dp += Q32Product(vec1[i], vec2[i]);
	}

	return dp;
}
```

5. Write C programs for the following Taylor series approximations using the Polynomial function from the previous question.

```
Q32 XReciprocal(Q32 x, Q32 coef[], uint32_t terms) {
     return Polynomial(x, coef, terms);
}

Q32 SineX(Q32 x, Q32 coef[], uint32_t terms) {
     return Polynomial(x, coef, terms);
}

Q32EtoX(Q32 x, Q32 coef[], uint32_t terms){
     return Polynomial(x, coef, terms);
}
```