17. **Solution:** (a) Trial division with all six mod-2 polynomials of degree 1 or 2 yields:

$$\begin{aligned} t^3+t+1 &= (t)(t^2+1)+1 \\ &= (t+1)(t^2+t)+1 \\ &= (t^2)(t)+t+1 \\ &= (t^2+1)(t)+1 \\ &= (t^2+t)(t+1)+1 \\ &= (t^2+t+1)(t+1)+t. \end{aligned}$$

All of these expressions have nonzero remainders, so t^3+t+1 is irreducible.

(b) The factorization $t^4+t^2+1 = (t^2+t+1)(t^2+t+1)+0 = (t^2+t+1)^2$ is discovered by trial division with the six mod-2 polynomials of degree 1 or 2. □

18. **Solution:** For all of these polynomials, we use the following:

Standard C Function: Test Mod-2 Polynomial Divisibility

```
unsigned int least_power (unsigned int mod2poly, int degree) {
  unsigned int rem, mask, highbit, N;
  if(mod2poly%2==0) return 0;  /* error: t divides mod2poly */
  highbit = 0x1<<(degree-1);   mask = highbit|(highbit-1);
  for(rem=mod2poly&mask, N=degree;  rem!=0x1;  ++N )
    if( rem & highbit ) rem = ((rem<<1)^mod2poly) & mask;
    else rem <<= 1;
  return N;
}
```

(a) Call `least_power()` with `mod2poly` set to 1011 (base 2) and `degree` set to 3. The return value shows that t^3+t+1 divides t^N+1, with $N = 7 = 2^3-1$, but no smaller $N > 0$.

(b) Mod-2 polynomial 1100000001111, of degree 12, divides t^N+1 for $N = 2047 = 2^{11}-1$, but no smaller $N > 0$.

(c) Mod-2 polynomial 11000000000000101, of degree 16, divides t^N+1 for $N = 32767 = 2^{15}-1$, but no smaller $N > 0$.

(d) Mod-2 polynomial 1100000000101000100000001, of degree 24, divides t^N+1 for $N = 7161$, but no smaller $N > 0$. □

19. **Solution:** On a computer that has integer types with 33 or more bits, we can use `least_power()` exactly as in Solution 18. Otherwise, on a computer with 32-bit integers, we simply remove the leftmost, most significant, 33rd bit, setting `mod2poly` to 00000100110000010001110110110111 (base 2), and