In-place FFT routines with permutations and temporary storage T.

$W = [1, \omega, \omega^2, ..., \omega^{n/2-1}, 1, \omega^2, \omega^4, ..., \omega^{n/2-2}, 1, \omega^4, \omega^8, ..., \omega^{n/2-4}, ..., 1, 0]$

```
void FFT1( int *A, int n,
       int *W, int p, int *T )
{
  int i,n2,t;
  if( n==1 ) return;
  n2 = n/2;
  for( i=0; i<n2; i++ ) {
     T[   i] = A[2*i];
     T[n2+i] = A[2*i+1];
  }
  FFT1( T,    n2, W+n2, p, A );
  FFT1( T+n2, n2, W+n2, p, A+n2 );
  for( i=0; i<n2; i++ ) {
     t = mulmod(W[i],T[n2+i],p);
     A[   i] = addmod(T[i],t,p);
     A[n2+i] = submod(T[i],t,p);
  }
  return;
}
```

```
void FFT2( int *A, int n,
       int *W, int p, int *T )
{
  int i,n2,t;
  if( n==1 ) return;
  n2 = n/2;
  for( i=0; i<n2; i++ ) {
     T[i] = addmod(A[i],A[n2+i],p);
     t = submod(A[i],A[n2+i],p);
     T[n2+i] = mulmod(t,W[i],p);
  }
  FFT2( T,    n2, W+n2, p, A );
  FFT2( T+n2, n2, W+n2, p, A+n2 );
  for( i=0; i<n2; i++ ) {
     A[ 2*i] = T[i];
     A[2*i+1] = T[n2+i];
  }
  return;
}
```