int main() {  
  
  // Initialize host variables ------------------------------------------------  
  
  const unsigned int n = 2048;  
  
  float\* A\_h = (float\*) malloc( sizeof(float)\*n );  
  for (unsigned int i=0; i < n; i++) { A\_h[i] = (rand()%100)/100.00; }  
  
  float\* B\_h = (float\*) malloc( sizeof(float)\*n );  
  for (unsigned int i=0; i < n; i++) { B\_h[i] = (rand()%100)/100.00; }  
    
  float\* C\_h = (float\*) malloc( sizeof(float)\*n );  
  
  // Allocate device variables ------------------------------------------------  
  
  float\* A\_d;  
  cudaMalloc((void\*\*) &A\_d, sizeof(float)\*n);  
  
  float\* B\_d;  
  cudaMalloc((void\*\*) &B\_d, sizeof(float)\*n);  
  
  float\* C\_d;  
  cudaMalloc((void\*\*) &C\_d, sizeof(float)\*n);  
  
  // Copy host variables to device --------------------------------------------  
  
  cudaMemcpy(A\_d, A\_h, sizeof(float)\*n, cudaMemcpyHostToDevice);  
  cudaMemcpy(B\_d, B\_h, sizeof(float)\*n, cudaMemcpyHostToDevice);  
  
  // Invoke kernel ------------------------------------------------------------  
  
  const unsigned int THREADS\_PER\_BLOCK = 512;  
  const unsigned int numBlocks = (n - 1)/THREADS\_PER\_BLOCK + 1;  
  dim3 gridDim(numBlocks, 1, 1), blockDim(THREADS\_PER\_BLOCK, 1, 1);  
  vecAddKernel<<< gridDim, blockDim >>> (A\_d, B\_d, C\_d, n);  
  
  // Copy device variables from host ------------------------------------------  
  
  cudaMemcpy(C\_h, C\_d, sizeof(float)\*n, cudaMemcpyDeviceToHost);  
  
  // Free memory --------------------------------------------------------------  
  
  free(A\_h);  
  free(B\_h);  
  free(C\_h);  
  cudaFree(A\_d);  
  cudaFree(B\_d);  
  cudaFree(C\_d);  
  
  return 0;  
  
}