```
//-------------------------------------------------------------
// Tony Hyun Kim
// CS 246, PS 3, Problem 2(c)
// Dense subgraph search
//-------------------------------------------------------------
#include <stdio.h>
#include <string.h>

int sum(int* v, int n)
{
  int acc = 0;
  for(int i=0; i<n; i++)
    acc += v[i];
  return acc;
}

float computeDensity(int *s, int *deg, int n) {
  return (0.5f*sum(deg,n))/sum(s,n);
}

int streamFile(int *s, int *deg, FILE* ifp)
{
  // Rewind file pointer to beginning
  rewind(ifp);
  int count = 0;
  int n1, n2;
  while(1)
  {
    count++;
    // Read a single edge
    fscanf(ifp,"%d %d",&n1,&n2);
    if(feof(ifp))
      return count;
    // Add edge to degree vector if it contained entirely within S
    if(s[n1]&&s[n2]) {
      deg[n1]++;
      deg[n2]++;
    }
  }
}

int main(int argc, char** argv)
{
  // Parameterize the problem
  char* source;
  int n;

//  source = "livejournal-undirected-small.txt"; n = 50
  source = "livejournal-undirected.txt"; n = 500000;
  float eps = 0.05; // Algorithm parameter

  printf("Input source: %s\n",source);
  printf("Expected node count: %d\n",n);
  printf("Algorithm epsilon: %.3f\n",eps);

  // No need to touch anything below this line
  //-------------------------------------------------------------
  FILE* ifp = fopen(source,"r");

  // Break if input file could not be opened
  if(ifp == NULL)
  {
    printf("Could not open file %s!\n",source);
    return 1;
  }

  // Declare and initialize algorithm state
  int s[n];
  int st[n];
```

```
  int deg[n];
  int used[n]; // This is a flag that indicates that the variable has
               // been used in a previous community and should be skipped
  for(int i=0; i<n; i++)
    used[i] = 0;

  float rhoS, rhoSt; // Densities of S and Stilde

  // We now find K communities by repeated applications of the algorithm
  for(int K=0; K<20; K++)
  {
    //printf("K=%d community\n",K);
    // Initialize S and St. Do not include variable if it has occurred
    //  previously in a community
    for(int i=0; i<n; i++) {
      s[i]   = used[i] ? 0 : 1;
      st[i]  = s[i];
      deg[i] = 0;
    }

    streamFile(s,deg,ifp);
    rhoS  = computeDensity(s,deg,n);
    rhoSt = rhoS;

    int iter = 0;
    while(sum(s,n)>0)
    {
      //printf("  Iter %2d, |S| %6d, |E(S)| %8d rho(S) %3.2f rho(St) %3.2f\n",
      //          iter++,sum(s,n),sum(deg,n)/2,rhoS,rhoSt);

      // Remove A(S) from S (see definition in assignment)
      for(int i=0; i<n; i++)
        if(deg[i] <= 2*(1+eps)*rhoS)
          s[i] = 0;

      // Stream through file, using the new definition of S
      memset(deg,0,sizeof(int)*n);
      streamFile(s,deg,ifp);
      rhoS = computeDensity(s,deg,n);

      if(rhoS > rhoSt)
      {
        rhoSt = rhoS;
        memcpy(st,s,sizeof(int)*n);
      }
    }

    // St holds our community
    for(int i=0; i<n; i++)
      if(st[i])
        used[i] = 1;
    int stLen = sum(st,n);
    printf("K %2d rho(St) %3.2f |St| %4d |E(St)| %6d\n",K,rhoSt,stLen,(int)(2*rhoSt*st
Len));
  }

  fclose(ifp);

  // Optional: We now write the output Stilde
  FILE* ofp = fopen("Stilde.txt","w");
  for(int i=0; i<n; i++)
    if(st[i])
      fprintf(ofp,"%d\n",i);
  fclose(ofp);

  return 0;
}
```