```c
# include <stdlib.h>
# include <stdio.h>
# include <math.h>
# include <time.h>

# include "cordic.h"

/******************************************************************************/

double compute_angle ( int idx, int nIter )

/******************************************************************************/
/*
  Purpose:

    Angle Array in Binary Tree Representation

  Discussion:


  Licensing:

    This code is distributed under the GNU LGPL license.

  Modified:

    2012.04.16

  Author:

    Young Won Lim

  Parameters:

*/
{
  double angle = 0.0;
  int i, j;
  char s[32];

  for (i=0; i<nIter; i++) {
    j = 1 << i;
    if (idx & (1 << i)) {
      angle += atan( 1. / j );
      s[nIter-i-1] = '1';
    } else {
      angle -= atan( 1. / j );
      s[nIter-i-1] = '0';
    }
    printf("i=%d j=%d 1/j=%f atan(1/j)=%f \n", i, j, 1./j, atan(1./j)*180/3.1416);
  }
  s[nIter] = '\0';

  printf("%d %d %s ---> %f \n", nIter, idx, s, angle*180/3.1416);



  return angle;
}




int main (int argc, char * argv[]) {

  double pi = 3.141592653589793;
  double K = 1.646760258121;
  int nIter = 3;
  int nAngle = 1 << nIter;
```

```
  int i, j;
  double *A;
  double x, y, z;
  double delta = 2.*pi / nAngle;
  FILE *fp;
  double r;

  A = (double *) malloc((1<<20) * sizeof (double));

  fp = fopen("angle.dat", "w");

  for (i=0; i<6; ++i) {
    nIter = i;
    nAngle = 1 << nIter;

    for (j=0; j<nAngle; ++j) {
      A[j] = compute_angle(j, nIter);
      printf("A[%d] = %f \n", j, A[j]);
      fprintf(fp, "%f %f 0.0 0.5\n", A[j]*180/pi, 0.5*i);
    }

  }

  fclose(fp);



  for (i=0; i<20; i+=4) {
    for (j=0; j<4; ++j) {
      r = atan( 1. / (1 << (i+j)) ) / atan( 1. / (1 << i) ) * 100;
      printf("index = %d --> r = %f \n", i+j, r);
    }
  }




  return 0;

}
```