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

  if (argc > 1 ) {
    nIter = atoi(argv[1]);
    nAngle = 1 << nIter;
  }
  printf("nIter = %d \n", nIter);

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

  for (i=0; i<nAngle; ++i) {

    A[i] = compute_angle(i, nIter);
  }

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

  for (i=0; i<nAngle; i++) {
    printf("A[%d] = %f \n", i, A[i]);
    fprintf(fp, "0.0 0.0 %f %f %f 0.0 0.0 0.5\n", cos(A[i]), sin(A[i]), A[i], A[i]);
  }

  fclose(fp);

  for (i=0; i<nAngle; i++) {
    x = 1 / K;
    y = 0.0;
    z = A[i] + delta*2;
    printf("-----------------------------------------------\n");
    printf("xi=%f yi=%f zi=%f\n", x, y, z);

    cordic(&x, &y, &z, nIter);

    printf("xo=%f yo=%f zo=%f\n", x, y, z);

  }

  printf("delta=%f \n", delta);

  return 0;

}
```