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

/* ベクトルの入力 */
void input_vector3( double *b, int m, int n, FILE *fin );
/* 行列の領域確保 */
double **dmatrix(int nr1, int nr2, int nl1, int nl2);
/* 行列の領域解放 */
void free_dmatrix(double **a, int nr1, int nr2, int nl1, int nl2);
/* ベクトル領域の確保 */
double *dvector(int i, int j);
/* ベクトル領域の解放 */
void free_dvector(double *a, int i);
/* ニュートン補間 */
double newton_ip( double *x, double *y, int n, double xi );

int main(void)
{
  FILE *fin, *fout;
  double *x, *y, xi;   /* xi は補間点 */
  int n;

  printf("データの個数を入力してください--->");
  scanf("%d", &n);
  n -= 1;  /* データ数がnなので,n <- n-1 として添え字を 0,1,...,n とする */

  printf("補間点を入力してください--->");
  scanf("%lf", &xi);

  /* ベクトル領域の確保 */
  x = dvector(0,n); /* x[0...n] */
  y = dvector(0,n); /* y[0...n] */

  /* ファイルのオープン */
  if ( (fin = fopen( "input_lag.dat", "r")) == NULL )
  {
      printf("ファイルが見つかりません : input_lag.dat \n");
      exit(1);
  }
  if( (fout = fopen( "output_lag.dat", "w")) == NULL )
  {
    printf("ファイルが作成できません : output_lag.dat \n");
    exit(1);
   }

  input_vector3( x, 0, n, fin );    /* ベクトル x の入出力 */
  input_vector3( y, 0, n, fin );    /* ベクトル y の入出力 */

  printf("補間の結果は, P(%f)=%f\n", xi, newton_ip(x,y,n,xi) );

  /* グラフを描くために結果をファイルに出力 */
  for( xi = x[0]; xi <= x[n]; xi += 0.01 )
  {
    fprintf(fout, "%f \t %f\n", xi, newton_ip(x,y,n,xi) );
  }
  fclose(fin); fclose(fout);  /* ファイルのクローズ */

  /* 領域の解放 */
  free_dvector( x, 0 );  free_dvector( y, 0 );

  return 0;
}

/* ニュートン補間 */
/* 添字は 0,1,...,n と仮定 */
double newton_ip( double *x, double *y, int n, double xi )
{
  int i, j;
  double pn = 0.0, li, **a;

  a = dmatrix(0, n, 0, n);

  for( i = 0; i <= n; i++) a[i][0] = y[i] ;

  /* 差商の計算 */
  for( j = 1; j <= n; j++)
  {
    for( i = 0; i <= n-j; i++ )
    {
      a[i][j] = ( a[i+1][j-1] - a[i][j-1] ) / ( x[i+j]-x[i] );
    }
  }

  /* 補間の計算 */
  pn = y[0]; li = 1.0;
  for ( j = 1; j <= n; j++ )
  {
    li *= ( xi - x[j-1] );
    pn += a[0][j] * li ;
  }

  free_dmatrix( a, 0, n, 0, n );

  return pn;
}

/* b[m...n] の入力 */
void input_vector3(double *b, int m, int n, FILE *fin)
{
  int i;
  for (i = m; i <= n; i++)
  {
    fscanf(fin, "%lf", &b[i]);
  }
}

double **dmatrix(int nr1, int nr2, int nl1, int nl2)
{
  int i, nrow, ncol;
  double **a;

  nrow = nr2 - nr1 + 1; /* 行の数 */
  ncol = nl2 - nl1 + 1; /* 列の数 */

  /* 行の確保 */
  if ((a = malloc(nrow * sizeof(double *))) == NULL)
  {
    printf("メモリが確保できません (行列 a)\n");
    exit(1);
  }
  a = a - nr1; /* 行をずらす */

  /* 列の確保 */
  for (i = nr1; i <= nr2; i++)
    a[i] = malloc(ncol * sizeof(double));
  for (i = nr1; i <= nr2; i++)
    a[i] = a[i] - nl1; /* 列をずらす */

  return (a);
}

void free_dmatrix(double **a, int nr1, int nr2, int nl1, int nl2)
{
  int i;

  /* メモリの解放 */
  for (i = nr1; i <= nr2; i++)
    free((void *)(a[i] + nl1));
  free((void *)(a + nr1));
}

double *dvector(int i, int j) /* a[i]～a[j] の領域を確保 */
{
  double *a;

  if ((a = malloc(((j - i + 1) * sizeof(double)))) == NULL)
  {
    printf("メモリが確保できません (from dvector) \n");
    exit(1);
  }

  return (a - i);
}

void free_dvector(double *a, int i)
{
  free((void *)(a + i)); /* (void *) 型へのキャストが必要 */
}

/* ベクトル a[m...n] と b[m...n] の内積を計算する */
double inner_product(int m, int n, double *a, double *b)
{
  int i;
  double s = 0.0;

  for (i = m; i <= n; i++)
    s += a[i] * b[i];

  return s;
}
```