```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 );
/* 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);
/* ラグランジュ補間 */
double lagrange( double *x, double *y, int m, 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);

  /* ベクトルの領域確保 */
  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, lagrange(x,y,0,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, lagrange(x,y,0,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;
}

/* ラグランジュ補間 */
double lagrange( double *x, double *y, int m, int n, double xi )
{
  int i, k;
  double pn = 0.0, li;

  /* P_n(x) の計算 */
  for ( i = m; i <= n ; i++ )
  {
    li = 1.0;
    /* l_i(x) の計算 */
    for( k = m; k <= n; k++ )
    {
      if( k != i ) li *= (xi -x[k]) / (x[i]-x[k]);
    }
    pn += li * y[i];
  }

  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;
}
```