```c
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include "common.h"

int main(int argc, char** argv)
{
  int rank, size;
  init_app(argc, argv, &rank, &size);

  //check number of processors used is a power of two
  if(size % 2){
    if(rank==0){
      printf("please choose a number of processors that is a multiple of 2,
      exiting\n");
    }
    close_app();
    return 0;
  }

  //maximum size of vector - 2^14 for Exercise 4
  int maxN = pow(2,14);

  //initialize some variables
  double mysum = 0;
  double sum=0;
  int k=4;
  int nextN=pow(2,k)/size;

  //split whole vector in parts, create subvector on each MPI process
  int *ofs, *cols;
  splitVector(maxN, size, &cols, &ofs);
  Vector v = createVector(cols[rank]);

  //fill vector
  for (int i=0;i<cols[rank];++i){
    //vector filling is interleaved - allows printing the difference
    //at each 2^k rather than recalculating entire vector each time
    v->data[i] = 1.0/pow((double)(i*size+1+rank),2);
    mysum += v->data[i];

    //print out difference at every 2^K n
    if (i+1==nextN){
      MPI_Reduce (&mysum, &sum, 1, MPI_DOUBLE, MPI_SUM, 0, MPI_COMM_WORLD);
      if (rank == 0) {
        double pi = 4.0 * atan(1.0);
        printf("difference at i=2^%2i: %1.16f\n", k, pi*pi/6.0-sum);
      }
      k++;
      nextN=pow(2,k)/size;
    }
  }

  //house cleaning. probably not needed (other than MPI_Finalize, but if you are
  //going to cut and paste somebody else's code, you might as well go for broke.
  freeVector(v);
  free(cols);
```

```
57      free(ofs);
58      close_app();
59      return 0;
60    }
61
```