

```

//compilation instructions/examples:
//gcc -fopenmp point_epsilon_starter.c -o point_epsilon_starter
//sometimes you need to link against the math library with -lm:
//gcc -fopenmp point_epsilon_starter.c -lm -o point_epsilon_starter

//math library needed for the square root

#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <string.h>
#include <math.h>
#include "omp.h"

//N is 100000 for the submission. However, you may use a smaller value of testing/debugging.
#define N 100000
//Do not change the seed, or your answer will not be correct
#define SEED 72

struct pointData{
double x;
double y;
};

void generateDataset(struct pointData * data);

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

    //Read epsilon distance from command line
    if (argc!=2)
    {
        printf("\nIncorrect number of input parameters. Please input an epsilon distance.\n");
        return 0;
    }

    char inputEpsilon[20];
    strcpy(inputEpsilon,argv[1]);
    double epsilon=atof(inputEpsilon);

    //generate dataset:
    struct pointData * data;
    data=(struct pointData*)malloc(sizeof(struct pointData)*N);
    printf("\nSize of dataset (MiB): %f",(2.0*sizeof(double)*N*1.0)/(1024.0*1024.0));
    generateDataset(data);

    //change OpenMP settings:
    omp_set_num_threads(1);

    double tstart=omp_get_wtime();

    //Write your code here:
    //The data you need to use is stored in the variable "data",

```

```
//which is of type pointData
```

```
double tend=omp_get_wtime();
```

```
printf("\nTotal time (s): %f",tend-tstart);
```

```
free(data);
```

```
printf("\n");
```

```
return 0;
```

```
}
```

```
//Do not modify the dataset generator or you will get the wrong answer
```

```
void generateDataset(struct pointData * data)
```

```
{
```

```
    //seed RNG
```

```
    srand(SEED);
```

```
    for (unsigned int i=0; i<N; i++){
```

```
        data[i].x=1000.0*((double)(rand()) / RAND_MAX);
```

```
        data[i].y=1000.0*((double)(rand()) / RAND_MAX);
```

```
    }
```

```
}
```