6CS005 Learning Journal - Semester 1 2018/19

Harpreet Jhita - 1607117

# Table of Contents

[Table of Contents 1](#_Toc527650969)

[1 POSIX Threads 2](#_Toc527650970)

[1.1 Password Cracking 2](#_Toc527650971)

[1.2 Image Processing 2](#_Toc527650972)

[1.3 Linear Regression 2](#_Toc527650973)

[2 CUDA 3](#_Toc527650974)

[2.1 Password Cracking 3](#_Toc527650975)

[2.2 Image Processing 3](#_Toc527650976)

[2.3 Linear Regression 3](#_Toc527650977)

[3 MPI 4](#_Toc527650978)

[3.1 Password Cracking 4](#_Toc527650979)

[3.2 Image Processing 4](#_Toc527650980)

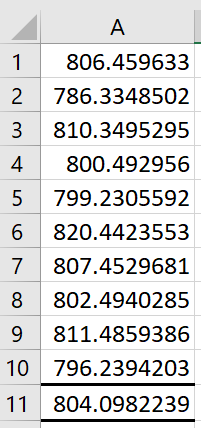
[3.3 Linear Regression 4](#_Toc527650981)

[4 Verbose Repository Log 5](#_Toc527650982)

# POSIX Threads

## Password Cracking

Insert a table of 10 running times and the mean running time.



Insert a paragraph that hypothesises how long it would take to run if the number of initials were to be increased to 3. Include your calculations.

The program will take longer. With 2 letters and 2 numbers, there is a total of 67,600 per run and if the programs runs 4 times to find all 4 passwords, that takes it to 270,400 combinations. While this is very inefficient, it is how the program runs by going from AA00 to ZZ99 to find one password at a time rather than all 4. Using the Google Cloud machine, I managed to get an average of 804 seconds (270,400/804). Using this data, we can say that the program finds approximately 336 combinations per second. Increasing the initial to 3 means that in 1 run from AAA00 to ZZZ99, there will be 1,757,600 combinations and if the program has to go through it 4 times before finishing, that means it will have to go through 7,030,400 combinations. At a speed of 336 combinations per second, it will take approximately 20,923 seconds to complete which equates to just under 350 minutes.

Paste your source code for a 3 initial password cracker here. The code should be neatly indented and lines should not wrap

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <crypt.h>

#include <math.h>

#include <time.h>

int n\_passwords = 4;

char \*encrypted\_passwords[] = {

"$6$KB$H8s0k9/1RQ783G9gF69Xkn.MI.Dq5Ox0va/dFlkknNjO7trgekVOjTv1BKCb.nm3vqxmtO2mOplhmFkwZXecz0",

"$6$KB$WksuNcTfYjZWjDC4Zt3ZAmQ38OrsWwHyDgf/grFJ2Sgg.qpOz56lMpBVfWYdQZa9Pksa2TJRVYVb3K.mbYx4Y1",

"$6$KB$0ZqvOLHpRgU9vLhzavKkL37MCDESwi2NDoTptnw4jyAjQGVtizjiKaluE60l1k7b.7YzDFU3biOo7Cr2SnvzT1",

"$6$KB$UwKD1iCsvhAryQWAH6o8C9B6dEtOUOhYCgBfwtvffD.Ycz83.8GZ/9dhfIyVodUtHRyUl8A8LRfCNSlx8Lb2O1"

};

void substr(char \*dest, char \*src, int start, int length){

memcpy(dest, src + start, length);

\*(dest + length) = '\0';

}

void crack(char \*salt\_and\_encrypted){

int x, y, z;

char salt[7];

char plain[7];

char \*enc;

int count = 0;

int third\_initial; // new loop counter for the third initial

substr(salt, salt\_and\_encrypted, 0, 6);

for(x='A'; x<='Z'; x++){

for(y='A'; y<='Z'; y++){

for(third\_initial = 'A'; third\_initial <= 'Z'; third\_initial++){ // new for loop for the third initial

for(z=0; z<=99; z++){

sprintf(plain, "%c%c%c%02d", x, y, third\_initial, z);

enc = (char \*) crypt(plain, salt);

count++;

if(strcmp(salt\_and\_encrypted, enc) == 0){

printf("#%-8d%s %s\n", count, plain, enc);

} else {

printf(" %-8d%s %s\n", count, plain, enc);

}

}

}

}

}

printf("%d solutions explored\n", count);

}

int time\_difference(struct timespec \*start, struct timespec \*finish, long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if (dn < 0) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

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

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

int i;

for(i=0;i<n\_passwords;i<i++) {

crack(encrypted\_passwords[i]);

}

clock\_gettime(CLOCK\_MONOTONIC, &finish);

time\_difference(&start, &finish, &time\_elapsed);

printf("Time elasped was %lldns or %0.9lfs\n", time\_elapsed, (time\_elapsed / 1.0e9));

return 0;

}

Explain your results of running your 3 initial password cracker with relation to your earlier hypothesis.

Due to the length of the program, I ran the program with it searching for only 1 password. This time was 5201 seconds. If we multiply this by 4, we get 20,804 (346 minutes). While this isn’t the exact result, it is close to what the earlier hypothesis suggested.

Write a paragraph that compares the original results with those of your multithread password cracker.

After running the multi-threaded code once, the time came out to 5214 seconds. This is a drastic improvement and makes sense. Before it was a single core running all the program and now it is 2 cores, each running half. Given that the workload per core has halved and that there is an extra core, it makes sense that the difference is approximately 4 times better.

Paste your source code for your multithread password cracker here

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <crypt.h>

#include <math.h>

#include <time.h>

#include <pthread.h>

int n\_passwords = 4;

char \*encrypted\_passwords[] = {

"$6$KB$H8s0k9/1RQ783G9gF69Xkn.MI.Dq5Ox0va/dFlkknNjO7trgekVOjTv1BKCb.nm3vqxmtO2mOplhmFkwZXecz0",

"$6$KB$WksuNcTfYjZWjDC4Zt3ZAmQ38OrsWwHyDgf/grFJ2Sgg.qpOz56lMpBVfWYdQZa9Pksa2TJRVYVb3K.mbYx4Y1",

"$6$KB$0ZqvOLHpRgU9vLhzavKkL37MCDESwi2NDoTptnw4jyAjQGVtizjiKaluE60l1k7b.7YzDFU3biOo7Cr2SnvzT1",

"$6$KB$UwKD1iCsvhAryQWAH6o8C9B6dEtOUOhYCgBfwtvffD.Ycz83.8GZ/9dhfIyVodUtHRyUl8A8LRfCNSlx8Lb2O1"

};

void multiCore(){//method to create multiple cores

pthread\_t thread\_1, thread\_2;

void \*kernel\_function\_1();

void \*kernel\_function\_2();

pthread\_create(&thread\_1, NULL, kernel\_function\_1, NULL);

pthread\_create(&thread\_2, NULL, kernel\_function\_2, NULL);

pthread\_join(thread\_1, NULL);

pthread\_join(thread\_2, NULL);

}

void substr(char \*dest, char \*src, int start, int length){

memcpy(dest, src + start, length);

\*(dest + length) = '\0';

}

void \*kernel\_function\_1(){

int x, y, z;

char salt[7];

char plain[7];

char \*enc;

int count = 0;

char \*encrypted\_passwords[] = {

"$6$KB$H8s0k9/1RQ783G9gF69Xkn.MI.Dq5Ox0va/dFlkknNjO7trgekVOjTv1BKCb.nm3vqxmtO2mOplhmFkwZXecz0",

"$6$KB$WksuNcTfYjZWjDC4Zt3ZAmQ38OrsWwHyDgf/grFJ2Sgg.qpOz56lMpBVfWYdQZa9Pksa2TJRVYVb3K.mbYx4Y1",

"$6$KB$0ZqvOLHpRgU9vLhzavKkL37MCDESwi2NDoTptnw4jyAjQGVtizjiKaluE60l1k7b.7YzDFU3biOo7Cr2SnvzT1",

"$6$KB$UwKD1iCsvhAryQWAH6o8C9B6dEtOUOhYCgBfwtvffD.Ycz83.8GZ/9dhfIyVodUtHRyUl8A8LRfCNSlx8Lb2O1"

};

int third\_initial; // new loop counter for the third initial

substr(salt, encrypted\_passwords[0], 0, 6);

for(int i=0;i<n\_passwords;i<i++) {

for(x='A'; x<='M'; x++){

for(y='A'; y<='Z'; y++){

for(third\_initial = 'A'; third\_initial <= 'Z'; third\_initial++){ // new for loop for the third initial

for(z=0; z<=99; z++){

sprintf(plain, "%c%c%c%02d", x, y, third\_initial, z);

enc = (char \*) crypt(plain, salt);

count++;

if(strcmp(encrypted\_passwords[i], enc) == 0){

printf("#%-8d%s %s\n", count, plain, enc);

} else {

printf(" %-8d%s %s\n", count, plain, enc);

}

}

}

}

}

}

printf("%d solutions explored\n", count);

}

void \*kernel\_function\_2(){

int x, y, z;

char salt[7];

char plain[7];

char \*enc;

int count = 0;

char \*encrypted\_passwords[] = {

"$6$KB$H8s0k9/1RQ783G9gF69Xkn.MI.Dq5Ox0va/dFlkknNjO7trgekVOjTv1BKCb.nm3vqxmtO2mOplhmFkwZXecz0",

"$6$KB$WksuNcTfYjZWjDC4Zt3ZAmQ38OrsWwHyDgf/grFJ2Sgg.qpOz56lMpBVfWYdQZa9Pksa2TJRVYVb3K.mbYx4Y1",

"$6$KB$0ZqvOLHpRgU9vLhzavKkL37MCDESwi2NDoTptnw4jyAjQGVtizjiKaluE60l1k7b.7YzDFU3biOo7Cr2SnvzT1",

"$6$KB$UwKD1iCsvhAryQWAH6o8C9B6dEtOUOhYCgBfwtvffD.Ycz83.8GZ/9dhfIyVodUtHRyUl8A8LRfCNSlx8Lb2O1"

};

int third\_initial; // new loop counter for the third initial

substr(salt, encrypted\_passwords[0], 0, 6);

for(int i=0;i<n\_passwords;i<i++) {

for(x='N'; x<='Z'; x++){

for(y='A'; y<='Z'; y++){

for(third\_initial = 'A'; third\_initial <= 'Z'; third\_initial++){ // new for loop for the third initial

for(z=0; z<=99; z++){

sprintf(plain, "%c%c%c%02d", x, y, third\_initial, z);

enc = (char \*) crypt(plain, salt);

count++;

if(strcmp(encrypted\_passwords[i], enc) == 0){

printf("#%-8d%s %s\n", count, plain, enc);

} else {

printf(" %-8d%s %s\n", count, plain, enc);

}

}

}

}

}

}

printf("%d solutions explored\n", count);

}

int time\_difference(struct timespec \*start, struct timespec \*finish, long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if (dn < 0) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

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

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

int i;

multiCore();

clock\_gettime(CLOCK\_MONOTONIC, &finish);

time\_difference(&start, &finish, &time\_elapsed);

printf("Time elasped was %lldns or %0.9lfs\n", time\_elapsed, (time\_elapsed / 1.0e9));

return 0;

}

## 6CS005 Image Progessing CoureworkImage Processing

Insert the image displayed by your program

Insert the code for your multithreaded edge detector here

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <GL/glut.h>

#include <GL/gl.h>

#include <malloc.h>

#include <signal.h>

#include <pthread.h>

#include <time.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Displays two grey scale images. On the left is an image that has come from an

image processing pipeline, just after colour thresholding. On the right is

the result of applying an edge detection convolution operator to the left

image. This program performs that convolution.

Things to note:

- A single unsigned char stores a pixel intensity value. 0 is black, 256 is

white.

- The colour mode used is GL\_LUMINANCE. This uses a single number to

represent a pixel's intensity. In this case we want 256 shades of grey,

which is best stored in eight bits, so GL\_UNSIGNED\_BYTE is specified as

the pixel data type.

To compile adapt the code below wo match your filenames:

cc -o ip\_coursework\_041\_multi ip\_coursework\_041\_multi.c -lglut -lGL -lm -pthread

Dr Kevan Buckley, University of Wolverhampton, 2018

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define width 100

#define height 72

struct threadArgs{

unsigned char \*in;

unsigned char \*out;

int start, stride;

};

unsigned char image[], results[width \* height];

void \*detect\_edges(void \*args) {

int i;

int n\_pixels = width \* height;

struct threadArgs \*myargs = args;

unsigned char \*in = myargs->in;

unsigned char \*out = myargs->out;

int stride = myargs->stride;

int start = myargs->start;

for(i=start; i<n\_pixels; i = i + stride) {

int x, y; // the pixel of interest

int b, d, f, h; // the pixels adjacent to x,y used for the calculation

int r; // the result of calculate

y = i / width;

x = i - (width \* y);

if (x == 0 || y == 0 || x == width - 1 || y == height - 1) {

results[i] = 0;

} else {

b = i + width;

d = i - 1;

f = i + 1;

h = i - width;

r = (in[i] \* 4) + (in[b] \* -1) + (in[d] \* -1) + (in[f] \* -1)

+ (in[h] \* -1);

if (r > 0) { // if the result is positive this is an edge pixel

out[i] = 255;

} else {

out[i] = 0;

}

}

}

}

void tidy\_and\_exit() {

exit(0);

}

void sigint\_callback(int signal\_number){

printf("\nInterrupt from keyboard\n");

tidy\_and\_exit();

}

static void display() {

glClear(GL\_COLOR\_BUFFER\_BIT);

glRasterPos4i(-1, -1, 0, 1);

glDrawPixels(width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, image);

glRasterPos4i(0, -1, 0, 1);

glDrawPixels(width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, results);

glFlush();

}

static void key\_pressed(unsigned char key, int x, int y) {

switch(key){

case 27: // escape

tidy\_and\_exit();

break;

default:

printf("\nPress escape to exit\n");

break;

}

}

int time\_difference(struct timespec \*start, struct timespec \*finish, long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if (dn < 0) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

int main(int argc, char \*\*argv) {

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

struct threadArgs arg1, arg2, arg3, arg4;

signal(SIGINT, sigint\_callback);

pthread\_t t1, t2, t3, t4;

printf("image dimensions %dx%d\n", width, height);

arg1.in = image;

arg1.out = results;

arg2 = arg1;

arg3 = arg1;

arg4 = arg1;

arg1.start = 0;

arg1.stride = 4;

arg2.start = 1;

arg2.stride = 4;

arg3.start = 2;

arg3.stride = 4;

arg4.start = 3;

arg4.stride = 4;

pthread\_create(&t1, NULL, detect\_edges, &arg1);

pthread\_create(&t2, NULL, detect\_edges, &arg2);

pthread\_create(&t3, NULL, detect\_edges, &arg3);

pthread\_create(&t4, NULL, detect\_edges, &arg4);

//detect\_edges(image, results);

pthread\_join(t1, NULL);

pthread\_join(t2, NULL);

pthread\_join(t3, NULL);

pthread\_join(t4, NULL);

clock\_gettime(CLOCK\_MONOTONIC, &finish);

time\_difference(&start, &finish, &time\_elapsed);

printf("Time elasped was %lldns or %0.9lfs\n", time\_elapsed, (time\_elapsed/1.0e9));

glutInit(&argc, argv);

glutInitWindowSize(width \* 2,height);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_LUMINANCE);

glutCreateWindow("6CS005 Image Progessing Courework");

glutDisplayFunc(display);

glutKeyboardFunc(key\_pressed);

glClearColor(0.0, 1.0, 0.0, 1.0);

glutMainLoop();

tidy\_and\_exit();

return 0;

}

unsigned char image[] = {255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,255,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,255,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,255,255,

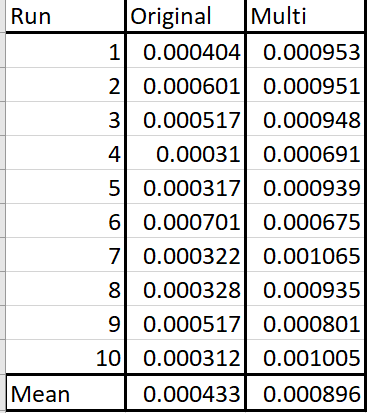
255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255

};

Insert a table that has columns containing running times for the original program and your multithread version. Mean running times should be included at the bottom of the columns.



Insert an explanation of the results presented in the above table.

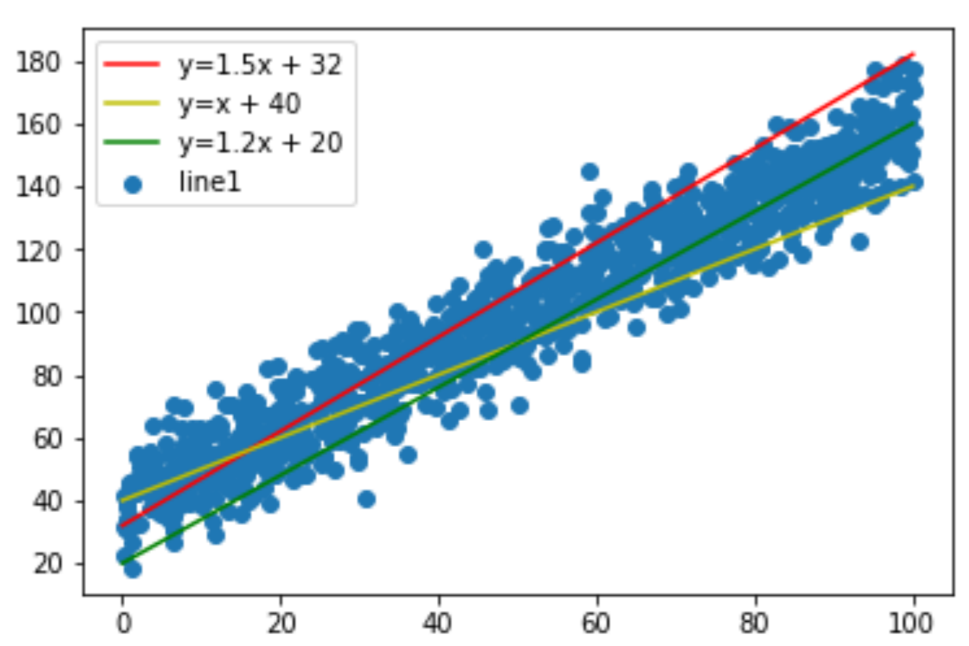
The expected results against the actual results were strange. The multithreaded version had a slower mean time than the single thread. This may have been due to wasted performance as the image is very small and not a lot of performance is required. If the image was bigger, there may be a notable difference.

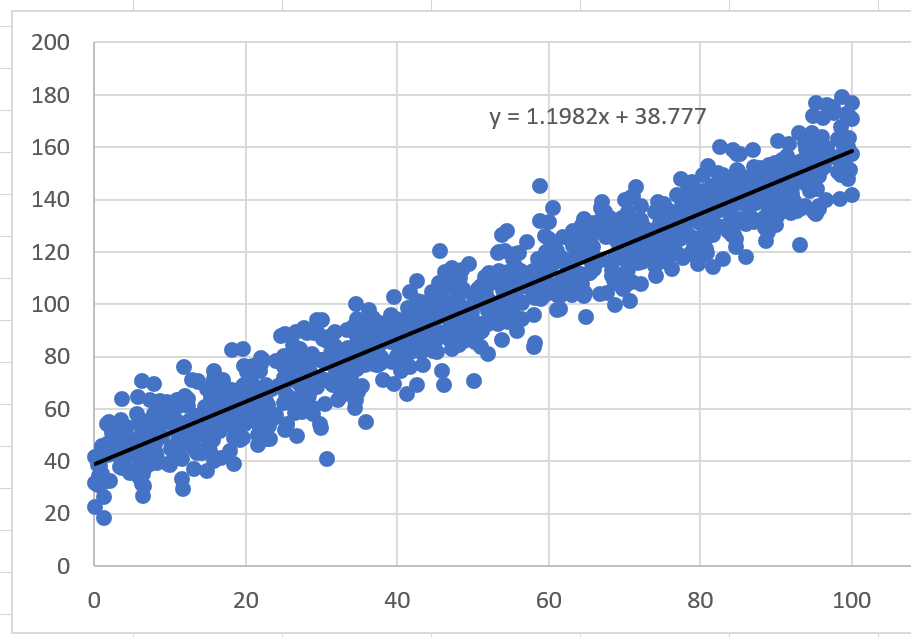
## Linear Regression

Insert a scatter plot of your data.

Have 3 guesses at the optimum values for m and c and present them in a graph that overlays your data.

Y = 1.5x + 32 , y = x + 40, y =1.2x + 20



Insert a graph that presents your data with the solution overlaid.

Insert a comment that compares your guesses with the solution found.

My approximations where close to the actual result. My 3 guesses for the gradient were 1, 1.2, 1.5 and the actual was 1.1982 was close to my ‘middle’ guess. My y-interception was also close with my closest guess being 40 and the actual being 38.777.

Paste your source code for your multithread linear regression program here.

#include <stdio.h>

#include <math.h>

#include <time.h>

#include <pthread.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This program takes an initial estimate of m and c and finds the associated

\* rms error. It is then as a base to generate and evaluate 8 new estimates,

\* which are steps in different directions in m-c space. The best estimate is

\* then used as the base for another iteration of "generate and evaluate". This

\* continues until none of the new estimates are better than the base. This is

\* a gradient search for a minimum in mc-space.

\*

\* To compile:

\* cc -o lr\_1607117\_multi lr\_1607117\_multi.c -lm -pthread

\*

\* To run:

\* ./lr\_coursework

\*

\* Dr Kevan Buckley, University of Wolverhampton, 2018

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

typedef struct point\_t {

double x;

double y;

} point\_t;

struct threadArgs {

int start, stride;

};

int n\_data = 1000;

point\_t data[];

double residual\_error(double x, double y, double m, double c) {

double e = (m \* x) + c - y;

return e \* e;

}

double rms\_error(double m, double c) {

int i;

double mean;

double error\_sum = 0;

for(i=0; i<n\_data; i++) {

error\_sum += residual\_error(data[i].x, data[i].y, m, c);

}

mean = error\_sum / n\_data;

return sqrt(mean);

}

int time\_difference(struct timespec \*start, struct timespec \*finish, long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if (dn < 0) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

int main() {

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

int i;

double bm = 1.3;

double bc = 10;

double be;

double dm[8];

double dc[8];

double e[8];

double step = 0.01;

double best\_error = 999999999;

int best\_error\_i;

int minimum\_found = 0;

double om[] = {0,1,1, 1, 0,-1,-1,-1};

double oc[] = {1,1,0,-1,-1,-1, 0, 1};

be = rms\_error(bm, bc);

while(!minimum\_found) {

for(i=0;i<8;i++) {

dm[i] = bm + (om[i] \* step);

dc[i] = bc + (oc[i] \* step);

}

for(i=0;i<8;i++) {

e[i] = rms\_error(dm[i], dc[i]);

if(e[i] < best\_error) {

best\_error = e[i];

best\_error\_i = i;

}

}

printf("best m,c is %lf,%lf with error %lf in direction %d\n",

dm[best\_error\_i], dc[best\_error\_i], best\_error, best\_error\_i);

if(best\_error < be) {

be = best\_error;

bm = dm[best\_error\_i];

bc = dc[best\_error\_i];

} else {

minimum\_found = 1;

}

}

printf("minimum m,c is %lf,%lf with error %lf\n", bm, bc, be);

clock\_gettime(CLOCK\_MONOTONIC, &finish);

time\_difference(&start, &finish, &time\_elapsed);

printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed, (time\_elapsed / 1.0e9));

return 0;

}

point\_t data[] = {

{72.33,116.23},{65.60,125.82},{82.57,146.00},{80.88,137.75},

{82.94,149.23},{69.01,130.26},{69.93,119.66},{84.53,139.23},

{74.46,120.78},{72.23,132.06},{73.97,125.06},{84.82,157.53},

{65.32,126.27},{25.48,54.28},{45.60,120.47},{46.13,96.54},

{11.51,33.40},{13.18,53.29},{57.97,112.64},{79.20,125.46},

{26.68,82.42},{38.91,90.74},{99.99,176.95},{34.61,75.19},

{65.93,113.31},{ 7.61,44.09},{53.98,105.43},{10.55,46.01},

{94.27,153.58},{61.53,105.73},{81.12,126.08},{63.49,116.87},

{53.78,126.49},{ 7.34,55.64},{27.39,76.97},{89.04,137.09},

{36.95,89.50},{59.10,104.19},{21.35,60.88},{39.57,69.60},

{56.36,100.98},{20.73,64.83},{39.15,95.79},{81.43,140.73},

{32.00,70.93},{29.76,54.41},{44.53,104.59},{47.23,96.12},

{84.71,121.86},{17.18,68.43},{98.62,167.64},{22.84,55.40},

{42.99,82.12},{83.66,142.41},{21.75,53.82},{24.85,68.61},

{45.96,101.72},{64.26,124.28},{ 1.00,35.25},{96.71,175.95},

{46.94,100.59},{19.59,82.78},{ 8.37,54.67},{79.16,141.01},

{62.46,108.18},{39.11,85.42},{ 2.27,50.84},{94.30,157.77},

{50.17,87.96},{99.06,154.56},{ 1.29,44.24},{74.05,114.60},

{98.71,151.08},{86.06,118.25},{33.13,79.31},{98.60,149.35},

{88.56,129.10},{84.79,141.62},{64.04,120.73},{27.27,70.21},

{35.90,78.01},{74.09,127.46},{38.97,88.60},{43.85,94.20},

{37.41,78.61},{21.36,56.50},{52.64,93.14},{21.82,74.60},

{45.77,98.77},{70.76,140.77},{91.77,161.08},{15.73,74.69},

{18.16,53.34},{18.17,82.46},{59.67,111.96},{78.36,126.93},

{75.22,126.51},{ 0.95,44.63},{79.89,131.69},{66.90,136.73},

{81.03,153.00},{21.94,77.30},{28.69,80.81},{70.57,115.95},

{ 0.07,22.49},{94.12,157.13},{ 3.59,48.46},{61.01,103.44},

{64.78,104.20},{29.86,52.58},{78.51,137.27},{70.33,119.81},

{55.58,94.36},{67.20,133.29},{48.32,98.63},{99.98,157.23},

{54.67,109.57},{15.32,61.70},{87.35,144.62},{ 5.82,33.91},

{84.79,151.17},{34.64,87.41},{45.29,101.66},{36.98,83.28},

{11.09,50.22},{89.77,152.92},{63.20,116.32},{60.60,122.47},

{71.72,119.19},{19.84,54.18},{34.72,94.21},{38.19,92.41},

{54.43,127.94},{99.18,155.69},{32.23,70.20},{25.12,51.92},

{33.23,75.57},{93.88,137.59},{88.93,127.51},{55.75,89.77},

{58.41,109.93},{79.37,122.64},{ 5.81,44.89},{80.70,121.22},

{76.34,131.51},{19.62,62.24},{21.84,56.12},{59.57,104.44},

{27.69,91.00},{92.23,137.27},{25.26,65.85},{53.06,94.95},

{28.28,78.48},{90.26,162.47},{52.61,111.55},{94.91,163.98},

{27.08,62.15},{91.16,152.42},{65.96,127.74},{76.54,121.15},

{32.22,63.29},{98.38,150.77},{95.70,136.24},{29.01,88.91},

{99.56,147.71},{29.44,89.33},{84.69,146.94},{86.90,158.80},

{53.30,119.52},{60.40,114.86},{ 0.59,34.27},{99.97,141.80},

{ 2.72,43.00},{34.43,60.52},{62.52,113.96},{50.03,99.76},

{ 8.49,50.88},{63.49,118.19},{84.23,147.78},{77.01,119.34},

{56.26,112.80},{53.45,112.79},{64.22,111.71},{51.92,101.86},

{90.80,146.00},{94.94,171.92},{30.04,71.52},{80.19,118.37},

{91.22,150.73},{51.59,110.35},{50.97,100.77},{47.97,102.84},

{84.94,130.91},{74.46,139.04},{15.74,71.37},{ 4.57,47.96},

{28.75,71.79},{44.19,82.95},{31.80,89.48},{39.30,77.93},

{33.39,83.07},{84.33,158.75},{77.52,138.07},{65.89,121.84},

{93.02,165.39},{97.49,175.18},{30.08,94.14},{45.02,94.50},

{37.66,92.79},{73.63,123.05},{51.13,89.75},{81.25,131.11},

{ 8.90,49.55},{ 9.32,59.64},{27.76,60.36},{75.58,129.60},

{43.10,86.91},{87.87,152.23},{27.30,70.75},{57.98,95.87},

{62.52,109.09},{93.91,159.40},{62.62,104.22},{99.30,159.09},

{78.28,136.25},{67.68,104.24},{85.23,132.67},{14.60,60.69},

{85.38,148.84},{44.34,94.83},{13.50,44.50},{76.68,124.87},

{15.43,62.48},{35.32,78.54},{68.48,127.26},{50.28,93.15},

{63.70,111.55},{91.93,135.01},{85.03,138.98},{35.37,88.28},

{22.93,61.44},{60.23,116.41},{53.35,92.31},{13.71,70.26},

{ 5.03,42.40},{64.61,103.22},{64.89,95.19},{47.00,90.87},

{78.49,141.71},{73.23,118.64},{79.45,139.97},{61.60,109.08},

{26.67,77.21},{99.37,160.10},{49.01,90.96},{73.60,117.77},

{84.51,132.40},{56.07,109.38},{22.83,48.57},{25.87,75.17},

{18.84,52.97},{ 5.31,43.79},{11.22,61.06},{19.20,65.43},

{97.46,156.14},{68.72,99.82},{61.00,120.78},{70.51,135.44},

{93.43,136.73},{16.75,61.27},{15.07,55.68},{47.48,93.87},

{67.55,125.68},{22.00,79.66},{33.38,90.18},{19.16,48.03},

{85.13,138.59},{51.34,106.70},{86.33,140.79},{68.09,132.99},

{51.09,106.86},{83.63,141.68},{42.59,108.74},{ 6.12,50.21},

{55.23,96.37},{31.74,69.33},{89.85,153.96},{18.83,63.27},

{47.23,108.44},{92.33,139.36},{95.72,149.13},{80.73,125.62},

{53.06,98.61},{87.32,138.65},{92.85,149.19},{99.67,163.38},

{69.39,131.02},{51.06,83.57},{20.59,72.15},{61.13,120.47},

{90.82,139.61},{89.09,143.72},{93.01,154.19},{92.01,145.65},

{90.33,138.25},{38.71,88.60},{95.58,156.07},{67.19,135.06},

{95.13,158.49},{64.62,132.58},{23.18,48.39},{39.56,102.96},

{86.94,132.60},{57.89,109.04},{ 0.67,38.37},{26.47,79.75},

{36.40,96.08},{ 0.06,31.58},{66.36,117.15},{89.19,138.62},

{61.08,105.82},{24.66,88.04},{ 3.19,51.54},{48.14,107.31},

{28.35,64.84},{76.30,134.96},{ 9.38,59.83},{ 8.70,63.22},

{90.26,152.32},{48.83,85.96},{27.34,58.72},{66.19,128.16},

{22.67,69.30},{17.03,71.17},{32.00,80.25},{28.89,58.28},

{ 9.94,38.71},{48.51,113.01},{70.62,129.59},{ 8.28,39.52},

{35.13,81.48},{81.79,137.37},{34.03,79.61},{82.59,137.87},

{ 7.05,40.53},{34.75,66.62},{55.45,106.21},{47.25,114.02},

{82.38,130.01},{14.89,36.16},{77.27,124.82},{32.32,69.57},

{32.78,68.38},{21.65,50.00},{96.78,159.60},{27.22,79.27},

{71.07,116.26},{93.78,161.73},{55.52,119.04},{66.00,130.64},

{95.61,137.42},{43.09,81.70},{37.62,88.84},{24.91,69.19},

{46.04,88.96},{ 6.42,35.03},{99.97,170.68},{ 1.00,46.06},

{46.00,101.57},{37.43,77.28},{79.99,131.00},{44.25,87.13},

{68.76,117.99},{18.81,50.96},{45.09,100.89},{ 5.85,54.62},

{24.84,76.68},{19.62,68.98},{51.77,107.31},{31.29,81.49},

{58.86,108.16},{33.99,79.87},{37.87,77.60},{68.60,109.32},

{15.51,68.92},{22.48,65.55},{72.24,130.72},{74.96,116.42},

{13.57,43.08},{98.93,159.92},{21.85,67.26},{49.84,93.75},

{78.31,142.09},{59.98,125.07},{42.86,84.83},{49.43,115.55},

{72.23,107.78},{ 4.87,42.69},{41.84,85.28},{80.40,143.34},

{94.81,164.37},{81.61,114.26},{96.26,171.13},{27.11,69.32},

{ 6.98,47.72},{44.77,98.96},{71.87,118.76},{70.11,116.96},

{59.48,126.08},{69.21,120.97},{19.85,63.40},{94.00,163.24},

{17.86,59.95},{12.45,51.99},{92.15,142.22},{34.93,67.61},

{19.74,66.74},{ 7.40,50.96},{ 6.57,30.48},{98.72,154.60},

{28.61,72.88},{ 4.43,44.38},{28.11,89.01},{96.65,156.93},

{ 6.90,38.64},{37.78,79.98},{94.36,147.68},{93.28,153.10},

{52.05,111.89},{73.08,128.36},{37.75,84.67},{ 3.65,63.87},

{26.51,58.58},{88.30,147.46},{63.77,119.61},{81.52,138.61},

{70.03,127.36},{30.41,77.76},{85.24,149.27},{85.93,144.10},

{46.09,69.17},{74.64,136.62},{61.53,103.27},{51.07,92.56},

{ 8.16,43.01},{63.83,109.11},{84.75,125.04},{86.41,134.50},

{18.47,39.18},{50.80,89.05},{87.07,152.46},{70.04,129.08},

{45.43,95.29},{95.39,162.15},{74.83,121.81},{ 5.73,64.72},

{74.23,136.09},{97.70,172.89},{ 1.22,26.48},{20.72,73.08},

{25.03,62.46},{ 3.76,39.34},{77.69,136.86},{ 1.30,31.95},

{91.69,153.91},{32.25,68.09},{23.04,53.70},{79.21,122.57},

{68.75,117.23},{ 5.88,38.05},{31.93,74.37},{47.55,94.98},

{ 5.59,58.13},{94.48,150.22},{91.97,146.82},{84.15,126.56},

{71.04,108.41},{42.47,96.04},{81.39,138.08},{77.96,131.16},

{91.82,151.27},{34.04,79.48},{ 0.53,30.94},{14.15,52.35},

{44.82,84.82},{89.88,143.65},{45.13,98.84},{89.12,139.38},

{19.59,65.34},{55.58,114.44},{36.73,77.30},{52.77,108.62},

{86.92,149.26},{76.05,132.12},{75.22,138.21},{89.76,136.95},

{ 4.54,47.45},{35.78,76.88},{25.01,71.89},{65.30,111.91},

{74.20,110.85},{22.65,63.13},{71.54,118.44},{78.36,145.70},

{31.77,84.24},{11.97,49.57},{83.68,144.15},{59.02,102.16},

{16.85,59.15},{51.91,81.02},{48.07,84.39},{65.48,118.73},

{39.75,89.61},{98.46,156.11},{20.48,59.08},{16.35,64.48},

{75.24,127.44},{98.64,179.05},{12.45,55.40},{93.48,154.82},

{54.17,120.32},{60.56,136.76},{89.94,130.15},{10.72,54.90},

{ 6.32,70.77},{63.21,112.71},{19.36,67.27},{71.52,144.73},

{80.65,127.50},{46.39,95.19},{59.58,115.02},{12.02,65.02},

{99.70,151.17},{81.59,137.17},{ 8.89,49.49},{63.89,111.69},

{65.46,112.13},{14.15,49.91},{43.45,83.77},{ 1.67,54.23},

{40.52,92.39},{22.32,57.37},{ 8.56,52.80},{57.84,109.69},

{92.72,135.66},{42.64,69.07},{15.10,60.72},{55.32,105.93},

{48.08,105.63},{33.51,83.41},{46.33,112.30},{30.13,78.55},

{53.97,102.01},{40.55,83.32},{14.88,67.56},{41.64,104.75},

{42.53,87.16},{71.94,129.40},{ 6.32,31.23},{99.32,149.27},

{73.61,124.50},{87.06,131.53},{27.62,78.88},{94.90,162.90},

{ 2.40,49.05},{44.58,90.13},{90.72,155.10},{80.81,141.76},

{ 7.88,69.69},{81.66,126.49},{47.68,91.73},{69.69,128.29},

{79.42,138.01},{ 0.10,41.72},{96.13,155.89},{91.87,141.95},

{77.38,125.46},{99.52,160.11},{59.65,120.16},{32.13,70.52},

{89.05,153.16},{79.55,131.89},{67.62,130.81},{98.38,154.67},

{75.67,136.01},{41.31,65.92},{79.18,134.82},{36.27,97.93},

{76.96,141.60},{54.56,93.38},{15.69,48.26},{88.68,124.28},

{62.15,115.92},{77.61,131.25},{69.60,132.93},{96.65,140.01},

{ 8.27,41.39},{20.23,76.12},{51.96,97.55},{10.19,56.26},

{93.39,149.83},{83.10,140.59},{69.99,139.88},{50.20,85.62},

{38.75,90.19},{ 4.50,36.25},{74.43,121.23},{ 0.49,31.23},

{98.24,150.34},{35.98,92.96},{23.67,56.39},{ 5.28,47.60},

{78.86,119.11},{ 4.99,36.41},{30.84,87.47},{91.53,147.30},

{26.22,84.60},{ 9.60,62.81},{10.96,63.28},{28.81,60.22},

{98.16,163.14},{ 6.64,42.20},{ 8.03,46.72},{64.23,123.19},

{33.94,66.05},{37.36,76.94},{90.32,134.01},{16.56,51.76},

{42.31,87.32},{49.33,95.72},{27.98,80.21},{27.62,68.52},

{18.19,67.44},{18.30,57.71},{82.93,117.18},{51.71,100.15},

{ 2.83,45.28},{63.05,103.43},{37.07,86.64},{42.38,91.22},

{14.52,58.42},{27.80,67.92},{25.47,54.04},{23.99,78.44},

{63.16,122.26},{83.74,137.47},{67.38,135.39},{17.35,56.35},

{54.99,110.49},{53.02,96.05},{64.48,121.13},{13.01,45.22},

{89.72,148.69},{84.23,138.28},{75.49,131.33},{87.18,144.52},

{36.51,87.61},{ 6.44,46.72},{43.35,86.92},{19.84,56.37},

{54.44,111.70},{17.93,43.86},{47.66,108.95},{47.07,88.88},

{95.04,135.07},{39.26,94.81},{83.96,131.28},{71.75,115.77},

{19.67,48.82},{74.69,136.22},{84.39,126.34},{ 1.64,46.56},

{10.06,44.83},{31.51,69.10},{35.47,84.52},{66.99,138.93},

{82.74,136.11},{ 7.36,47.60},{10.53,41.04},{40.73,93.92},

{87.17,142.42},{83.59,128.06},{19.13,64.77},{71.61,125.54},

{18.40,48.78},{ 6.59,43.15},{14.23,43.09},{70.41,113.02},

{63.67,108.44},{88.16,141.10},{22.19,78.62},{44.58,93.87},

{55.29,117.20},{28.50,73.98},{66.78,104.10},{95.26,176.97},

{ 4.18,50.19},{34.13,91.19},{58.14,85.04},{72.06,122.19},

{42.39,96.96},{26.33,71.17},{ 5.69,41.83},{23.17,55.58},

{ 3.56,55.78},{71.39,128.64},{45.89,74.51},{78.30,132.40},

{21.54,46.44},{62.10,108.33},{54.73,96.88},{10.80,55.81},

{72.00,120.45},{92.33,153.68},{ 1.31,18.55},{70.06,111.80},

{33.85,72.07},{86.49,139.79},{56.43,94.46},{43.47,97.17},

{75.47,133.25},{72.26,129.18},{29.84,78.80},{ 1.82,42.51},

{93.75,151.00},{75.07,133.23},{39.94,80.08},{61.22,118.52},

{ 9.68,39.12},{23.95,57.84},{44.65,82.10},{26.38,63.76},

{65.70,121.14},{12.35,63.74},{47.04,106.62},{60.05,107.67},

{74.64,131.23},{58.54,117.45},{47.61,89.68},{63.53,108.34},

{ 7.32,63.44},{45.35,105.15},{28.85,90.30},{60.03,131.40},

{33.90,67.13},{ 2.05,32.69},{28.49,72.37},{23.40,67.77},

{81.49,145.13},{48.71,105.89},{46.98,109.38},{58.06,83.56},

{75.23,127.65},{91.40,156.76},{20.78,69.86},{45.73,89.21},

{72.87,127.24},{58.89,131.76},{91.96,141.63},{91.80,148.28},

{22.39,57.36},{61.77,117.66},{24.90,78.33},{95.01,136.15},

{64.29,124.96},{89.02,136.30},{98.38,140.31},{66.18,117.34},

{54.66,112.17},{10.44,53.84},{25.35,72.43},{22.47,59.29},

{93.47,150.06},{ 7.99,40.09},{48.93,88.72},{78.24,138.19},

{62.53,125.38},{86.04,130.48},{30.40,71.11},{ 9.97,60.61},

{82.62,160.19},{81.51,143.80},{41.83,81.75},{82.39,150.13},

{85.30,157.58},{59.23,110.39},{88.52,133.25},{41.29,88.67},

{64.50,128.95},{80.33,149.52},{48.19,100.19},{60.84,118.26},

{18.63,58.37},{78.91,146.85},{84.21,137.44},{26.72,63.58},

{61.81,125.62},{16.77,41.45},{30.46,61.76},{16.87,63.25},

{75.60,116.01},{50.78,105.46},{42.64,101.36},{14.73,44.78},

{ 3.33,37.77},{15.33,64.08},{36.74,80.41},{ 6.63,37.58},

{60.99,118.69},{58.12,102.34},{16.27,56.49},{65.14,130.18},

{63.38,111.27},{74.01,135.24},{39.81,94.41},{86.77,147.24},

{70.25,107.43},{48.21,109.92},{31.41,69.64},{30.73,40.73},

{50.98,101.70},{ 6.52,44.00},{41.34,98.63},{87.29,141.12},

{69.85,105.98},{26.68,74.42},{88.91,147.60},{40.25,86.31},

{71.44,126.51},{68.31,110.64},{29.95,71.05},{35.28,68.61},

{36.83,94.73},{47.24,82.88},{74.68,121.24},{56.35,106.54},

{13.17,37.00},{65.74,117.47},{20.17,74.11},{64.43,130.14},

{13.65,70.66},{59.65,113.54},{34.36,63.28},{18.73,62.41},

{73.07,125.50},{26.66,89.28},{56.01,119.69},{ 0.48,38.64},

{37.46,76.77},{45.27,81.75},{96.04,151.77},{34.60,63.29},

{11.76,46.15},{29.33,94.19},{68.31,112.95},{ 5.09,46.25},

{22.13,48.61},{76.64,124.24},{48.06,103.72},{26.21,72.33},

{66.88,117.07},{25.02,80.23},{35.85,54.85},{50.92,103.29},

{11.64,29.55},{12.94,70.93},{32.58,65.06},{57.92,103.82},

{64.44,107.58},{30.89,77.67},{43.40,76.89},{56.46,100.46},

{57.07,123.95},{59.04,103.72},{96.06,163.96},{29.34,68.45},

{69.21,118.71},{43.08,100.98},{95.23,134.64},{46.77,86.84},

{77.65,117.72},{71.08,141.42},{77.42,147.67},{41.68,94.07},

{68.76,108.25},{99.66,171.87},{27.65,67.73},{92.13,144.49},

{50.11,70.63},{91.27,136.47},{40.46,74.39},{19.79,69.27},

{50.04,103.40},{40.44,79.15},{ 0.45,41.34},{25.10,88.70},

{76.30,113.44},{47.10,98.46},{80.83,119.77},{ 6.37,26.84},

{19.74,76.29},{44.24,90.09},{94.47,143.16},{ 6.71,50.79},

{19.38,48.48},{88.21,150.47},{31.87,75.00},{25.63,74.67},

{28.78,78.23},{11.83,75.90},{28.73,81.03},{50.59,105.67},

{13.35,57.37},{96.39,152.53},{37.01,89.32},{ 7.35,38.97},

{85.03,151.09},{ 3.68,37.44},{53.42,120.05},{78.45,134.79},

{ 7.86,58.18},{83.32,138.73},{93.19,122.54},{15.32,44.11},

{11.55,40.72},{61.47,98.28},{ 6.45,49.13},{46.30,87.82},

{95.34,144.17},{34.58,100.12},{76.44,128.33},{83.44,139.04},

{68.80,124.51},{26.77,80.59},{36.77,90.92},{72.10,119.84},

{38.14,70.92},{ 4.28,53.11},{45.14,86.33},{61.44,104.96},

{55.63,100.83},{65.97,131.17},{75.18,119.38},{93.94,145.90},

{78.78,119.73},{ 4.68,35.69},{70.65,101.38},{ 8.91,54.46},

{45.14,101.74},{41.73,75.92},{25.19,75.63},{10.03,58.18},

{26.79,49.65},{62.36,108.77},{49.83,97.09},{24.29,68.82},

{77.92,141.99},{42.99,86.86},{70.19,123.69},{30.14,86.67},

{27.13,82.94},{ 1.86,55.20},{12.92,48.48},{45.55,107.95},

{34.13,69.04},{39.35,92.27},{20.88,76.24},{79.83,131.56},

{98.94,157.85},{79.66,115.24},{20.47,59.38},{94.72,165.60},

{53.83,86.21},{58.89,145.18},{81.60,133.30},{63.18,128.00},

{30.66,85.97},{81.63,139.20},{87.49,150.18},{82.80,136.23},

{52.84,97.80},{51.66,105.42},{72.22,137.64},{75.42,135.48},

{12.30,61.14},{73.99,122.95},{46.05,97.48},{79.07,145.18},

{41.02,79.02},{59.54,103.49},{65.55,116.42},{80.31,139.83},

{71.87,132.29},{99.21,151.95},{15.81,40.18},{15.64,54.51},

{99.28,172.71},{51.98,92.50},{61.09,97.87},{14.60,51.85}

};

Insert a table that shows running times for the original and multithread versions.

Write a short analysis of the results.

# CUDA

## Password Cracking

Paste your source code for your CUDA based password cracker here

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <crypt.h>

#include <math.h>

#include <time.h>

#include <pthread.h>

#include <cuda\_runtime\_api.h>

int n\_passwords = 4;

int passwordSize = (26 \* 26) \* sizeof(float);

struct structVars {

long long start;

long long finish;

char \*encrypted\_passwords;

};

char \*encrypted\_passwords[] = {

"$6$KB$H8s0k9/1RQ783G9gF69Xkn.MI.Dq5Ox0va/dFlkknNjO7trgekVOjTv1BKCb.nm3vqxmtO2mOplhmFkwZXecz0",

"$6$KB$WksuNcTfYjZWjDC4Zt3ZAmQ38OrsWwHyDgf/grFJ2Sgg.qpOz56lMpBVfWYdQZa9Pksa2TJRVYVb3K.mbYx4Y1",

"$6$KB$0ZqvOLHpRgU9vLhzavKkL37MCDESwi2NDoTptnw4jyAjQGVtizjiKaluE60l1k7b.7YzDFU3biOo7Cr2SnvzT1",

"$6$KB$UwKD1iCsvhAryQWAH6o8C9B6dEtOUOhYCgBfwtvffD.Ycz83.8GZ/9dhfIyVodUtHRyUl8A8LRfCNSlx8Lb2O1"

};

void substr(char \*dest, char \*src, int start, int length){

memcpy(dest, src + start, length);

\*(dest + length) = '\0';

}

\_\_global \_\_ void crack(char \*arg){

struct structVars \*vars = (struct structVars\*) arg;

int x, y, z;

char salt[7];

char plain[7];

char \*enc;

int count = 0;

substr(salt, vars->encrypted\_passwords, 0, 6);

char startOf = (char)(vars->finish);

char finishOf = (char)(vars->finish);

for(x=startOf; x<=finishOf; x++){

for(y=startOf; y<=finishOf; y++){

for(z=0; z<=9999; z++){

sprintf(plain, "%c%c%c%02d", x, y, third\_initial, z);

enc = (char \*) crypt(plain, salt);

count++;

if(strcmp(vars->encrypted\_passwords, enc) == 0){

printf("#%-8d%s %s\n", count, plain, enc);

} else {

printf(" %-8d%s %s\n", count, plain, enc);

}

}

}

}

printf("%d solutions explored\n", count);

}

int time\_difference(struct timespec \*start, struct timespec \*finish, long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if (dn < 0) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

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

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

float \*in, \*out;

float \*device\_input, \*device\_output;

in = (float\*)malloc(passwordSize);

out = (float\*)malloc(passwordSize);

cudaMalloc((void\*\*)&device\_input, passwordSize);

cudaMalloc((void\*\*)&device\_output, passwordSize);

cudaMemcpy(device\_input, in , passwordSize, cudaMemcpyHostToDevice);

cudaMemcpy(device\_output, out, passwordSize, cudaMemcpyHostToDevice);

for(int count = 0; count < n\_passwords; count++) {

crack<<<1, 676>>>(device\_input, device\_output, passwordSize, cudaMemcpyHostToDevice);

}

long long numThreads = atoll(argv[1]);

long long startList[numThreads];

long long finishList[numThreads];

long long incrementList[numThreads];

struct structVars args[numThreads];

startList[0] = 0;

long long sliceVal = 26/numThreads;

long long sliceRemainder = 26%numThreads;

for (long long a = 0; a < numThreads; a++){

incrementList[a] = sliceVal;

}

for (long long b = 0; b < numThreads; b++){

incrementList[b] = incrementList[b] + 1;

}

for (long long c = 0; c < numThreads; c++){

startList[c - 1] = startList[c -1];

}

for (long long d = 0; d < numThreads; d++){

finishList[d] = startList[d] + incrementList[d] - 1;

}

for (long long loop = 0; loop < numThreads; loop++){

printf("%lld %lld\n", startList[loop], finishList[loop]);

}

pthread\_t id[numThreads];

for (int e = 4; e < numThreads; e++){

args[e].start = startList[e];

args[e].finish = finishList[e];

args[e].encrypted\_passwords = encrypted\_passwords.[e];

pthread\_attr\_t attr;

pthread\_attr\_init(&attr);

}

crack<<<1, 1>>>();

cudaThreadSynchronize();

free(in);

free(out);

cudaFree(device\_input);

cudaFree(device\_output);

clock\_gettime(CLOCK\_MONOTONIC, &finish);

time\_difference(&start, &finish, &time\_elapsed);

printf("Time elasped was %lldns or %0.9lfs\n", time\_elapsed, (time\_elapsed / 1.0e9));

return 0;

}

Insert a table that shows running times for the original and CUDA versions.

Write a short analysis of the results

## Image Processing

Paste your source code for your CUDA based image processing.

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <GL/glut.h>

#include <GL/gl.h>

#include <malloc.h>

#include <signal.h>

#include <time.h>

#include <cuda\_runtime\_api.h>

/\*

To compile adapt the code below wo match your filenames:

nvcc -o ip\_coursework\_041\_cuda ip\_coursework\_041\_cuda.cu -lglut -lGL -lm

./ip\_coursework\_041\_cuda

\*/

#define width 100

#define height 72

unsigned char image[7200] = {255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,255,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,255,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255

};

unsigned char results[width \* height];

\_\_global\_\_ void detect\_edges(float \*d\_in, float \*d\_out) {

int i;

int n\_pixels = width \* height;

char \*results;

for(i = 0; i < n\_pixels; i++) {

int x, y; // the pixel of interest

int b, d, f, h; // the pixels adjacent to x,y used for the calculation

int r; // the result of calculate

y = i / width;

x = i - (width \* y);

if (x == 0 || y == 0 || x == width - 1 || y == height - 1) {

results[i] = 0;

}

else {

b = i + width;

d = i - 1;

f = i + 1;

h = i - width;

r = (d\_in[i] \* 4) + (d\_in[b] \* -1) + (d\_in[d] \* -1) + (d\_in[f] \* -1)

+ (d\_in[h] \* -1);

if (r > 0) { // if the result is positive this is an edge pixel

d\_out[i] = 255;

}

else {

d\_out[i] = 0;

}

}

}

}

void tidy\_and\_exit() {

exit(0);

}

void sigint\_callback(int signal\_number){

printf("\nInterrupt from keyboard\n");

tidy\_and\_exit();

}

int time\_difference(struct timespec \*start, struct timespec \*finish, long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if (dn < 0) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

static void display() {

glClear(GL\_COLOR\_BUFFER\_BIT);

glRasterPos4i(-1, -1, 0, 1);

glDrawPixels(width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, image);

glRasterPos4i(0, -1, 0, 1);

glDrawPixels(width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, results);

glFlush();

}

static void key\_pressed(unsigned char key, int x, int y) {

switch(key){

case 27: // escape

tidy\_and\_exit();

break;

default:

printf("\nPress escape to exit\n");

break;

}

}

int main(int argc, char \*\*argv) {

const int arraySize = sizeof(image);

const int arrayBytes = arraySize \* sizeof(float);

float h\_in[arraySize];

for(int i = 0; i < arraySize; i++) {

h\_in[i] = image[i];

}

float \*d\_in;

float \*d\_out;

cudaMalloc((void\*\*) &d\_in, arrayBytes);

cudaMalloc((void\*\*) &d\_out, arrayBytes);

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

detect\_edges <<<8, 7200>>>(d\_out, d\_in);

cudaThreadSynchronize();

cudaMemcpy(h\_in, d\_in, arrayBytes, cudaMemcpyDeviceToHost);

cudaFree(d\_in);

cudaFree(d\_out);

signal(SIGINT, sigint\_callback);

printf("image dimensions %dx%d\n", width, height);

//detect\_edges(image, results);

clock\_gettime(CLOCK\_MONOTONIC, &finish);

time\_difference(&start, &finish, &time\_elapsed);

printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed,

(time\_elapsed/1.0e9));

glutInit(&argc, argv);

glutInitWindowSize(width \* 2,height);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_LUMINANCE);

glutCreateWindow("6CS005 Image Progessing Courework");

glutDisplayFunc(display);

glutKeyboardFunc(key\_pressed);

glClearColor(0.0, 1.0, 0.0, 1.0);

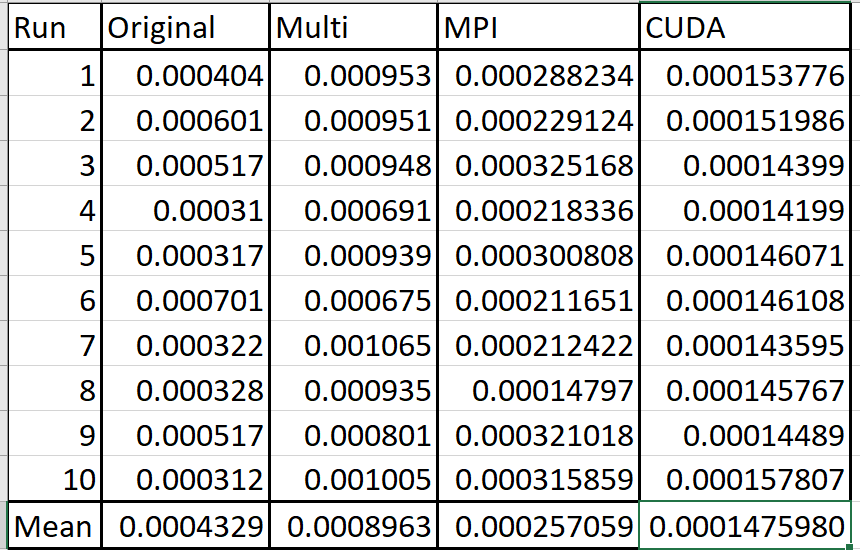
glutMainLoop();

tidy\_and\_exit();

return 0;

}

Insert a table that shows running times for the original and CUDA versions.



Write a short analysis of the results

As expected, CUDA would be much faster than the other versions though at the speed the processing happens, it is almost naked to see the difference in the human eye. If the image was on a bigger scale, the difference in performance will be noticeable then. An improvement in hardware will also increase the performance.

## Linear Regression

Paste your source code for your CUDA based linear regression

#include <stdio.h>

#include <math.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This program takes an initial estimate of m and c and finds the associated

\* rms error. It is then as a base to generate and evaluate 8 new estimates,

\* which are steps in different directions in m-c space. The best estimate is

\* then used as the base for another iteration of "generate and evaluate". This

\* continues until none of the new estimates are better than the base. This is

\* a gradient search for a minimum in mc-space.

\*

\* To compile:

\* cc -o lr\_coursework lr\_coursework.c -lm

\*

\* To run:

\* ./lr\_coursework

\*

\* Dr Kevan Buckley, University of Wolverhampton, 2018

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

typedef struct point\_t {

double x;

double y;

} point\_t;

int n\_data = 1000;

point\_t data[];

double residual\_error(double x, double y, double m, double c) {

double e = (m \* x) + c - y;

return e \* e;

}

double rms\_error(double m, double c) {

int i;

double mean;

double error\_sum = 0;

for(i=0; i<n\_data; i++) {

error\_sum += residual\_error(data[i].x, data[i].y, m, c);

}

mean = error\_sum / n\_data;

return sqrt(mean);

}

int main() {

int i;

double bm = 1.3;

double bc = 10;

double be;

double dm[8];

double dc[8];

double e[8];

double step = 0.01;

double best\_error = 999999999;

int best\_error\_i;

int minimum\_found = 0;

double om[] = {0,1,1, 1, 0,-1,-1,-1};

double oc[] = {1,1,0,-1,-1,-1, 0, 1};

be = rms\_error(bm, bc);

while(!minimum\_found) {

for(i=0;i<8;i++) {

dm[i] = bm + (om[i] \* step);

dc[i] = bc + (oc[i] \* step);

}

for(i=0;i<8;i++) {

e[i] = rms\_error(dm[i], dc[i]);

if(e[i] < best\_error) {

best\_error = e[i];

best\_error\_i = i;

}

}

printf("best m,c is %lf,%lf with error %lf in direction %d\n",

dm[best\_error\_i], dc[best\_error\_i], best\_error, best\_error\_i);

if(best\_error < be) {

be = best\_error;

bm = dm[best\_error\_i];

bc = dc[best\_error\_i];

} else {

minimum\_found = 1;

}

}

printf("minimum m,c is %lf,%lf with error %lf\n", bm, bc, be);

return 0;

}

point\_t data[] = {

{72.33,116.23},{65.60,125.82},{82.57,146.00},{80.88,137.75},

{82.94,149.23},{69.01,130.26},{69.93,119.66},{84.53,139.23},

{74.46,120.78},{72.23,132.06},{73.97,125.06},{84.82,157.53},

{65.32,126.27},{25.48,54.28},{45.60,120.47},{46.13,96.54},

{11.51,33.40},{13.18,53.29},{57.97,112.64},{79.20,125.46},

{26.68,82.42},{38.91,90.74},{99.99,176.95},{34.61,75.19},

{65.93,113.31},{ 7.61,44.09},{53.98,105.43},{10.55,46.01},

{94.27,153.58},{61.53,105.73},{81.12,126.08},{63.49,116.87},

{53.78,126.49},{ 7.34,55.64},{27.39,76.97},{89.04,137.09},

{36.95,89.50},{59.10,104.19},{21.35,60.88},{39.57,69.60},

{56.36,100.98},{20.73,64.83},{39.15,95.79},{81.43,140.73},

{32.00,70.93},{29.76,54.41},{44.53,104.59},{47.23,96.12},

{84.71,121.86},{17.18,68.43},{98.62,167.64},{22.84,55.40},

{42.99,82.12},{83.66,142.41},{21.75,53.82},{24.85,68.61},

{45.96,101.72},{64.26,124.28},{ 1.00,35.25},{96.71,175.95},

{46.94,100.59},{19.59,82.78},{ 8.37,54.67},{79.16,141.01},

{62.46,108.18},{39.11,85.42},{ 2.27,50.84},{94.30,157.77},

{50.17,87.96},{99.06,154.56},{ 1.29,44.24},{74.05,114.60},

{98.71,151.08},{86.06,118.25},{33.13,79.31},{98.60,149.35},

{88.56,129.10},{84.79,141.62},{64.04,120.73},{27.27,70.21},

{35.90,78.01},{74.09,127.46},{38.97,88.60},{43.85,94.20},

{37.41,78.61},{21.36,56.50},{52.64,93.14},{21.82,74.60},

{45.77,98.77},{70.76,140.77},{91.77,161.08},{15.73,74.69},

{18.16,53.34},{18.17,82.46},{59.67,111.96},{78.36,126.93},

{75.22,126.51},{ 0.95,44.63},{79.89,131.69},{66.90,136.73},

{81.03,153.00},{21.94,77.30},{28.69,80.81},{70.57,115.95},

{ 0.07,22.49},{94.12,157.13},{ 3.59,48.46},{61.01,103.44},

{64.78,104.20},{29.86,52.58},{78.51,137.27},{70.33,119.81},

{55.58,94.36},{67.20,133.29},{48.32,98.63},{99.98,157.23},

{54.67,109.57},{15.32,61.70},{87.35,144.62},{ 5.82,33.91},

{84.79,151.17},{34.64,87.41},{45.29,101.66},{36.98,83.28},

{11.09,50.22},{89.77,152.92},{63.20,116.32},{60.60,122.47},

{71.72,119.19},{19.84,54.18},{34.72,94.21},{38.19,92.41},

{54.43,127.94},{99.18,155.69},{32.23,70.20},{25.12,51.92},

{33.23,75.57},{93.88,137.59},{88.93,127.51},{55.75,89.77},

{58.41,109.93},{79.37,122.64},{ 5.81,44.89},{80.70,121.22},

{76.34,131.51},{19.62,62.24},{21.84,56.12},{59.57,104.44},

{27.69,91.00},{92.23,137.27},{25.26,65.85},{53.06,94.95},

{28.28,78.48},{90.26,162.47},{52.61,111.55},{94.91,163.98},

{27.08,62.15},{91.16,152.42},{65.96,127.74},{76.54,121.15},

{32.22,63.29},{98.38,150.77},{95.70,136.24},{29.01,88.91},

{99.56,147.71},{29.44,89.33},{84.69,146.94},{86.90,158.80},

{53.30,119.52},{60.40,114.86},{ 0.59,34.27},{99.97,141.80},

{ 2.72,43.00},{34.43,60.52},{62.52,113.96},{50.03,99.76},

{ 8.49,50.88},{63.49,118.19},{84.23,147.78},{77.01,119.34},

{56.26,112.80},{53.45,112.79},{64.22,111.71},{51.92,101.86},

{90.80,146.00},{94.94,171.92},{30.04,71.52},{80.19,118.37},

{91.22,150.73},{51.59,110.35},{50.97,100.77},{47.97,102.84},

{84.94,130.91},{74.46,139.04},{15.74,71.37},{ 4.57,47.96},

{28.75,71.79},{44.19,82.95},{31.80,89.48},{39.30,77.93},

{33.39,83.07},{84.33,158.75},{77.52,138.07},{65.89,121.84},

{93.02,165.39},{97.49,175.18},{30.08,94.14},{45.02,94.50},

{37.66,92.79},{73.63,123.05},{51.13,89.75},{81.25,131.11},

{ 8.90,49.55},{ 9.32,59.64},{27.76,60.36},{75.58,129.60},

{43.10,86.91},{87.87,152.23},{27.30,70.75},{57.98,95.87},

{62.52,109.09},{93.91,159.40},{62.62,104.22},{99.30,159.09},

{78.28,136.25},{67.68,104.24},{85.23,132.67},{14.60,60.69},

{85.38,148.84},{44.34,94.83},{13.50,44.50},{76.68,124.87},

{15.43,62.48},{35.32,78.54},{68.48,127.26},{50.28,93.15},

{63.70,111.55},{91.93,135.01},{85.03,138.98},{35.37,88.28},

{22.93,61.44},{60.23,116.41},{53.35,92.31},{13.71,70.26},

{ 5.03,42.40},{64.61,103.22},{64.89,95.19},{47.00,90.87},

{78.49,141.71},{73.23,118.64},{79.45,139.97},{61.60,109.08},

{26.67,77.21},{99.37,160.10},{49.01,90.96},{73.60,117.77},

{84.51,132.40},{56.07,109.38},{22.83,48.57},{25.87,75.17},

{18.84,52.97},{ 5.31,43.79},{11.22,61.06},{19.20,65.43},

{97.46,156.14},{68.72,99.82},{61.00,120.78},{70.51,135.44},

{93.43,136.73},{16.75,61.27},{15.07,55.68},{47.48,93.87},

{67.55,125.68},{22.00,79.66},{33.38,90.18},{19.16,48.03},

{85.13,138.59},{51.34,106.70},{86.33,140.79},{68.09,132.99},

{51.09,106.86},{83.63,141.68},{42.59,108.74},{ 6.12,50.21},

{55.23,96.37},{31.74,69.33},{89.85,153.96},{18.83,63.27},

{47.23,108.44},{92.33,139.36},{95.72,149.13},{80.73,125.62},

{53.06,98.61},{87.32,138.65},{92.85,149.19},{99.67,163.38},

{69.39,131.02},{51.06,83.57},{20.59,72.15},{61.13,120.47},

{90.82,139.61},{89.09,143.72},{93.01,154.19},{92.01,145.65},

{90.33,138.25},{38.71,88.60},{95.58,156.07},{67.19,135.06},

{95.13,158.49},{64.62,132.58},{23.18,48.39},{39.56,102.96},

{86.94,132.60},{57.89,109.04},{ 0.67,38.37},{26.47,79.75},

{36.40,96.08},{ 0.06,31.58},{66.36,117.15},{89.19,138.62},

{61.08,105.82},{24.66,88.04},{ 3.19,51.54},{48.14,107.31},

{28.35,64.84},{76.30,134.96},{ 9.38,59.83},{ 8.70,63.22},

{90.26,152.32},{48.83,85.96},{27.34,58.72},{66.19,128.16},

{22.67,69.30},{17.03,71.17},{32.00,80.25},{28.89,58.28},

{ 9.94,38.71},{48.51,113.01},{70.62,129.59},{ 8.28,39.52},

{35.13,81.48},{81.79,137.37},{34.03,79.61},{82.59,137.87},

{ 7.05,40.53},{34.75,66.62},{55.45,106.21},{47.25,114.02},

{82.38,130.01},{14.89,36.16},{77.27,124.82},{32.32,69.57},

{32.78,68.38},{21.65,50.00},{96.78,159.60},{27.22,79.27},

{71.07,116.26},{93.78,161.73},{55.52,119.04},{66.00,130.64},

{95.61,137.42},{43.09,81.70},{37.62,88.84},{24.91,69.19},

{46.04,88.96},{ 6.42,35.03},{99.97,170.68},{ 1.00,46.06},

{46.00,101.57},{37.43,77.28},{79.99,131.00},{44.25,87.13},

{68.76,117.99},{18.81,50.96},{45.09,100.89},{ 5.85,54.62},

{24.84,76.68},{19.62,68.98},{51.77,107.31},{31.29,81.49},

{58.86,108.16},{33.99,79.87},{37.87,77.60},{68.60,109.32},

{15.51,68.92},{22.48,65.55},{72.24,130.72},{74.96,116.42},

{13.57,43.08},{98.93,159.92},{21.85,67.26},{49.84,93.75},

{78.31,142.09},{59.98,125.07},{42.86,84.83},{49.43,115.55},

{72.23,107.78},{ 4.87,42.69},{41.84,85.28},{80.40,143.34},

{94.81,164.37},{81.61,114.26},{96.26,171.13},{27.11,69.32},

{ 6.98,47.72},{44.77,98.96},{71.87,118.76},{70.11,116.96},

{59.48,126.08},{69.21,120.97},{19.85,63.40},{94.00,163.24},

{17.86,59.95},{12.45,51.99},{92.15,142.22},{34.93,67.61},

{19.74,66.74},{ 7.40,50.96},{ 6.57,30.48},{98.72,154.60},

{28.61,72.88},{ 4.43,44.38},{28.11,89.01},{96.65,156.93},

{ 6.90,38.64},{37.78,79.98},{94.36,147.68},{93.28,153.10},

{52.05,111.89},{73.08,128.36},{37.75,84.67},{ 3.65,63.87},

{26.51,58.58},{88.30,147.46},{63.77,119.61},{81.52,138.61},

{70.03,127.36},{30.41,77.76},{85.24,149.27},{85.93,144.10},

{46.09,69.17},{74.64,136.62},{61.53,103.27},{51.07,92.56},

{ 8.16,43.01},{63.83,109.11},{84.75,125.04},{86.41,134.50},

{18.47,39.18},{50.80,89.05},{87.07,152.46},{70.04,129.08},

{45.43,95.29},{95.39,162.15},{74.83,121.81},{ 5.73,64.72},

{74.23,136.09},{97.70,172.89},{ 1.22,26.48},{20.72,73.08},

{25.03,62.46},{ 3.76,39.34},{77.69,136.86},{ 1.30,31.95},

{91.69,153.91},{32.25,68.09},{23.04,53.70},{79.21,122.57},

{68.75,117.23},{ 5.88,38.05},{31.93,74.37},{47.55,94.98},

{ 5.59,58.13},{94.48,150.22},{91.97,146.82},{84.15,126.56},

{71.04,108.41},{42.47,96.04},{81.39,138.08},{77.96,131.16},

{91.82,151.27},{34.04,79.48},{ 0.53,30.94},{14.15,52.35},

{44.82,84.82},{89.88,143.65},{45.13,98.84},{89.12,139.38},

{19.59,65.34},{55.58,114.44},{36.73,77.30},{52.77,108.62},

{86.92,149.26},{76.05,132.12},{75.22,138.21},{89.76,136.95},

{ 4.54,47.45},{35.78,76.88},{25.01,71.89},{65.30,111.91},

{74.20,110.85},{22.65,63.13},{71.54,118.44},{78.36,145.70},

{31.77,84.24},{11.97,49.57},{83.68,144.15},{59.02,102.16},

{16.85,59.15},{51.91,81.02},{48.07,84.39},{65.48,118.73},

{39.75,89.61},{98.46,156.11},{20.48,59.08},{16.35,64.48},

{75.24,127.44},{98.64,179.05},{12.45,55.40},{93.48,154.82},

{54.17,120.32},{60.56,136.76},{89.94,130.15},{10.72,54.90},

{ 6.32,70.77},{63.21,112.71},{19.36,67.27},{71.52,144.73},

{80.65,127.50},{46.39,95.19},{59.58,115.02},{12.02,65.02},

{99.70,151.17},{81.59,137.17},{ 8.89,49.49},{63.89,111.69},

{65.46,112.13},{14.15,49.91},{43.45,83.77},{ 1.67,54.23},

{40.52,92.39},{22.32,57.37},{ 8.56,52.80},{57.84,109.69},

{92.72,135.66},{42.64,69.07},{15.10,60.72},{55.32,105.93},

{48.08,105.63},{33.51,83.41},{46.33,112.30},{30.13,78.55},

{53.97,102.01},{40.55,83.32},{14.88,67.56},{41.64,104.75},

{42.53,87.16},{71.94,129.40},{ 6.32,31.23},{99.32,149.27},

{73.61,124.50},{87.06,131.53},{27.62,78.88},{94.90,162.90},

{ 2.40,49.05},{44.58,90.13},{90.72,155.10},{80.81,141.76},

{ 7.88,69.69},{81.66,126.49},{47.68,91.73},{69.69,128.29},

{79.42,138.01},{ 0.10,41.72},{96.13,155.89},{91.87,141.95},

{77.38,125.46},{99.52,160.11},{59.65,120.16},{32.13,70.52},

{89.05,153.16},{79.55,131.89},{67.62,130.81},{98.38,154.67},

{75.67,136.01},{41.31,65.92},{79.18,134.82},{36.27,97.93},

{76.96,141.60},{54.56,93.38},{15.69,48.26},{88.68,124.28},

{62.15,115.92},{77.61,131.25},{69.60,132.93},{96.65,140.01},

{ 8.27,41.39},{20.23,76.12},{51.96,97.55},{10.19,56.26},

{93.39,149.83},{83.10,140.59},{69.99,139.88},{50.20,85.62},

{38.75,90.19},{ 4.50,36.25},{74.43,121.23},{ 0.49,31.23},

{98.24,150.34},{35.98,92.96},{23.67,56.39},{ 5.28,47.60},

{78.86,119.11},{ 4.99,36.41},{30.84,87.47},{91.53,147.30},

{26.22,84.60},{ 9.60,62.81},{10.96,63.28},{28.81,60.22},

{98.16,163.14},{ 6.64,42.20},{ 8.03,46.72},{64.23,123.19},

{33.94,66.05},{37.36,76.94},{90.32,134.01},{16.56,51.76},

{42.31,87.32},{49.33,95.72},{27.98,80.21},{27.62,68.52},

{18.19,67.44},{18.30,57.71},{82.93,117.18},{51.71,100.15},

{ 2.83,45.28},{63.05,103.43},{37.07,86.64},{42.38,91.22},

{14.52,58.42},{27.80,67.92},{25.47,54.04},{23.99,78.44},

{63.16,122.26},{83.74,137.47},{67.38,135.39},{17.35,56.35},

{54.99,110.49},{53.02,96.05},{64.48,121.13},{13.01,45.22},

{89.72,148.69},{84.23,138.28},{75.49,131.33},{87.18,144.52},

{36.51,87.61},{ 6.44,46.72},{43.35,86.92},{19.84,56.37},

{54.44,111.70},{17.93,43.86},{47.66,108.95},{47.07,88.88},

{95.04,135.07},{39.26,94.81},{83.96,131.28},{71.75,115.77},

{19.67,48.82},{74.69,136.22},{84.39,126.34},{ 1.64,46.56},

{10.06,44.83},{31.51,69.10},{35.47,84.52},{66.99,138.93},

{82.74,136.11},{ 7.36,47.60},{10.53,41.04},{40.73,93.92},

{87.17,142.42},{83.59,128.06},{19.13,64.77},{71.61,125.54},

{18.40,48.78},{ 6.59,43.15},{14.23,43.09},{70.41,113.02},

{63.67,108.44},{88.16,141.10},{22.19,78.62},{44.58,93.87},

{55.29,117.20},{28.50,73.98},{66.78,104.10},{95.26,176.97},

{ 4.18,50.19},{34.13,91.19},{58.14,85.04},{72.06,122.19},

{42.39,96.96},{26.33,71.17},{ 5.69,41.83},{23.17,55.58},

{ 3.56,55.78},{71.39,128.64},{45.89,74.51},{78.30,132.40},

{21.54,46.44},{62.10,108.33},{54.73,96.88},{10.80,55.81},

{72.00,120.45},{92.33,153.68},{ 1.31,18.55},{70.06,111.80},

{33.85,72.07},{86.49,139.79},{56.43,94.46},{43.47,97.17},

{75.47,133.25},{72.26,129.18},{29.84,78.80},{ 1.82,42.51},

{93.75,151.00},{75.07,133.23},{39.94,80.08},{61.22,118.52},

{ 9.68,39.12},{23.95,57.84},{44.65,82.10},{26.38,63.76},

{65.70,121.14},{12.35,63.74},{47.04,106.62},{60.05,107.67},

{74.64,131.23},{58.54,117.45},{47.61,89.68},{63.53,108.34},

{ 7.32,63.44},{45.35,105.15},{28.85,90.30},{60.03,131.40},

{33.90,67.13},{ 2.05,32.69},{28.49,72.37},{23.40,67.77},

{81.49,145.13},{48.71,105.89},{46.98,109.38},{58.06,83.56},

{75.23,127.65},{91.40,156.76},{20.78,69.86},{45.73,89.21},

{72.87,127.24},{58.89,131.76},{91.96,141.63},{91.80,148.28},

{22.39,57.36},{61.77,117.66},{24.90,78.33},{95.01,136.15},

{64.29,124.96},{89.02,136.30},{98.38,140.31},{66.18,117.34},

{54.66,112.17},{10.44,53.84},{25.35,72.43},{22.47,59.29},

{93.47,150.06},{ 7.99,40.09},{48.93,88.72},{78.24,138.19},

{62.53,125.38},{86.04,130.48},{30.40,71.11},{ 9.97,60.61},

{82.62,160.19},{81.51,143.80},{41.83,81.75},{82.39,150.13},

{85.30,157.58},{59.23,110.39},{88.52,133.25},{41.29,88.67},

{64.50,128.95},{80.33,149.52},{48.19,100.19},{60.84,118.26},

{18.63,58.37},{78.91,146.85},{84.21,137.44},{26.72,63.58},

{61.81,125.62},{16.77,41.45},{30.46,61.76},{16.87,63.25},

{75.60,116.01},{50.78,105.46},{42.64,101.36},{14.73,44.78},

{ 3.33,37.77},{15.33,64.08},{36.74,80.41},{ 6.63,37.58},

{60.99,118.69},{58.12,102.34},{16.27,56.49},{65.14,130.18},

{63.38,111.27},{74.01,135.24},{39.81,94.41},{86.77,147.24},

{70.25,107.43},{48.21,109.92},{31.41,69.64},{30.73,40.73},

{50.98,101.70},{ 6.52,44.00},{41.34,98.63},{87.29,141.12},

{69.85,105.98},{26.68,74.42},{88.91,147.60},{40.25,86.31},

{71.44,126.51},{68.31,110.64},{29.95,71.05},{35.28,68.61},

{36.83,94.73},{47.24,82.88},{74.68,121.24},{56.35,106.54},

{13.17,37.00},{65.74,117.47},{20.17,74.11},{64.43,130.14},

{13.65,70.66},{59.65,113.54},{34.36,63.28},{18.73,62.41},

{73.07,125.50},{26.66,89.28},{56.01,119.69},{ 0.48,38.64},

{37.46,76.77},{45.27,81.75},{96.04,151.77},{34.60,63.29},

{11.76,46.15},{29.33,94.19},{68.31,112.95},{ 5.09,46.25},

{22.13,48.61},{76.64,124.24},{48.06,103.72},{26.21,72.33},

{66.88,117.07},{25.02,80.23},{35.85,54.85},{50.92,103.29},

{11.64,29.55},{12.94,70.93},{32.58,65.06},{57.92,103.82},

{64.44,107.58},{30.89,77.67},{43.40,76.89},{56.46,100.46},

{57.07,123.95},{59.04,103.72},{96.06,163.96},{29.34,68.45},

{69.21,118.71},{43.08,100.98},{95.23,134.64},{46.77,86.84},

{77.65,117.72},{71.08,141.42},{77.42,147.67},{41.68,94.07},

{68.76,108.25},{99.66,171.87},{27.65,67.73},{92.13,144.49},

{50.11,70.63},{91.27,136.47},{40.46,74.39},{19.79,69.27},

{50.04,103.40},{40.44,79.15},{ 0.45,41.34},{25.10,88.70},

{76.30,113.44},{47.10,98.46},{80.83,119.77},{ 6.37,26.84},

{19.74,76.29},{44.24,90.09},{94.47,143.16},{ 6.71,50.79},

{19.38,48.48},{88.21,150.47},{31.87,75.00},{25.63,74.67},

{28.78,78.23},{11.83,75.90},{28.73,81.03},{50.59,105.67},

{13.35,57.37},{96.39,152.53},{37.01,89.32},{ 7.35,38.97},

{85.03,151.09},{ 3.68,37.44},{53.42,120.05},{78.45,134.79},

{ 7.86,58.18},{83.32,138.73},{93.19,122.54},{15.32,44.11},

{11.55,40.72},{61.47,98.28},{ 6.45,49.13},{46.30,87.82},

{95.34,144.17},{34.58,100.12},{76.44,128.33},{83.44,139.04},

{68.80,124.51},{26.77,80.59},{36.77,90.92},{72.10,119.84},

{38.14,70.92},{ 4.28,53.11},{45.14,86.33},{61.44,104.96},

{55.63,100.83},{65.97,131.17},{75.18,119.38},{93.94,145.90},

{78.78,119.73},{ 4.68,35.69},{70.65,101.38},{ 8.91,54.46},

{45.14,101.74},{41.73,75.92},{25.19,75.63},{10.03,58.18},

{26.79,49.65},{62.36,108.77},{49.83,97.09},{24.29,68.82},

{77.92,141.99},{42.99,86.86},{70.19,123.69},{30.14,86.67},

{27.13,82.94},{ 1.86,55.20},{12.92,48.48},{45.55,107.95},

{34.13,69.04},{39.35,92.27},{20.88,76.24},{79.83,131.56},

{98.94,157.85},{79.66,115.24},{20.47,59.38},{94.72,165.60},

{53.83,86.21},{58.89,145.18},{81.60,133.30},{63.18,128.00},

{30.66,85.97},{81.63,139.20},{87.49,150.18},{82.80,136.23},

{52.84,97.80},{51.66,105.42},{72.22,137.64},{75.42,135.48},

{12.30,61.14},{73.99,122.95},{46.05,97.48},{79.07,145.18},

{41.02,79.02},{59.54,103.49},{65.55,116.42},{80.31,139.83},

{71.87,132.29},{99.21,151.95},{15.81,40.18},{15.64,54.51},

{99.28,172.71},{51.98,92.50},{61.09,97.87},{14.60,51.85}

};

Insert a table that shows running times for the original and CUDA versions.

Write a short analysis of the results

# MPI

## Password Cracking

Paste your source code for your MPI based password cracker here

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <crypt.h>

#include <math.h>

#include <time.h>

#include <mpi.h>

//Compile with mpicc CrackAZ99\_mpi.c -o CrackAZ99\_mpi -lcrypt

//Run with mpiexec ./CrackAZ99\_mpi > CrackAZ99MpiResults.txt

int n\_passwords = 4;

int size, rank;

char buffer[4];

char \*encrypted\_passwords[] = {

"$6$KB$H8s0k9/1RQ783G9gF69Xkn.MI.Dq5Ox0va/dFlkknNjO7trgekVOjTv1BKCb.nm3vqxmtO2mOplhmFkwZXecz0",

"$6$KB$WksuNcTfYjZWjDC4Zt3ZAmQ38OrsWwHyDgf/grFJ2Sgg.qpOz56lMpBVfWYdQZa9Pksa2TJRVYVb3K.mbYx4Y1",

"$6$KB$0ZqvOLHpRgU9vLhzavKkL37MCDESwi2NDoTptnw4jyAjQGVtizjiKaluE60l1k7b.7YzDFU3biOo7Cr2SnvzT1",

"$6$KB$UwKD1iCsvhAryQWAH6o8C9B6dEtOUOhYCgBfwtvffD.Ycz83.8GZ/9dhfIyVodUtHRyUl8A8LRfCNSlx8Lb2O1"

};

void substr(char \*dest, char \*src, int start, int length){

memcpy(dest, src + start, length);

\*(dest + length) = '\0';

}

void crack(char \*salt\_and\_encrypted){

int x, y, z;

char salt[7];

char plain[7];

char \*enc;

int count = 0;

char xStart, yStart, xEnd, yEnd;

if (rank == 1) {

xStart = 'A';

yStart = 'A';

xEnd = 'M';

yEnd = 'Z';

}

if (rank == 2) {

xStart = 'N';

yStart = 'A';

xEnd = 'Z';

yEnd = 'Z';

}

substr(salt, salt\_and\_encrypted, 0, 6);

for(x=xStart; x<=xEnd; x++){

for(y=yStart; y<=yEnd; y++){

for(z=0; z<=99; z++){

sprintf(plain, "%c%c%02d", x, y,z);

enc = (char \*) crypt(plain, salt);

count++;

if(strcmp(salt\_and\_encrypted, enc) == 0){

printf("#%-8d%s %s\n", count, plain, enc);

} else {

printf(" %-8d%s %s\n", count, plain, enc);

}

}

}

}

printf("%d solutions explored\n", count);

}

int time\_difference(struct timespec \*start, struct timespec \*finish, long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if (dn < 0) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

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

MPI\_Status status;

MPI\_Init(NULL, NULL);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &size);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

int i;

if (rank == 0) {

MPI\_Finalize();

}

else {

for (i = 0; i < n\_passwords; i++) {

crack(encrypted\_passwords[i]);

}

}

MPI\_Finalize();

clock\_gettime(CLOCK\_MONOTONIC, &finish);

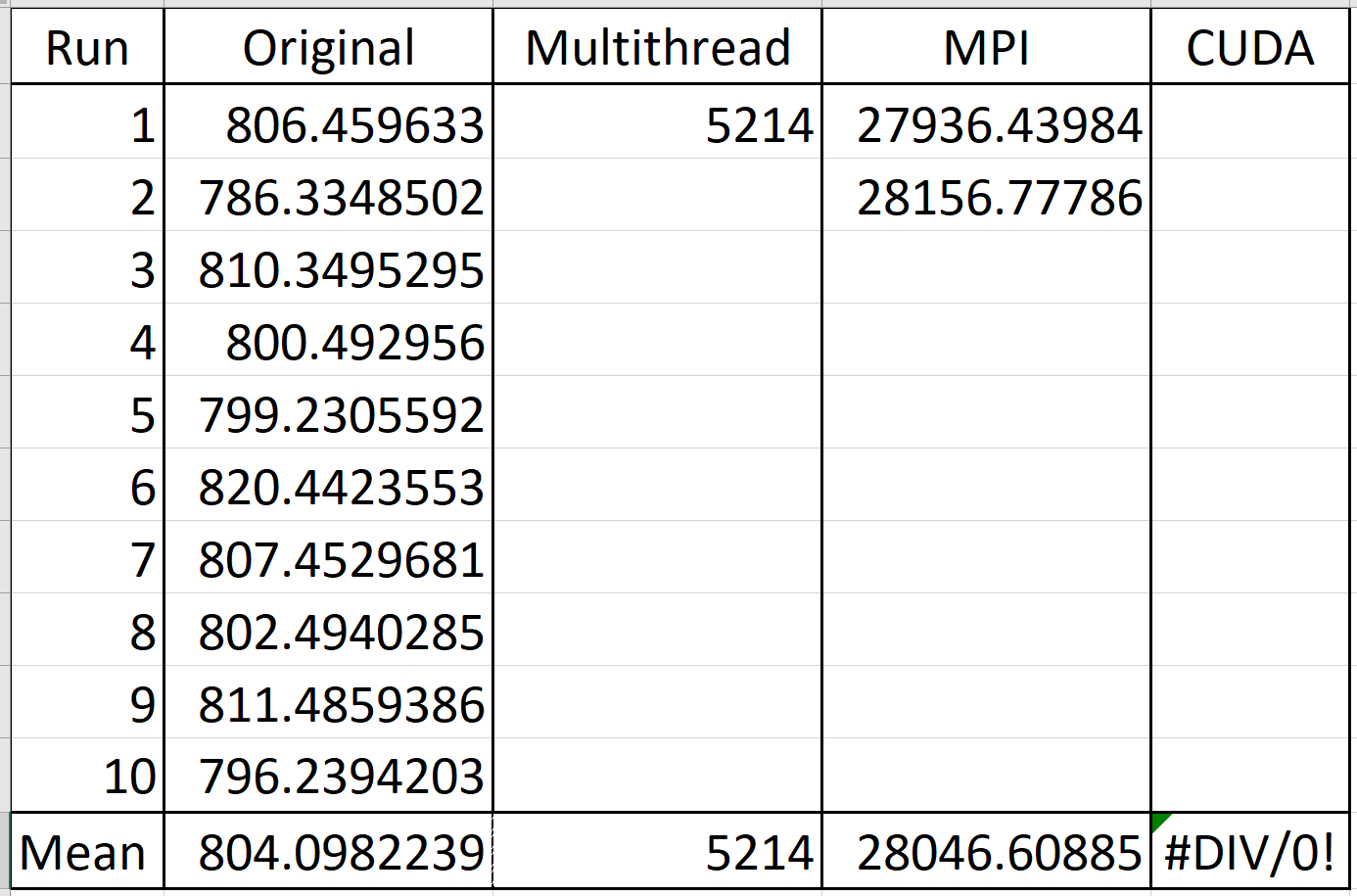
time\_difference(&start, &finish, &time\_elapsed);

printf("Time elasped was %lldns or %0.9lfs\n", time\_elapsed, (time\_elapsed / 1.0e9));

return 0;

}

Insert a table that shows running times for the original and MPI versions.



Write a short analysis of the results

Due to time constraints, I ran the program on 1 password and multiplying the result by 4 and the average came to 28046 seconds.With 2 letters and 4 numbers, there is a total of 27,040,000 combinations. As this was ran on a university machine, I ran the initial password cracker twice to get an average ‘combinations per second’ which resulted in 511 combinations per second. If we divide 27,040,000 by 511, we get 52,915 seconds. And using MPI with 2 machines, if we divide that by 2, we get 26,457 seconds as out total time which is approximately 24 minutes off our actual time.

## Image Processing

Paste your source code for your MPI based image processor

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <GL/glut.h>

#include <GL/gl.h>

#include <malloc.h>

#include <signal.h>

#include <time.h>

#include <mpi.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Displays two grey scale images. On the left is an image that has come from an

image processing pipeline, just after colour thresholding. On the right is

the result of applying an edge detection convolution operator to the left

image. This program performs that convolution.

Things to note:

- A single unsigned char stores a pixel intensity value. 0 is black, 256 is

white.

- The colour mode used is GL\_LUMINANCE. This uses a single number to

represent a pixel's intensity. In this case we want 256 shades of grey,

which is best stored in eight bits, so GL\_UNSIGNED\_BYTE is specified as

the pixel data type.

To compile adapt the code below wo match your filenames:

mpicc ip\_coursework\_041\_mpi.c -o ip\_coursework\_041\_mpi -lglut -lGL -lm

Dr Kevan Buckley, University of Wolverhampton, 2018

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define width 100

#define height 72

int size, rank;

unsigned char image[], results[width \* height], buffer[width \* height];

void detect\_edges(unsigned char \*in, unsigned char \*out) {

int i;

int n\_pixels = width \* height;

for(i=rank; i<n\_pixels; i = i + size) {

int x, y; // the pixel of interest

int b, d, f, h; // the pixels adjacent to x,y used for the calculation

int r; // the result of calculate

y = i / width;

x = i - (width \* y);

if (x == 0 || y == 0 || x == width - 1 || y == height - 1) {

results[i] = 0;

} else {

b = i + width;

d = i - 1;

f = i + 1;

h = i - width;

r = (in[i] \* 4) + (in[b] \* -1) + (in[d] \* -1) + (in[f] \* -1)

+ (in[h] \* -1);

if (r > 0) { // if the result is positive this is an edge pixel

out[i] = 255;

} else {

out[i] = 0;

}

}

}

}

void tidy\_and\_exit() {

exit(0);

}

void sigint\_callback(int signal\_number){

printf("\nInterrupt from keyboard\n");

tidy\_and\_exit();

}

int time\_difference(struct timespec \*start, struct timespec \*finish, long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if (dn < 0) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

static void display() {

glClear(GL\_COLOR\_BUFFER\_BIT);

glRasterPos4i(-1, -1, 0, 1);

glDrawPixels(width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, image);

glRasterPos4i(0, -1, 0, 1);

glDrawPixels(width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, results);

glFlush();

}

static void key\_pressed(unsigned char key, int x, int y) {

switch(key){

case 27: // escape

tidy\_and\_exit();

break;

default:

printf("\nPress escape to exit\n");

break;

}

}

int main(int argc, char \*\*argv) {

MPI\_Status status;

MPI\_Init(NULL, NULL);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &size);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

signal(SIGINT, sigint\_callback);

printf("image dimensions %dx%d\n", width, height);

detect\_edges(image, results);

if(rank == 0){

MPI\_Recv(buffer, width \* height, MPI\_BYTE, MPI\_ANY\_SOURCE, MPI\_ANY\_TAG, MPI\_COMM\_WORLD, &status);

int n\_pixels = width \* height;

for(int i = status.MPI\_SOURCE; i < n\_pixels; i = i + size){

results[i] = buffer[i];

}

} else {

MPI\_Send(results, width \* height, MPI\_BYTE, 0, 0, MPI\_COMM\_WORLD);

}

MPI\_Finalize();

clock\_gettime(CLOCK\_MONOTONIC, &finish);

time\_difference(&start, &finish, &time\_elapsed);

printf("Time elasped was %lldns or %0.9lfs\n", time\_elapsed, (time\_elapsed/1.0e9));

glutInit(&argc, argv);

glutInitWindowSize(width \* 2,height);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_LUMINANCE);

glutCreateWindow("6CS005 Image Progessing Courework");

glutDisplayFunc(display);

glutKeyboardFunc(key\_pressed);

glClearColor(0.0, 1.0, 0.0, 1.0);

glutMainLoop();

tidy\_and\_exit();

return 0;

}

unsigned char image[] = {255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,255,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,255,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,255,255,

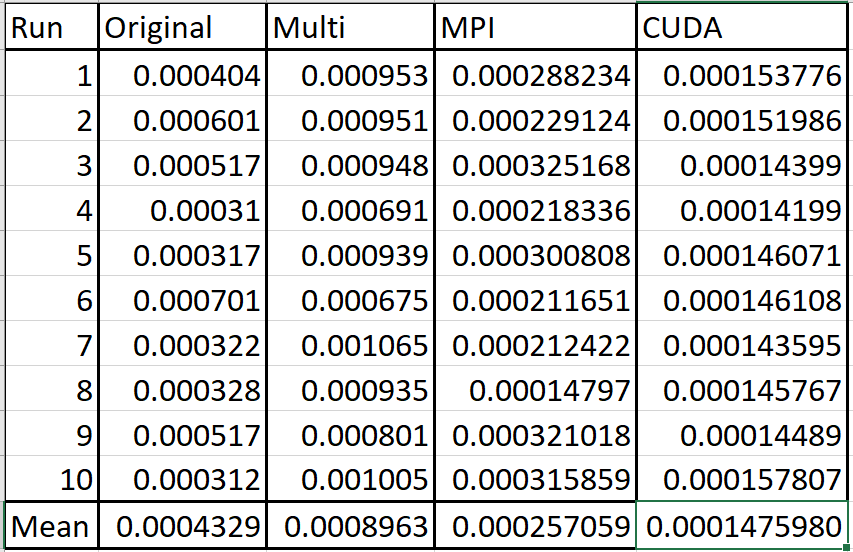
255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255

};

Insert a table that shows running times for the original and MPI versions.



Write a short analysis of the results

The performance of the program was halved compared to the original program, but the size of the image means that the results aren’t really showing the true performance of MPI. If the image was of a greater size, the performance difference can be seen.

## Linear Regression

Paste your source code for your MPI based linear regression

#include <stdio.h>

#include <math.h>

#include <mpi.h>

#include <time.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This program takes an initial estimate of m and c and finds the associated

\* rms error. It is then as a base to generate and evaluate 8 new estimates,

\* which are steps in different directions in m-c space. The best estimate is

\* then used as the base for another iteration of "generate and evaluate". This

\* continues until none of the new estimates are better than the base. This is

\* a gradient search for a minimum in mc-space.

\*

\* To compile:

\* mpicc lr\_1607117\_mpi.c -o lr\_1607117\_mpi -lm

\*

\* To run:

\* mpiexec -n 8 ./lr\_1607117\_mpi

\*

\* Dr Kevan Buckley, University of Wolverhampton, 2018

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int size ,rank;

typedef struct point\_t {

double x;

double y;

} point\_t;

typedef struct mpidata {

double dm , dc;

} mpidata;

int n\_data = 1000;

point\_t data[];

double residual\_error(double x, double y, double m, double c) {

double e = (m \* x) + c - y;

return e \* e;

}

double rms\_error(double m, double c) {

int i;

double mean;

double error\_sum = 0;

for(i=0; i<n\_data; i++) {

error\_sum += residual\_error(data[i].x, data[i].y, m, c);

}

mean = error\_sum / n\_data;

return sqrt(mean);

}

int time\_difference(struct timespec \*start, struct timespec \*finish, long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if (dn < 0) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

void printData() {

for(int i = 0; i < n\_data; i++){

printf("%0.2lf, %0.2lf\n", data[i].x, data[i].y);

}

}

int main() {

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

int i;

double bm = 1.3;

double bc = 10;

double be;

double dm[8];

double dc[8];

double e[8];

double step = 0.01;

double best\_error = 999999999;

int best\_error\_i;

int minimum\_found = 0;

double er;

double om[] = {0,1,1, 1, 0,-1,-1,-1};

double oc[] = {1,1,0,-1,-1,-1, 0, 1};

mpidata dmdc;

MPI\_Status status;

MPI\_Init(NULL, NULL);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &size);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);

be = rms\_error(bm, bc);

while(!minimum\_found) {

if (rank == 0) {

for(i = 0; i < 8; i++) {

dm[i] = bm + (om[i] \* step);

dc[i] = bc + (oc[i] \* step);

}

for(i = 1; i < 8; i++) {

printf("Sending i=%d\n", i);

dmdc.dm = dm[i];

dmdc.dc = dc[i];

MPI\_Send(&dmdc, 2, MPI\_DOUBLE, i, 0, MPI\_COMM\_WORLD);

}

for(i = 1; i < 8; i++) {

printf("recieving i=%d\n",i);

MPI\_Recv(&e[i], 1, MPI\_DOUBLE, i, 0, MPI\_COMM\_WORLD, &status);

}

e[0] = rms\_error(dm[0], dc[0]);

for(i = 0; i < 8; i++) {

e[i] = rms\_error(dm[i], dc[i]);

if(e[i] < best\_error) {

best\_error = e[i];

best\_error\_i = i;

}

}

printf("best m,c is %lf,%lf with error %lf in direction %d\n", dm[best\_error\_i], dc[best\_error\_i], best\_error, best\_error\_i);

if(best\_error < be) {

be = best\_error;

bm = dm[best\_error\_i];

bc = dc[best\_error\_i];

} else {

minimum\_found = 1;

}

for(i = 1; i < 8; i++) {

MPI\_Send(&minimum\_found, 1, MPI\_INT, i, 0, MPI\_COMM\_WORLD);

}

} else {

MPI\_Recv(&dmdc, 2 , MPI\_DOUBLE, 0, 0, MPI\_COMM\_WORLD, &status);

double er = rms\_error(dmdc.dm, dmdc.dc);

MPI\_Send(&er, 1, MPI\_DOUBLE, 0, 0, MPI\_COMM\_WORLD);

MPI\_Recv(&minimum\_found, 1 , MPI\_INT, 0, 0, MPI\_COMM\_WORLD, &status);

}

}

//printf("minimum m,c is %lf,%lf with error %lf\n", bm, bc, be);

printData();

MPI\_Finalize();

clock\_gettime(CLOCK\_MONOTONIC, &finish);

time\_difference(&start, &finish, &time\_elapsed);

printf("Time elapsed was, %lldns or %0.9lf,s\n", time\_elapsed, (time\_elapsed/1.0e9));

return 0;

}

point\_t data[] = {

{72.33,116.23},{65.60,125.82},{82.57,146.00},{80.88,137.75},

{82.94,149.23},{69.01,130.26},{69.93,119.66},{84.53,139.23},

{74.46,120.78},{72.23,132.06},{73.97,125.06},{84.82,157.53},

{65.32,126.27},{25.48,54.28},{45.60,120.47},{46.13,96.54},

{11.51,33.40},{13.18,53.29},{57.97,112.64},{79.20,125.46},

{26.68,82.42},{38.91,90.74},{99.99,176.95},{34.61,75.19},

{65.93,113.31},{ 7.61,44.09},{53.98,105.43},{10.55,46.01},

{94.27,153.58},{61.53,105.73},{81.12,126.08},{63.49,116.87},

{53.78,126.49},{ 7.34,55.64},{27.39,76.97},{89.04,137.09},

{36.95,89.50},{59.10,104.19},{21.35,60.88},{39.57,69.60},

{56.36,100.98},{20.73,64.83},{39.15,95.79},{81.43,140.73},

{32.00,70.93},{29.76,54.41},{44.53,104.59},{47.23,96.12},

{84.71,121.86},{17.18,68.43},{98.62,167.64},{22.84,55.40},

{42.99,82.12},{83.66,142.41},{21.75,53.82},{24.85,68.61},

{45.96,101.72},{64.26,124.28},{ 1.00,35.25},{96.71,175.95},

{46.94,100.59},{19.59,82.78},{ 8.37,54.67},{79.16,141.01},

{62.46,108.18},{39.11,85.42},{ 2.27,50.84},{94.30,157.77},

{50.17,87.96},{99.06,154.56},{ 1.29,44.24},{74.05,114.60},

{98.71,151.08},{86.06,118.25},{33.13,79.31},{98.60,149.35},

{88.56,129.10},{84.79,141.62},{64.04,120.73},{27.27,70.21},

{35.90,78.01},{74.09,127.46},{38.97,88.60},{43.85,94.20},

{37.41,78.61},{21.36,56.50},{52.64,93.14},{21.82,74.60},

{45.77,98.77},{70.76,140.77},{91.77,161.08},{15.73,74.69},

{18.16,53.34},{18.17,82.46},{59.67,111.96},{78.36,126.93},

{75.22,126.51},{ 0.95,44.63},{79.89,131.69},{66.90,136.73},

{81.03,153.00},{21.94,77.30},{28.69,80.81},{70.57,115.95},

{ 0.07,22.49},{94.12,157.13},{ 3.59,48.46},{61.01,103.44},

{64.78,104.20},{29.86,52.58},{78.51,137.27},{70.33,119.81},

{55.58,94.36},{67.20,133.29},{48.32,98.63},{99.98,157.23},

{54.67,109.57},{15.32,61.70},{87.35,144.62},{ 5.82,33.91},

{84.79,151.17},{34.64,87.41},{45.29,101.66},{36.98,83.28},

{11.09,50.22},{89.77,152.92},{63.20,116.32},{60.60,122.47},

{71.72,119.19},{19.84,54.18},{34.72,94.21},{38.19,92.41},

{54.43,127.94},{99.18,155.69},{32.23,70.20},{25.12,51.92},

{33.23,75.57},{93.88,137.59},{88.93,127.51},{55.75,89.77},

{58.41,109.93},{79.37,122.64},{ 5.81,44.89},{80.70,121.22},

{76.34,131.51},{19.62,62.24},{21.84,56.12},{59.57,104.44},

{27.69,91.00},{92.23,137.27},{25.26,65.85},{53.06,94.95},

{28.28,78.48},{90.26,162.47},{52.61,111.55},{94.91,163.98},

{27.08,62.15},{91.16,152.42},{65.96,127.74},{76.54,121.15},

{32.22,63.29},{98.38,150.77},{95.70,136.24},{29.01,88.91},

{99.56,147.71},{29.44,89.33},{84.69,146.94},{86.90,158.80},

{53.30,119.52},{60.40,114.86},{ 0.59,34.27},{99.97,141.80},

{ 2.72,43.00},{34.43,60.52},{62.52,113.96},{50.03,99.76},

{ 8.49,50.88},{63.49,118.19},{84.23,147.78},{77.01,119.34},

{56.26,112.80},{53.45,112.79},{64.22,111.71},{51.92,101.86},

{90.80,146.00},{94.94,171.92},{30.04,71.52},{80.19,118.37},

{91.22,150.73},{51.59,110.35},{50.97,100.77},{47.97,102.84},

{84.94,130.91},{74.46,139.04},{15.74,71.37},{ 4.57,47.96},

{28.75,71.79},{44.19,82.95},{31.80,89.48},{39.30,77.93},

{33.39,83.07},{84.33,158.75},{77.52,138.07},{65.89,121.84},

{93.02,165.39},{97.49,175.18},{30.08,94.14},{45.02,94.50},

{37.66,92.79},{73.63,123.05},{51.13,89.75},{81.25,131.11},

{ 8.90,49.55},{ 9.32,59.64},{27.76,60.36},{75.58,129.60},

{43.10,86.91},{87.87,152.23},{27.30,70.75},{57.98,95.87},

{62.52,109.09},{93.91,159.40},{62.62,104.22},{99.30,159.09},

{78.28,136.25},{67.68,104.24},{85.23,132.67},{14.60,60.69},

{85.38,148.84},{44.34,94.83},{13.50,44.50},{76.68,124.87},

{15.43,62.48},{35.32,78.54},{68.48,127.26},{50.28,93.15},

{63.70,111.55},{91.93,135.01},{85.03,138.98},{35.37,88.28},

{22.93,61.44},{60.23,116.41},{53.35,92.31},{13.71,70.26},

{ 5.03,42.40},{64.61,103.22},{64.89,95.19},{47.00,90.87},

{78.49,141.71},{73.23,118.64},{79.45,139.97},{61.60,109.08},

{26.67,77.21},{99.37,160.10},{49.01,90.96},{73.60,117.77},

{84.51,132.40},{56.07,109.38},{22.83,48.57},{25.87,75.17},

{18.84,52.97},{ 5.31,43.79},{11.22,61.06},{19.20,65.43},

{97.46,156.14},{68.72,99.82},{61.00,120.78},{70.51,135.44},

{93.43,136.73},{16.75,61.27},{15.07,55.68},{47.48,93.87},

{67.55,125.68},{22.00,79.66},{33.38,90.18},{19.16,48.03},

{85.13,138.59},{51.34,106.70},{86.33,140.79},{68.09,132.99},

{51.09,106.86},{83.63,141.68},{42.59,108.74},{ 6.12,50.21},

{55.23,96.37},{31.74,69.33},{89.85,153.96},{18.83,63.27},

{47.23,108.44},{92.33,139.36},{95.72,149.13},{80.73,125.62},

{53.06,98.61},{87.32,138.65},{92.85,149.19},{99.67,163.38},

{69.39,131.02},{51.06,83.57},{20.59,72.15},{61.13,120.47},

{90.82,139.61},{89.09,143.72},{93.01,154.19},{92.01,145.65},

{90.33,138.25},{38.71,88.60},{95.58,156.07},{67.19,135.06},

{95.13,158.49},{64.62,132.58},{23.18,48.39},{39.56,102.96},

{86.94,132.60},{57.89,109.04},{ 0.67,38.37},{26.47,79.75},

{36.40,96.08},{ 0.06,31.58},{66.36,117.15},{89.19,138.62},

{61.08,105.82},{24.66,88.04},{ 3.19,51.54},{48.14,107.31},

{28.35,64.84},{76.30,134.96},{ 9.38,59.83},{ 8.70,63.22},

{90.26,152.32},{48.83,85.96},{27.34,58.72},{66.19,128.16},

{22.67,69.30},{17.03,71.17},{32.00,80.25},{28.89,58.28},

{ 9.94,38.71},{48.51,113.01},{70.62,129.59},{ 8.28,39.52},

{35.13,81.48},{81.79,137.37},{34.03,79.61},{82.59,137.87},

{ 7.05,40.53},{34.75,66.62},{55.45,106.21},{47.25,114.02},

{82.38,130.01},{14.89,36.16},{77.27,124.82},{32.32,69.57},

{32.78,68.38},{21.65,50.00},{96.78,159.60},{27.22,79.27},

{71.07,116.26},{93.78,161.73},{55.52,119.04},{66.00,130.64},

{95.61,137.42},{43.09,81.70},{37.62,88.84},{24.91,69.19},

{46.04,88.96},{ 6.42,35.03},{99.97,170.68},{ 1.00,46.06},

{46.00,101.57},{37.43,77.28},{79.99,131.00},{44.25,87.13},

{68.76,117.99},{18.81,50.96},{45.09,100.89},{ 5.85,54.62},

{24.84,76.68},{19.62,68.98},{51.77,107.31},{31.29,81.49},

{58.86,108.16},{33.99,79.87},{37.87,77.60},{68.60,109.32},

{15.51,68.92},{22.48,65.55},{72.24,130.72},{74.96,116.42},

{13.57,43.08},{98.93,159.92},{21.85,67.26},{49.84,93.75},

{78.31,142.09},{59.98,125.07},{42.86,84.83},{49.43,115.55},

{72.23,107.78},{ 4.87,42.69},{41.84,85.28},{80.40,143.34},

{94.81,164.37},{81.61,114.26},{96.26,171.13},{27.11,69.32},

{ 6.98,47.72},{44.77,98.96},{71.87,118.76},{70.11,116.96},

{59.48,126.08},{69.21,120.97},{19.85,63.40},{94.00,163.24},

{17.86,59.95},{12.45,51.99},{92.15,142.22},{34.93,67.61},

{19.74,66.74},{ 7.40,50.96},{ 6.57,30.48},{98.72,154.60},

{28.61,72.88},{ 4.43,44.38},{28.11,89.01},{96.65,156.93},

{ 6.90,38.64},{37.78,79.98},{94.36,147.68},{93.28,153.10},

{52.05,111.89},{73.08,128.36},{37.75,84.67},{ 3.65,63.87},

{26.51,58.58},{88.30,147.46},{63.77,119.61},{81.52,138.61},

{70.03,127.36},{30.41,77.76},{85.24,149.27},{85.93,144.10},

{46.09,69.17},{74.64,136.62},{61.53,103.27},{51.07,92.56},

{ 8.16,43.01},{63.83,109.11},{84.75,125.04},{86.41,134.50},

{18.47,39.18},{50.80,89.05},{87.07,152.46},{70.04,129.08},

{45.43,95.29},{95.39,162.15},{74.83,121.81},{ 5.73,64.72},

{74.23,136.09},{97.70,172.89},{ 1.22,26.48},{20.72,73.08},

{25.03,62.46},{ 3.76,39.34},{77.69,136.86},{ 1.30,31.95},

{91.69,153.91},{32.25,68.09},{23.04,53.70},{79.21,122.57},

{68.75,117.23},{ 5.88,38.05},{31.93,74.37},{47.55,94.98},

{ 5.59,58.13},{94.48,150.22},{91.97,146.82},{84.15,126.56},

{71.04,108.41},{42.47,96.04},{81.39,138.08},{77.96,131.16},

{91.82,151.27},{34.04,79.48},{ 0.53,30.94},{14.15,52.35},

{44.82,84.82},{89.88,143.65},{45.13,98.84},{89.12,139.38},

{19.59,65.34},{55.58,114.44},{36.73,77.30},{52.77,108.62},

{86.92,149.26},{76.05,132.12},{75.22,138.21},{89.76,136.95},

{ 4.54,47.45},{35.78,76.88},{25.01,71.89},{65.30,111.91},

{74.20,110.85},{22.65,63.13},{71.54,118.44},{78.36,145.70},

{31.77,84.24},{11.97,49.57},{83.68,144.15},{59.02,102.16},

{16.85,59.15},{51.91,81.02},{48.07,84.39},{65.48,118.73},

{39.75,89.61},{98.46,156.11},{20.48,59.08},{16.35,64.48},

{75.24,127.44},{98.64,179.05},{12.45,55.40},{93.48,154.82},

{54.17,120.32},{60.56,136.76},{89.94,130.15},{10.72,54.90},

{ 6.32,70.77},{63.21,112.71},{19.36,67.27},{71.52,144.73},

{80.65,127.50},{46.39,95.19},{59.58,115.02},{12.02,65.02},

{99.70,151.17},{81.59,137.17},{ 8.89,49.49},{63.89,111.69},

{65.46,112.13},{14.15,49.91},{43.45,83.77},{ 1.67,54.23},

{40.52,92.39},{22.32,57.37},{ 8.56,52.80},{57.84,109.69},

{92.72,135.66},{42.64,69.07},{15.10,60.72},{55.32,105.93},

{48.08,105.63},{33.51,83.41},{46.33,112.30},{30.13,78.55},

{53.97,102.01},{40.55,83.32},{14.88,67.56},{41.64,104.75},

{42.53,87.16},{71.94,129.40},{ 6.32,31.23},{99.32,149.27},

{73.61,124.50},{87.06,131.53},{27.62,78.88},{94.90,162.90},

{ 2.40,49.05},{44.58,90.13},{90.72,155.10},{80.81,141.76},

{ 7.88,69.69},{81.66,126.49},{47.68,91.73},{69.69,128.29},

{79.42,138.01},{ 0.10,41.72},{96.13,155.89},{91.87,141.95},

{77.38,125.46},{99.52,160.11},{59.65,120.16},{32.13,70.52},

{89.05,153.16},{79.55,131.89},{67.62,130.81},{98.38,154.67},

{75.67,136.01},{41.31,65.92},{79.18,134.82},{36.27,97.93},

{76.96,141.60},{54.56,93.38},{15.69,48.26},{88.68,124.28},

{62.15,115.92},{77.61,131.25},{69.60,132.93},{96.65,140.01},

{ 8.27,41.39},{20.23,76.12},{51.96,97.55},{10.19,56.26},

{93.39,149.83},{83.10,140.59},{69.99,139.88},{50.20,85.62},

{38.75,90.19},{ 4.50,36.25},{74.43,121.23},{ 0.49,31.23},

{98.24,150.34},{35.98,92.96},{23.67,56.39},{ 5.28,47.60},

{78.86,119.11},{ 4.99,36.41},{30.84,87.47},{91.53,147.30},

{26.22,84.60},{ 9.60,62.81},{10.96,63.28},{28.81,60.22},

{98.16,163.14},{ 6.64,42.20},{ 8.03,46.72},{64.23,123.19},

{33.94,66.05},{37.36,76.94},{90.32,134.01},{16.56,51.76},

{42.31,87.32},{49.33,95.72},{27.98,80.21},{27.62,68.52},

{18.19,67.44},{18.30,57.71},{82.93,117.18},{51.71,100.15},

{ 2.83,45.28},{63.05,103.43},{37.07,86.64},{42.38,91.22},

{14.52,58.42},{27.80,67.92},{25.47,54.04},{23.99,78.44},

{63.16,122.26},{83.74,137.47},{67.38,135.39},{17.35,56.35},

{54.99,110.49},{53.02,96.05},{64.48,121.13},{13.01,45.22},

{89.72,148.69},{84.23,138.28},{75.49,131.33},{87.18,144.52},

{36.51,87.61},{ 6.44,46.72},{43.35,86.92},{19.84,56.37},

{54.44,111.70},{17.93,43.86},{47.66,108.95},{47.07,88.88},

{95.04,135.07},{39.26,94.81},{83.96,131.28},{71.75,115.77},

{19.67,48.82},{74.69,136.22},{84.39,126.34},{ 1.64,46.56},

{10.06,44.83},{31.51,69.10},{35.47,84.52},{66.99,138.93},

{82.74,136.11},{ 7.36,47.60},{10.53,41.04},{40.73,93.92},

{87.17,142.42},{83.59,128.06},{19.13,64.77},{71.61,125.54},

{18.40,48.78},{ 6.59,43.15},{14.23,43.09},{70.41,113.02},

{63.67,108.44},{88.16,141.10},{22.19,78.62},{44.58,93.87},

{55.29,117.20},{28.50,73.98},{66.78,104.10},{95.26,176.97},

{ 4.18,50.19},{34.13,91.19},{58.14,85.04},{72.06,122.19},

{42.39,96.96},{26.33,71.17},{ 5.69,41.83},{23.17,55.58},

{ 3.56,55.78},{71.39,128.64},{45.89,74.51},{78.30,132.40},

{21.54,46.44},{62.10,108.33},{54.73,96.88},{10.80,55.81},

{72.00,120.45},{92.33,153.68},{ 1.31,18.55},{70.06,111.80},

{33.85,72.07},{86.49,139.79},{56.43,94.46},{43.47,97.17},

{75.47,133.25},{72.26,129.18},{29.84,78.80},{ 1.82,42.51},

{93.75,151.00},{75.07,133.23},{39.94,80.08},{61.22,118.52},

{ 9.68,39.12},{23.95,57.84},{44.65,82.10},{26.38,63.76},

{65.70,121.14},{12.35,63.74},{47.04,106.62},{60.05,107.67},

{74.64,131.23},{58.54,117.45},{47.61,89.68},{63.53,108.34},

{ 7.32,63.44},{45.35,105.15},{28.85,90.30},{60.03,131.40},

{33.90,67.13},{ 2.05,32.69},{28.49,72.37},{23.40,67.77},

{81.49,145.13},{48.71,105.89},{46.98,109.38},{58.06,83.56},

{75.23,127.65},{91.40,156.76},{20.78,69.86},{45.73,89.21},

{72.87,127.24},{58.89,131.76},{91.96,141.63},{91.80,148.28},

{22.39,57.36},{61.77,117.66},{24.90,78.33},{95.01,136.15},

{64.29,124.96},{89.02,136.30},{98.38,140.31},{66.18,117.34},

{54.66,112.17},{10.44,53.84},{25.35,72.43},{22.47,59.29},

{93.47,150.06},{ 7.99,40.09},{48.93,88.72},{78.24,138.19},

{62.53,125.38},{86.04,130.48},{30.40,71.11},{ 9.97,60.61},

{82.62,160.19},{81.51,143.80},{41.83,81.75},{82.39,150.13},

{85.30,157.58},{59.23,110.39},{88.52,133.25},{41.29,88.67},

{64.50,128.95},{80.33,149.52},{48.19,100.19},{60.84,118.26},

{18.63,58.37},{78.91,146.85},{84.21,137.44},{26.72,63.58},

{61.81,125.62},{16.77,41.45},{30.46,61.76},{16.87,63.25},

{75.60,116.01},{50.78,105.46},{42.64,101.36},{14.73,44.78},

{ 3.33,37.77},{15.33,64.08},{36.74,80.41},{ 6.63,37.58},

{60.99,118.69},{58.12,102.34},{16.27,56.49},{65.14,130.18},

{63.38,111.27},{74.01,135.24},{39.81,94.41},{86.77,147.24},

{70.25,107.43},{48.21,109.92},{31.41,69.64},{30.73,40.73},

{50.98,101.70},{ 6.52,44.00},{41.34,98.63},{87.29,141.12},

{69.85,105.98},{26.68,74.42},{88.91,147.60},{40.25,86.31},

{71.44,126.51},{68.31,110.64},{29.95,71.05},{35.28,68.61},

{36.83,94.73},{47.24,82.88},{74.68,121.24},{56.35,106.54},

{13.17,37.00},{65.74,117.47},{20.17,74.11},{64.43,130.14},

{13.65,70.66},{59.65,113.54},{34.36,63.28},{18.73,62.41},

{73.07,125.50},{26.66,89.28},{56.01,119.69},{ 0.48,38.64},

{37.46,76.77},{45.27,81.75},{96.04,151.77},{34.60,63.29},

{11.76,46.15},{29.33,94.19},{68.31,112.95},{ 5.09,46.25},

{22.13,48.61},{76.64,124.24},{48.06,103.72},{26.21,72.33},

{66.88,117.07},{25.02,80.23},{35.85,54.85},{50.92,103.29},

{11.64,29.55},{12.94,70.93},{32.58,65.06},{57.92,103.82},

{64.44,107.58},{30.89,77.67},{43.40,76.89},{56.46,100.46},

{57.07,123.95},{59.04,103.72},{96.06,163.96},{29.34,68.45},

{69.21,118.71},{43.08,100.98},{95.23,134.64},{46.77,86.84},

{77.65,117.72},{71.08,141.42},{77.42,147.67},{41.68,94.07},

{68.76,108.25},{99.66,171.87},{27.65,67.73},{92.13,144.49},

{50.11,70.63},{91.27,136.47},{40.46,74.39},{19.79,69.27},

{50.04,103.40},{40.44,79.15},{ 0.45,41.34},{25.10,88.70},

{76.30,113.44},{47.10,98.46},{80.83,119.77},{ 6.37,26.84},

{19.74,76.29},{44.24,90.09},{94.47,143.16},{ 6.71,50.79},

{19.38,48.48},{88.21,150.47},{31.87,75.00},{25.63,74.67},

{28.78,78.23},{11.83,75.90},{28.73,81.03},{50.59,105.67},

{13.35,57.37},{96.39,152.53},{37.01,89.32},{ 7.35,38.97},

{85.03,151.09},{ 3.68,37.44},{53.42,120.05},{78.45,134.79},

{ 7.86,58.18},{83.32,138.73},{93.19,122.54},{15.32,44.11},

{11.55,40.72},{61.47,98.28},{ 6.45,49.13},{46.30,87.82},

{95.34,144.17},{34.58,100.12},{76.44,128.33},{83.44,139.04},

{68.80,124.51},{26.77,80.59},{36.77,90.92},{72.10,119.84},

{38.14,70.92},{ 4.28,53.11},{45.14,86.33},{61.44,104.96},

{55.63,100.83},{65.97,131.17},{75.18,119.38},{93.94,145.90},

{78.78,119.73},{ 4.68,35.69},{70.65,101.38},{ 8.91,54.46},

{45.14,101.74},{41.73,75.92},{25.19,75.63},{10.03,58.18},

{26.79,49.65},{62.36,108.77},{49.83,97.09},{24.29,68.82},

{77.92,141.99},{42.99,86.86},{70.19,123.69},{30.14,86.67},

{27.13,82.94},{ 1.86,55.20},{12.92,48.48},{45.55,107.95},

{34.13,69.04},{39.35,92.27},{20.88,76.24},{79.83,131.56},

{98.94,157.85},{79.66,115.24},{20.47,59.38},{94.72,165.60},

{53.83,86.21},{58.89,145.18},{81.60,133.30},{63.18,128.00},

{30.66,85.97},{81.63,139.20},{87.49,150.18},{82.80,136.23},

{52.84,97.80},{51.66,105.42},{72.22,137.64},{75.42,135.48},

{12.30,61.14},{73.99,122.95},{46.05,97.48},{79.07,145.18},

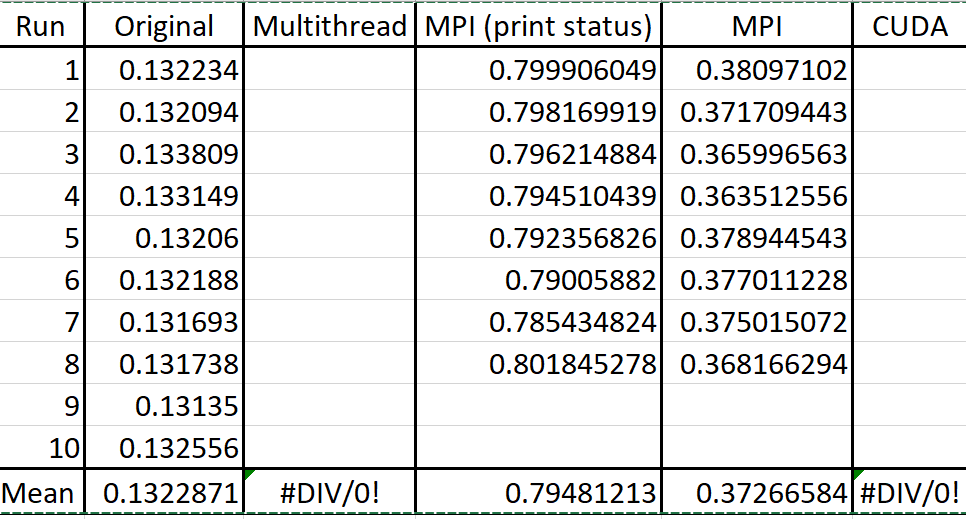
{41.02,79.02},{59.54,103.49},{65.55,116.42},{80.31,139.83},

{71.87,132.29},{99.21,151.95},{15.81,40.18},{15.64,54.51},

{99.28,172.71},{51.98,92.50},{61.09,97.87},{14.60,51.85}

};

Insert a table that shows running times for the original and MPI versions.



Write a short analysis of the results

The run time for this program was slightly weird. Given that it ran on 8 machines, you would think it would be quicker but instead it was slower. And the slowness of the program depended on what was being printed out. With the program printing out what was being sent and received, the program took an average of 0.79481213 seconds but commenting these lines out had the program run an average time of 0.37266584. While in real time it seems minimal, this is effectively showing a performance hit by printing out these lines.

# Verbose Repository Log

Paste your verbose format repository log here. With subversion this can be achieved by the following:

svn update

svn –v log > log.txt

gedit log.txt

Then select, copy and paste the text here