# POSIX

## 1.1 Password Cracking

### 1.1.1 Run the program 10 times and calculate the mean running time

|  |
| --- |
| 1. #include <stdio.h> 2. #include <string.h> 3. #include <stdlib.h> 4. #include <crypt.h> 5. #include <time.h> 7. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 8. \*\*\*\*\*\*\* 9. Demonstrates how to crack an encrypted password using a simple 10. "brute force" algorithm. Works on passwords that consist only of 2 11. uppercase 12. letters and a 2 digit integer. Your personalised data set is included 13. in the 14. code. 16. Compile with: 17. cc -o CrackAZ99-With-Data CrackAZ99-With-Data.c -lcrypt 19. If you want to analyse the results then use the redirection operator 20. to send 21. output to a file that you can view using an editor or the less 22. utility: 24. ./CrackAZ99-With-Data > results.txt 26. Name:1928720\_Adhikar\_chaudhary 27. @University of wolverhamption 29. \* Dr Kevan Buckley, University of Wolverhampton, 2018 30. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ 32. **int** n\_passwords = 4; 34. **char** \*encrypted\_passwords[] = { 36. "$6$KB$0G24VuNaA9ApVG4z8LkI/OOr9a54nBfzgQjbebhqBZxMHNg0HiYYf1Lx/HcGg6q1nnOSArPtZYbGy7yc5V.wP/", 37. "$6$KB$uTdNob3WgiJel6BANUV7Yc94ra1yxDGopfCSu4zRuiZwNHzqNbgTkbtFOXNZQWB0q5ItG9tr3iI4Fq5Cp5tTA.", 38. "$6$KB$Wq1dzyVEN2NNJvf6v0XFA/Nsx8HDDmwiX5RVNrPfPK6.ciNVJKEUG1urR0K2wrsDy2xQxbqrbmM5QmH.x6Im71", 39. "$6$KB$wbRHzUBWMCWXZDeiA0VAnCjZGiFEeK/RME1eHhYp5Uc3nGLJzrGrmeYYftsFLImL.8JMhFDbnRetMR7KLM0qh." 40. }; 42. /\*\* 43. Required by lack of standard function in C. 44. \*/ 46. **void** substr(**char** \*dest, **char** \*src, **int** start, **int** length){ 47. memcpy(dest, src + start, length); 48. \*(dest + length) = '\0'; 49. } 51. /\*\* 52. This function can crack the kind of password explained above. All 53. combinations 54. that are tried are displayed and when the password is found, #, is put 55. at the 56. start of the line. Note that one of the most time consuming operations 57. that 58. it performs is the output of intermediate results, so performance 59. experiments 60. for this kind of program should not include this. i.e. comment out the 61. printfs. 62. \*/ 64. **void** adhikar(**char** \*salt\_and\_encrypted){ 65. **int** a, b, h;     // Loop counters 66. **char** salt[7];    // String used in hahttps://www.youtube.com/watch?v=L8yJjIGleMwshing the password. Need space 67. **char** plain[7];   // The combination of letters currently being checked 68. **char** \*enc;       // Pointer to the encrypted password 69. **int** count = 0;   // The number of combinations explored so far 71. substr(salt, salt\_and\_encrypted, 0, 6); 73. **for**(a='A'; a<='Z'; a++){ 74. **for**(b='A'; b<='Z'; b++){ 75. **for**(h=0; h<=99; h++){ 76. sprintf(plain, "%c%c%02d", a, b, h); 77. enc = (**char** \*) crypt(plain, salt); 78. count++; 79. **if**(strcmp(salt\_and\_encrypted, enc) == 0){ 80. printf("#%-8d%s %s\n", count, plain, enc); 81. } **else** { 82. printf(" %-8d%s %s\n", count, plain, enc); 83. } 84. } 85. } 86. } 87. printf("%d solutions explored\n", count); 88. }  91. //Calculating time 93. **int** time\_difference(**struct** timespec \*start, **struct** timespec \*finish, **long** **long** **int** \*difference) 94. { 95. **long** **long** **int** ds =  finish->tv\_sec - start->tv\_sec; 96. **long** **long** **int** dn =  finish->tv\_nsec - start->tv\_nsec; 98. **if**(dn < 0 ) { 99. ds--; 100. dn += 1000000000; 101. } 102. \*difference = ds \* 1000000000 + dn; 103. **return** !(\*difference > 0); 104. } 105. **int** main(**int** argc, **char** \*argv[]) 106. { 107. **int** i; 108. **struct** timespec start, finish; 109. **long** **long** **int** time\_elapsed; 111. clock\_gettime(CLOCK\_MONOTONIC, &start); 113. **for**(i=0;i<n\_passwords;i<i++) 114. { 115. adhikar(encrypted\_passwords[i]); 116. } 117. clock\_gettime(CLOCK\_MONOTONIC, &finish); 118. time\_difference(&start, &finish, &time\_elapsed); 119. printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed, 120. (time\_elapsed/1.0e9)); 121. **return** 0; 122. } |

**Mean Running time**

|  |  |
| --- | --- |
| **Run time** | **Time taken in second** |
| 1 | 515.327416109 |
| 2 | 513.725913738 |
| 3 | 516.503597715 |
| 4 | 510.567181476 |
| 5 | 511.684660337 |
| 6 | 510.520288292 |
| 7 | 504.789551079 |
| 8 | 505.053580512 |
| 9 | 505.854465487 |
| 10 | 506.524673766 |
| Mean time | 510.0551 |

### 1.1.2 If the number of initials were increased to 3

In my perspective, we have initialize one numeric loop, for (h=0; h<=99; h++) and two upper case loop for (a='A'; a<='Z'; a++) & for (b='A'; b<='Z'; b++) to run this program it take 8.5009188808516658 Minutes. As per the question asked “ if the number of initials were increased to 3 “ We will add another loop and program looks like, for(a='A'; a<='Z'; a++), for(b='A'; b<='Z'; b++), for(c= 'C'; c<='C'; c++).If it has taken 8.5009188808516658 Minutes to run two number initialization, it will take 8.5009188808516658 Minutes \* 26 second more if the number of initials were increased to 3 because there is 26 range between A to Z.

### 1.1.3 Modify the program to crack the three-initials-two-digits

|  |
| --- |
| 1. #include <stdio.h> 2. #include <string.h> 3. #include <stdlib.h> 4. #include <crypt.h> 5. #include <time.h> 7. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 8. Demonstrates how to crack an encrypted password using a simple 9. "brute force" algorithm. Works on passwords that consist only of 2 uppercase 10. letters and a 2 digit integer. Your personalised data set is included in the 11. code. 13. Compile with: 14. cc -o CrackAZZ99-Crack-Pass CrackAZZ99-Crack-Pass.c -lcrypt 16. If you want to analyse the results then use the redirection operator to send 17. output to a file that you can view using an editor or the less utility: 19. ./CrackAZZ99-Crack-Pass | grep -a Time | awk '{print $4}'| sed 's/ns//'>Crack-Pass.csv 21. Name:1928720\_Adhikar\_chaudhary 22. @University of wolverhamption 23. Dr Kevan Buckley, University of Wolverhampton, 2018 24. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ 25. **int** n\_passwords = 4; 27. **char** \*encrypted\_passwords[] = { 28. // 29. "$6$KB$eDWVA04r/0UQ3PuPi8UY7KQrpTdQpaNyaFi3mtSeNTD6.6tw3WrRIs/XeP5ICj.wIUiELZVn674PTN2OB77pt1", 30. "$6$KB$Xzp3Fc1Vk4rHzhoJGnWKHQBSkkkYqH7FJ319SZ7LIEC7OlasEZpONEf//Gt8omZJRqQAiSlMo1ZlmBsMf2OR30", 31. "$6$KB$8.IFaaVMqyz9YR1/OHy5wX1bdTNPbyVcfN0g1QV8zRhJRcVesSOVdy0N2WMw4yv1Rbef5xISVFkyUapeXi7AT.", 32. "$6$KB$V4is2qePPOJNVGCJ1dr/GhZgfFkuRx1ZT48wHnQ9k97aXMagpf7gdR0xb9gu2qIdtW2r9MX.ARqJiOXYqCTQD1" 33. }; 35. /\*\* 36. Required by lack of standard function in C. 37. \*/ 39. **void** adhikar(**char** \*dest, **char** \*src, **int** start, **int** length){ 40. memcpy(dest, src + start, length); 41. \*(dest + length) = '\0'; 42. } 44. /\*\* 45. This function can crack the kind of password explained above. All combinations 46. that are tried are displayed and when the password is found, #, is put at the 47. start of the line. Note that one of the most time consuming operations that 48. it performs is the output of intermediate results, so performance experiments 49. for this kind of program should not include this. i.e. comment out the printfs. 50. \*/ 52. **void** crack(**char** \*salt\_and\_encrypted){ 53. **int** l,o,v,e;     // Loop counters 54. **char** salt[7];    // String used in hashing the password. Need space for \0 55. **char** plain[7];   // The combination of letters currently being checked 56. **char** \*enc;       // Pointer to the encrypted password 57. **int** count = 0;   // The number of combinations explored so far 59. adhikar(salt, salt\_and\_encrypted, 0, 6); 60. **for**(l='A'; l<='Z'; l++){ 61. **for**(o='A'; o<='Z'; o++){ 62. **for**(v='A'; v<='Z'; v++){ 63. **for**(e=0; e<=99; e++){ 64. sprintf(plain, "%c%c%c%02d", l,o,v,e); 65. enc = (**char** \*) crypt(plain, salt); 66. count++; 67. **if**(strcmp(salt\_and\_encrypted, enc) == 0){ 68. printf("#%-8d%s %s\n", count, plain, enc); 69. } **else** { 70. printf(" %-8d%s %s\n", count, plain, enc); 71. } 72. } 73. } 74. } 75. } 76. printf("%d solutions explored\n", count); 77. } 79. **int** time\_difference(**struct** timespec \*start, **struct** timespec \*finish, 80. **long** **long** **int** \*difference) { 81. **long** **long** **int** ds =  finish->tv\_sec - start->tv\_sec; 82. **long** **long** **int** dn =  finish->tv\_nsec - start->tv\_nsec;  85. **if**(dn < 0 ) { 86. ds--; 87. dn += 1000000000; 88. } 89. \*difference = ds \* 1000000000 + dn; 90. **return** !(\*difference > 0); 91. } 93. **int** main(**int** argc, **char** \*argv[]){ 94. **int** i; 95. **struct** timespec start, finish; 96. **long** **long** **int** time\_elapsed; 98. clock\_gettime(CLOCK\_MONOTONIC, &start);  101. **for**(i=0;i<n\_passwords;i<i++) { 102. crack(encrypted\_passwords[i]); 103. } 105. clock\_gettime(CLOCK\_MONOTONIC, &finish); 106. time\_difference(&start, &finish, &time\_elapsed); 107. printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed, 108. (time\_elapsed/1.0e9)); 110. **return** 0; 111. } |

### 1.1.4 Modify the original version of the program to run on 2 threads.

|  |
| --- |
| 1. #include <stdio.h> 2. #include <string.h> 3. #include <stdlib.h> 4. #include <crypt.h> 5. #include <time.h> 6. #include <pthread.h> 8. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 9. \*\*\*\*\*\*\* 10. Demonstrates how to crack an encrypted password using a simple 11. "brute force" algorithm. Works on passwords that consist only of 2 12. uppercase 13. letters and a 2 digit integer. Your personalised data set is included 14. in the 15. code. 17. Compile with: 18. cc -o possix\_thread possix\_thread.c -lcrypt -pthread 20. If you want to analyse the results then use the redirection operator 21. to send 22. output to a file that you can view using an editor or the less 23. utility: 25. ./possix\_thread > T\_results.txt 27. Dr Kevan Buckley, University of Wolverhampton, 2018 28. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 29. \*\*\*\*\*\*/ 30. **int** n\_passwords = 4; 32. **char** \*encrypted\_passwords[] = {  35. "$6$KB$0G24VuNaA9ApVG4z8LkI/OOr9a54nBfzgQjbebhqBZxMHNg0HiYYf1Lx/HcGg6q1nnOSArPtZYbGy7yc5V.wP/", 36. "$6$KB$uTdNob3WgiJel6BANUV7Yc94ra1yxDGopfCSu4zRuiZwNHzqNbgTkbtFOXNZQWB0q5ItG9tr3iI4Fq5Cp5tTA.", 37. "$6$KB$Wq1dzyVEN2NNJvf6v0XFA/Nsx8HDDmwiX5RVNrPfPK6.ciNVJKEUG1urR0K2wrsDy2xQxbqrbmM5QmH.x6Im71", 38. "$6$KB$wbRHzUBWMCWXZDeiA0VAnCjZGiFEeK/RME1eHhYp5Uc3nGLJzrGrmeYYftsFLImL.8JMhFDbnRetMR7KLM0qh." 40. }; 41. /\*\* 42. Required by lack of standard function in C. 43. \*/ 45. **void** adhikar(**char** \*dest, **char** \*src, **int** start, **int** length){ 46. memcpy(dest, src + start, length); 47. \*(dest + length) = '\0'; 48. } 50. /\*\* 51. This function can crack the kind of password explained above. All 52. combinations# 53. that are tried are displayed and when the password is found, #, is put 54. at the 55. start of the line. Note that one of the most time consuming operations 56. that 57. it performs is the output of intermediate results, so performance 58. experiments 59. for this kind of program should not include this. i.e. comment out the 60. printfs. 61. \*/ 63. **void** posix\_thread() 64. { 65. **int** j; 66. pthread\_t t1, t2; 68. **void** \*kernel\_function\_1(); 69. **void** \*kernel\_function\_2(); 71. **for**(j=0;j<n\_passwords;j<j++) { 72. pthread\_create(&t1, NULL,kernel\_function\_1, encrypted\_passwords[j]); 73. pthread\_create(&t2, NULL,kernel\_function\_2, encrypted\_passwords[j]); 75. pthread\_join(t1, NULL); 76. pthread\_join(t2, NULL); 77. } 78. } 80. **void** \*kernel\_function\_1(**char** \*salt\_and\_encrypted){ 81. **int** a, d, h;     // Loop counters 82. **char** salt[7];    // String used in hahttps://www.youtube.com/watch?v=L8yJjIGleMwshing the password. Need space 83. **char** plain[7];   // The combination of letters currently being checked 84. **char** \*enc;       // Pointer to the encrypted password 85. **int** count = 0;   // The number of combinations explored so far 87. adhikar(salt, salt\_and\_encrypted, 0, 6); 89. **for**(a='A'; a<='M'; a++){ 90. **for**(d='A'; d<='Z'; d++){ 91. **for**(h=0; h<=99;  h++){ 92. sprintf(plain, "%c%c%02d", a,d,h); 93. enc = (**char** \*) crypt(plain, salt); 94. count++; 95. **if**(strcmp(salt\_and\_encrypted, enc) == 0){ 96. printf("#%-8d%s %s\n", count, plain, enc); 97. } 98. **else** { 99. printf(" %-8d%s %s\n", count, plain, enc); 100. } 101. } 102. } 103. } 104. printf("%d solutions explored\n", count); 105. }  108. **void** \*kernel\_function\_2(**char** \*salt\_and\_encrypted){ 109. **int** n, e, p;     // Loop counters 110. **char** salt[7];    // String used in hahttps://www.youtube.com/watch?v=L8yJjIGleMwshing the password.      Need space 111. **char** plain[7];   // The combination of letters currently being checked 112. **char** \*enc;       // Pointer to the encrypted password 113. **int** count = 0;   // The number of combinations explored so far 115. adhikar(salt, salt\_and\_encrypted, 0, 6); 117. **for**(n='N'; n<='Z'; n++){ 118. **for**(e='A'; e<='Z'; e++){ 119. **for**(p=0; p<=99; p++){ 120. sprintf(plain, "%c%c%02d", n,e,p); 121. enc = (**char** \*) crypt(plain, salt); 122. count++; 123. **if**(strcmp(salt\_and\_encrypted, enc) == 0){ 124. printf("#%-8d%s %s\n", count, plain, enc); 125. } 126. **else** { 127. printf(" %-8d%s %s\n", count, plain, enc); 128. } 129. } 130. } 131. } 132. printf("%d solutions explored\n", count); 133. } 135. //Calculating time 137. **int** time\_difference(**struct** timespec \*start, **struct** timespec \*finish, **long** **long** **int** \*difference) 138. { 139. **long** **long** **int** ds =  finish->tv\_sec - start->tv\_sec; 140. **long** **long** **int** dn =  finish->tv\_nsec - start->tv\_nsec; 142. **if**(dn < 0 ) { 143. ds--; 144. dn += 1000000000; 145. } 146. \*difference = ds \* 1000000000 + dn; 147. **return** !(\*difference > 0); 148. } 149. **int** main(**int** argc, **char** \*argv[]) 150. { 152. **struct** timespec start, finish; 153. **long** **long** **int** time\_elapsed; 155. clock\_gettime(CLOCK\_MONOTONIC, &start);   159. posix\_thread(); 161. clock\_gettime(CLOCK\_MONOTONIC, &finish); 162. time\_difference(&start, &finish, &time\_elapsed); 163. printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed, 164. (time\_elapsed/1.0e9)); 165. **return** 0; 166. } |

### 1.1.5 Compare the results of the mean running time of the original program with the mean running time of the multithread version.

**By Using multithread program**

|  |  |
| --- | --- |
| Run Time | Time Taken in Nanosecond |
| 1 | 253364837088 |
| 2 | 254653411494 |
| 3 | 257242191080 |
| 4 | 266215610321 |
| 5 | 260193453845 |
| 6 | 269390773805 |
| 7 | 258632131485 |
| 8 | 258429777495 |
| 9 | 258139711870 |
| 10 | 260346290280 |

Meantime in nano second = 259660818876.3

Meantime in second =259.66081887630002711

Meantime in minute =4.3276803146050007

**Original Program**

Mean running time in nano second = 510055132851.1

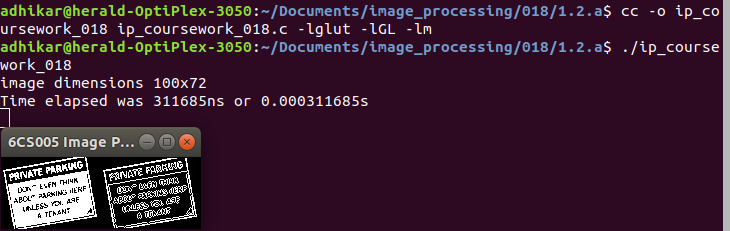
Mean running time in Second = 510.0551328511

Mean running time in minutes = 8.5009188808516658

Let us compare the original program and the program using multi threads. Original program without using multithread take 8.5009188808516658 Minutes and Program using multi thread take Mean time in minute is 4.3276803146050007. As a result, the average running time of multi-threads is almost half that of the original program. There are two loops, one loop search from A to M and the next one search from N to Z that reduce the time by using multi thread, because multi thread can work simultaneously. The kernel works independently and shares the same address space with other threads so that multithread program take less time then original program.

## 1.2 Image Processing using POSIX Threads

### 1.2.1 Run the program and capture the resulting image to put in your learning journal.



### 1.2.2 Implement a version of the edge detector that can process 4 pixels in parallel using multithread

|  |
| --- |
| 1. #include <stdio.h> 2. #include <stdlib.h> 3. #include <time.h> 4. #include <GL/glut.h> 5. #include <GL/gl.h> 6. #include <malloc.h> 7. #include <signal.h> 8. #include <pthread.h> 10. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 11. Displays two grey scale images. On the left is an image that has come from an 12. image processing pipeline, just after colour thresholding. On the right is 13. the result of applying an edge detection convolution operator to the left 14. image. This program performs that convolution. 16. Things to note: 17. - A single unsigned char stores a pixel intensity value. 0 is black, 256 is 18. white. 19. - The colour mode used is GL\_LUMINANCE. This uses a single number to 20. represent a pixel's intensity. In this case we want 256 shades of grey, 21. which is best stored in eight bits, so GL\_UNSIGNED\_BYTE is specified as 22. the pixel data type. 24. To compile adapt the code below wo match your filenames: 25. cc -o ip\_coursework\_018 ip\_coursework\_018.c -lglut -lGL -lm -pthread 27. Dr Kevan Buckley, University of Wolverhampton, 2018 28. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ 29. #define width 100 30. #define height 72 32. unsigned **char** image[], results[width \* height]; 34. **typedef** **struct** arguments 35. { 36. unsigned **char** \*input; 37. unsigned **char** \*output; 38. **int** start; 39. **int** stride;         //Stride is Logic memory which address gap between two successive pixels of an image 40. } 41. arguments\_t; 43. **void** 44. edges (unsigned **char** \*image, unsigned **char** \*results) 45. { 47. // defining t1,t2,t3,t4 are four threads 48. pthread\_t t1, t2, t3, t4; 49. arguments\_t t1\_arguments; 50. t1\_arguments.start = 0; 51. t1\_arguments.stride = 4; 52. t1\_arguments.input = image; 53. t1\_arguments.output = results;  56. arguments\_t t2\_arguments; 57. t2\_arguments.start = 1; 58. t2\_arguments.stride = 4; 59. t2\_arguments.input = image; 60. t2\_arguments.output = results; 62. arguments\_t t3\_arguments; 63. t3\_arguments.start = 2; 64. t3\_arguments.stride = 4; 65. t3\_arguments.input = image; 66. t3\_arguments.output = results; 68. arguments\_t t4\_arguments; 69. t4\_arguments.start = 3; 70. t4\_arguments.stride = 4; 71. t4\_arguments.input = image; 72. t4\_arguments.output = results; 74. **void** \*detect\_edges (); 76. //creating threads t1,t2,t3,t4 78. pthread\_create (&t1, NULL, detect\_edges, &t1\_arguments); 79. pthread\_create (&t2, NULL, detect\_edges, &t2\_arguments); 80. pthread\_create (&t3, NULL, detect\_edges, &t3\_arguments); 81. pthread\_create (&t4, NULL, detect\_edges, &t4\_arguments); 83. //joining four threads 85. pthread\_join (t1, NULL); 86. pthread\_join (t2, NULL); 87. pthread\_join (t3, NULL); 88. pthread\_join (t4, NULL); 89. }   93. **void** \* 94. detect\_edges (arguments\_t \* args) 95. { 96. **int** i; 97. unsigned **char** \*in = args->input; 98. unsigned **char** \*out = args->output; 99. **int** n\_pixels = width \* height; 101. **for** (i = args->start; i < n\_pixels; i += args->stride) 102. { 103. **int** x, y;           // the pixel of interest 104. **int** b, d, f, h;     // the pixels adjacent to x,y used for the calculation 105. **int** r;          // the result of calculate 107. y = i / width; 108. x = i - (width \* y); 110. **if** (x == 0 || y == 0 || x == width - 1 || y == height - 1) 111. { 112. results[i] = 0; 113. } 114. **else** 115. { 116. b = i + width; 117. d = i - 1; 118. f = i + 1; 119. h = i - width; 121. r = (in[i] \* 4) + (in[b] \* -1) + (in[d] \* -1) + (in[f] \* -1) 122. + (in[h] \* -1); 124. **if** (r > 0) 125. {   // if the result is positive this is an edge pixel 126. out[i] = 255; 127. } 128. **else** 129. { 130. out[i] = 0; 131. } 132. } 133. } 134. } 136. **int** 137. time\_difference (**struct** timespec \*start, **struct** timespec \*finish, 138. **long** **long** **int** \*difference) 139. { 140. **long** **long** **int** ds = finish->tv\_sec - start->tv\_sec; 141. **long** **long** **int** dn = finish->tv\_nsec - start->tv\_nsec; 143. **if** (dn < 0) 144. { 145. ds--; 146. dn += 1000000000; 147. } 148. \*difference = ds \* 1000000000 + dn; 149. **return** !(\*difference > 0); 150. } 152. **void** 153. tidy\_and\_exit () 154. { 155. exit (0); 156. } 158. **void** 159. sigint\_callback (**int** signal\_number) 160. { 161. printf ("\nInterrupt from keyboard\n"); 162. tidy\_and\_exit (); 163. } 165. **static** **void** 166. display () 167. { 168. glClear (GL\_COLOR\_BUFFER\_BIT); 169. glRasterPos4i (-1, -1, 0, 1); 170. glDrawPixels (width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, image); 171. glRasterPos4i (0, -1, 0, 1); 172. glDrawPixels (width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, results); 173. glFlush (); 174. } 176. **static** **void** 177. key\_pressed (unsigned **char** key, **int** x, **int** y) 178. { 179. **switch** (key) 180. { 181. **case** 27:            // escape 182. tidy\_and\_exit (); 183. **break**; 184. **default**: 185. printf ("\nPress escape to exit\n"); 186. **break**; 187. } 188. } 190. **int** 191. main (**int** argc, **char** \*\*argv) 192. { 193. signal (SIGINT, sigint\_callback); 195. printf ("image dimensions %dx%d\n", width, height); 197. **struct** timespec start, finish; 198. **long** **long** **int** time\_elapsed; 200. clock\_gettime (CLOCK\_MONOTONIC, &start); 202. edges (image, results); 204. clock\_gettime (CLOCK\_MONOTONIC, &finish); 205. time\_difference (&start, &finish, &time\_elapsed); 206. printf ("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed, 207. (time\_elapsed / 1.0e9)); 209. glutInit (&argc, argv); 210. glutInitWindowSize (width \* 2, height); 211. glutInitDisplayMode (GLUT\_SINGLE | GLUT\_LUMINANCE); 213. glutCreateWindow ("6CS005 Image Progessing Courework"); 214. glutDisplayFunc (display); 215. glutKeyboardFunc (key\_pressed); 216. glClearColor (0.0, 1.0, 0.0, 1.0); 218. glutMainLoop (); 220. tidy\_and\_exit (); 222. **return** 0; 223. } 224. unsigned **char** image[] = 225. {   0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 0, 0, 0, 255, 226. 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 227. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 228. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 229. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 230. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 231. 255, 0, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 232. 255, 233. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 234. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 235. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 236. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 237. 0, 0, 0, 0, 0, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 238. 0, 239. 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 240. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 241. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 242. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 243. 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 255, 255, 255, 255, 255, 255, 255, 244. 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 245. 255, 255, 246. 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 247. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 248. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 249. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 250. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 251. 255, 255, 255, 255, 252. 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 253. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 254. 0, 0, 0, 0, 0, 0, 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, 0, 0, 0, 0, 0, 0, 0, 256. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 257. 255, 255, 255, 255, 258. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 259. 255, 255, 260. 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 261. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 262. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 263. 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 264. 255, 255, 265. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 266. 255, 255, 255, 255, 267. 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 268. 0, 0, 269. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 270. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 271. 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 255, 255, 255, 255, 255, 255, 255, 272. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 273. 255, 255, 255, 255, 274. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 275. 0, 0, 0, 276. 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 277. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 278. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 255, 255, 279. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 280. 255, 255, 255, 255, 281. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 282. 255, 255, 255, 255, 283. 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 284. 255, 255, 285. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 286. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 287. 255, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 288. 255, 255, 255, 255, 289. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 290. 255, 255, 255, 255, 291. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 292. 255, 0, 0, 0, 293. 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 294. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 295. 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 296. 255, 255, 297. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 298. 255, 255, 255, 255, 299. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 300. 255, 255, 255, 255, 301. 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 302. 255, 255, 303. 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 304. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 305. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 306. 255, 255, 0, 255, 307. 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 308. 255, 255, 255, 255, 309. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 310. 255, 255, 255, 255, 311. 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 312. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 255, 255, 313. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 314. 255, 255, 255, 255, 315. 255, 255, 255, 0, 0, 0, 255, 255, 255, 255, 255, 0, 255, 255, 0, 0, 255, 316. 255, 255, 317. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 318. 255, 255, 255, 255, 319. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 320. 255, 255, 321. 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 322. 255, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 323. 255, 255, 255, 255, 324. 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 0, 255, 255, 255, 255, 255, 325. 0, 255, 255, 326. 0, 255, 255, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 327. 255, 255, 255, 328. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 329. 255, 255, 255, 255, 330. 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 331. 0, 0, 0, 0, 0, 255, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 332. 255, 255, 333. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 255, 334. 255, 255, 255, 335. 255, 255, 0, 255, 255, 0, 255, 255, 255, 0, 255, 0, 255, 0, 255, 255, 255, 336. 255, 255, 337. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 338. 255, 255, 255, 255, 339. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 0, 340. 0, 0, 0, 341. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 342. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 343. 255, 255, 255, 255, 344. 0, 255, 255, 255, 255, 255, 255, 0, 255, 255, 0, 255, 255, 255, 0, 0, 0, 345. 255, 255, 346. 0, 0, 0, 255, 0, 255, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 347. 255, 255, 348. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 255, 255, 349. 255, 255, 0, 255, 350. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 0, 255, 255, 351. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 352. 255, 255, 255, 255, 353. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 0, 255, 354. 255, 255, 355. 0, 0, 255, 255, 255, 0, 255, 0, 255, 0, 255, 0, 0, 255, 255, 0, 255, 255, 356. 255, 357. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 358. 255, 0, 255, 0, 359. 0, 255, 255, 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 360. 0, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 361. 255, 255, 255, 362. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 363. 255, 255, 255, 255, 364. 255, 0, 0, 0, 255, 0, 255, 255, 255, 255, 0, 0, 255, 255, 0, 0, 255, 0, 255, 365. 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 366. 255, 255, 255, 255, 367. 255, 255, 0, 0, 0, 0, 255, 255, 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 368. 0, 0, 0, 0, 0, 0, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 369. 0, 370. 0, 255, 0, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 371. 255, 255, 255, 372. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 373. 255, 0, 255, 255, 374. 0, 0, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 375. 255, 255, 255, 376. 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255, 0, 0, 377. 0, 0, 378. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 255, 255, 255, 255, 255, 255, 379. 255, 255, 255, 255, 0, 255, 255, 0, 0, 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 380. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 381. 255, 255, 255, 255, 382. 255, 255, 255, 255, 255, 0, 255, 255, 0, 255, 0, 0, 255, 255, 255, 255, 255, 383. 255, 255, 384. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 385. 255, 0, 255, 386. 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 387. 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 0, 0, 0, 255, 0, 388. 255, 0, 389. 255, 255, 255, 0, 255, 0, 255, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 390. 255, 255, 391. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 392. 0, 0, 0, 255, 393. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 394. 255, 255, 255, 255, 395. 0, 255, 255, 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 396. 0, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 397. 255, 0, 0, 398. 0, 0, 255, 255, 0, 255, 255, 255, 0, 255, 255, 255, 0, 255, 0, 255, 0, 0, 0, 399. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 400. 255, 255, 255, 255, 401. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 402. 255, 255, 255, 255, 403. 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 404. 0, 0, 0, 0, 0, 0, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 405. 255, 406. 255, 255, 255, 255, 0, 0, 255, 255, 255, 0, 255, 255, 255, 0, 0, 0, 255, 407. 255, 0, 408. 0, 0, 255, 255, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 409. 255, 255, 255, 410. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 411. 255, 255, 255, 255, 412. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 0, 413. 0, 0, 0, 414. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 0, 255, 255, 255, 255, 255, 255, 415. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 416. 255, 255, 255, 0, 417. 255, 255, 255, 0, 0, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255, 0, 255, 418. 255, 0, 419. 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 420. 255, 255, 255, 421. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 422. 255, 255, 255, 255, 423. 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 255, 424. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 425. 255, 255, 255, 255, 426. 255, 255, 255, 255, 0, 0, 0, 255, 0, 255, 255, 255, 0, 0, 255, 255, 255, 427. 255, 255, 428. 255, 0, 255, 255, 0, 255, 255, 0, 255, 0, 255, 0, 255, 255, 255, 255, 255, 429. 255, 255, 430. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 431. 255, 255, 255, 255, 432. 255, 255, 255, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 433. 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 434. 255, 255, 255, 255, 435. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 436. 255, 0, 255, 437. 255, 255, 255, 255, 255, 255, 0, 0, 255, 255, 255, 255, 255, 0, 0, 255, 255, 438. 255, 255, 439. 255, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 440. 255, 255, 255, 255, 441. 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 442. 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 0, 255, 255, 0, 255, 0, 0, 255, 443. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 444. 255, 255, 255, 255, 445. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 0, 0, 0, 255, 255, 446. 0, 0, 447. 0, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 0, 448. 255, 449. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 450. 255, 0, 0, 0, 451. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 0, 0, 0, 452. 0, 255, 0, 255, 0, 255, 255, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 453. 255, 255, 454. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 455. 255, 255, 255, 0, 456. 0, 0, 0, 0, 255, 0, 255, 255, 255, 255, 255, 255, 0, 255, 0, 255, 255, 255, 457. 255, 458. 0, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 459. 255, 255, 0, 460. 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 461. 255, 255, 255, 255, 0, 255, 255, 0, 255, 255, 0, 0, 255, 255, 0, 255, 0, 0, 462. 0, 463. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 464. 255, 255, 255, 255, 465. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 466. 255, 255, 255, 0, 467. 0, 255, 255, 0, 0, 0, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 468. 255, 255, 469. 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 470. 0, 255, 255, 0, 255, 255, 255, 255, 0, 0, 255, 255, 0, 0, 0, 255, 0, 255, 471. 255, 472. 255, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 473. 255, 255, 255, 474. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 475. 255, 255, 255, 255, 476. 255, 255, 255, 255, 0, 0, 255, 0, 255, 255, 0, 255, 0, 0, 0, 255, 255, 255, 477. 255, 478. 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 479. 0, 0, 0, 0, 0, 0, 255, 255, 0, 255, 255, 255, 255, 0, 255, 255, 255, 0, 255, 480. 0, 255, 0, 255, 255, 255, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 481. 255, 255, 482. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 483. 255, 255, 255, 255, 484. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 485. 255, 0, 255, 486. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 487. 255, 0, 0, 0, 488. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 255, 255, 255, 255, 255, 489. 255, 255, 255, 0, 0, 0, 255, 0, 255, 255, 0, 255, 0, 255, 255, 0, 255, 255, 490. 255, 491. 255, 255, 255, 255, 0, 255, 255, 0, 0, 255, 0, 0, 255, 255, 255, 255, 255, 492. 255, 255, 493. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 494. 255, 255, 255, 255, 495. 255, 255, 255, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 496. 255, 255, 0, 497. 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 498. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 255, 499. 255, 0, 255, 500. 255, 0, 255, 0, 255, 255, 255, 255, 255, 0, 0, 0, 255, 0, 0, 0, 255, 0, 255, 501. 255, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 502. 255, 255, 255, 503. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 504. 255, 255, 255, 255, 505. 255, 255, 255, 255, 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 506. 0, 0, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 507. 255, 255, 255, 508. 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255, 255, 255, 0, 509. 255, 255, 510. 0, 0, 255, 0, 0, 0, 255, 255, 0, 255, 0, 255, 0, 255, 0, 255, 255, 255, 255, 511. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 512. 255, 255, 255, 255, 513. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 514. 0, 515. 0, 0, 0, 0, 0, 0, 0, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 516. 255, 517. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 518. 255, 255, 255, 255, 519. 0, 0, 0, 255, 0, 0, 255, 255, 0, 255, 0, 255, 0, 0, 0, 0, 255, 0, 255, 520. 0, 255, 0, 0, 255, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 521. 255, 255, 522. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 523. 255, 0, 0, 0, 524. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 255, 255, 255, 255, 255, 525. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 526. 255, 255, 255, 255, 527. 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 255, 255, 0, 255, 255, 0, 528. 0, 0, 529. 0, 255, 255, 0, 255, 0, 0, 0, 0, 0, 255, 255, 255, 0, 255, 255, 255, 255, 530. 255, 531. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 532. 255, 255, 255, 255, 533. 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 534. 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 255, 255, 535. 255, 255, 255, 536. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 537. 255, 255, 255, 255, 538. 0, 0, 255, 0, 255, 255, 0, 255, 0, 255, 0, 0, 255, 0, 0, 255, 255, 0, 0, 539. 255, 255, 255, 0, 255, 0, 255, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 540. 255, 255, 541. 255, 255, 255, 255, 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 542. 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 0, 543. 255, 0, 544. 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 545. 255, 255, 255, 546. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 0, 547. 255, 255, 0, 548. 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 0, 255, 0, 255, 255, 255, 0, 549. 255, 255, 550. 0, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 551. 0, 0, 0, 0, 0, 0, 0, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 552. 255, 255, 0, 0, 255, 255, 255, 0, 0, 255, 255, 0, 255, 255, 255, 255, 255, 553. 255, 255, 554. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 555. 255, 255, 255, 255, 556. 255, 255, 255, 255, 0, 255, 0, 255, 0, 255, 255, 255, 255, 0, 255, 0, 255, 557. 0, 0, 558. 255, 255, 0, 255, 0, 0, 255, 0, 255, 255, 255, 255, 255, 255, 0, 255, 0, 0, 559. 0, 560. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 0, 255, 255, 255, 255, 255, 561. 255, 255, 255, 255, 0, 255, 255, 0, 0, 255, 255, 255, 0, 0, 255, 0, 0, 255, 562. 255, 563. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 564. 255, 255, 255, 255, 565. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 566. 255, 255, 255, 255, 567. 255, 0, 255, 0, 255, 255, 255, 0, 0, 0, 255, 255, 0, 255, 255, 255, 255, 568. 255, 255, 569. 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 570. 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 0, 255, 255, 255, 571. 255, 0, 0, 572. 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 573. 255, 255, 574. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 575. 255, 255, 255, 255, 576. 255, 255, 255, 255, 255, 255, 0, 255, 0, 255, 255, 255, 0, 255, 255, 0, 255, 577. 0, 0, 578. 0, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 579. 0, 0, 255, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 580. 255, 255, 581. 0, 255, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 582. 0, 255, 583. 255, 255, 255, 0, 0, 255, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 584. 255, 255, 255, 585. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 586. 0, 255, 0, 0, 587. 255, 0, 255, 0, 255, 255, 255, 255, 255, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 588. 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 589. 255, 590. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 591. 255, 255, 255, 255, 592. 255, 255, 255, 0, 0, 255, 255, 255, 0, 0, 255, 0, 0, 255, 255, 0, 255, 255, 593. 0, 594. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 595. 255, 255, 255, 255, 596. 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 255, 255, 255, 0, 255, 0, 597. 0, 0, 598. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 0, 255, 255, 255, 255, 255, 599. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 600. 255, 255, 255, 255, 601. 255, 0, 0, 0, 255, 255, 255, 255, 0, 255, 255, 255, 0, 255, 255, 255, 0, 0, 602. 255, 603. 255, 0, 255, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 604. 255, 255, 255, 605. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 606. 255, 255, 255, 255, 607. 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 0, 608. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 609. 255, 255, 255, 255, 610. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 611. 255, 255, 0, 255, 612. 255, 255, 0, 0, 0, 255, 0, 255, 0, 0, 255, 255, 255, 255, 255, 255, 255, 0, 613. 255, 614. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 615. 255, 255, 255, 255, 616. 255, 255, 255, 255, 255, 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 617. 0, 0, 0, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 618. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 619. 255, 255, 255, 255, 620. 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 255, 0, 0, 0, 0, 255, 255, 621. 255, 255, 622. 0, 255, 255, 0, 255, 255, 0, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 623. 255, 255, 624. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 625. 0, 0, 626. 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 627. 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 628. 255, 255, 255, 629. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 255, 630. 0, 0, 255, 631. 0, 255, 255, 255, 255, 0, 255, 255, 0, 0, 0, 0, 255, 0, 255, 255, 255, 0, 632. 255, 633. 0, 255, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 634. 255, 0, 0, 635. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 0, 0, 0, 0, 636. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 637. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 638. 255, 255, 255, 255, 639. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 0, 255, 255, 640. 0, 255, 0, 641. 255, 0, 0, 0, 255, 0, 255, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 642. 255, 255, 643. 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 644. 0, 0, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 645. 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 646. 255, 255, 647. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 648. 0, 0, 0, 0, 649. 255, 255, 0, 255, 0, 255, 0, 255, 0, 255, 0, 0, 0, 255, 255, 255, 255, 255, 650. 255, 651. 255, 255, 255, 255, 255, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 652. 0, 0, 0, 255, 0, 0, 0, 0, 255, 0, 0, 0, 0, 255, 255, 0, 0, 0, 0, 653. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 654. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 655. 255, 255, 255, 255, 656. 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 0, 255, 0, 255, 0, 0, 657. 0, 255, 658. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 0, 0, 0, 0, 0, 0, 659. 0, 660. 0, 0, 0, 0, 0, 0, 0, 0, 255, 0, 0, 0, 0, 255, 0, 0, 0, 0, 255, 661. 255, 0, 255, 255, 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 662. 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 663. 255, 664. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 665. 255, 255, 255, 255, 666. 255, 0, 255, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 667. 255, 0, 0, 668. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 0, 0, 0, 0, 255, 669. 255, 255, 0, 0, 255, 0, 255, 255, 0, 0, 255, 0, 0, 0, 255, 255, 0, 255, 255, 670. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 671. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 672. 255, 255, 255, 255, 673. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 674. 255, 255, 255, 255, 675. 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 676. 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 0, 677. 0, 255, 678. 255, 255, 0, 255, 255, 0, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 679. 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 680. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 681. 255, 255, 255, 255, 682. 255, 255, 255, 255, 255, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 683. 0, 0, 0, 255, 255, 0, 0, 0, 255, 255, 0, 255, 255, 255, 255, 255, 255, 255, 684. 0, 685. 255, 255, 0, 0, 255, 255, 255, 0, 255, 255, 255, 255, 255, 0, 0, 255, 0, 0, 686. 0, 687. 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 688. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 689. 255, 255, 255, 255, 690. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 0, 0, 0, 0, 691. 0, 0, 692. 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 0, 0, 255, 255, 255, 255, 255, 255, 693. 255, 0, 255, 255, 0, 255, 255, 0, 255, 255, 0, 255, 0, 255, 255, 255, 255, 694. 255, 0, 695. 0, 255, 0, 0, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 696. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 697. 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 698. 0, 255, 0, 0, 699. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 0, 0, 0, 0, 0, 700. 0, 255, 255, 0, 255, 255, 255, 255, 255, 0, 255, 255, 0, 255, 0, 0, 255, 0, 701. 255, 702. 255, 255, 255, 0, 0, 255, 255, 0, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 703. 0, 0, 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 704. 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 705. 255, 255, 255, 706. 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 707. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 0, 255, 255, 255, 255, 708. 0, 0, 255, 0, 255, 255, 255, 255, 0, 0, 255, 255, 0, 0, 255, 255, 0, 0, 0, 709. 0, 0, 0, 255, 255, 0, 0, 0, 255, 0, 0, 0, 255, 0, 255, 0, 0, 0, 0, 710. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 711. 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 712. 0, 0, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 713. 0, 0, 0, 255, 0, 0, 0, 255, 0, 0, 255, 255, 0, 0, 0, 255, 255, 0, 0, 714. 255, 255, 255, 255, 0, 0, 0, 0, 255, 255, 0, 0, 0, 255, 255, 255, 255, 255, 715. 0, 716. 255, 255, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 717. 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 718. 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 719. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 255, 720. 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 0, 721. 255, 722. 255, 255, 255, 0, 255, 255, 0, 255, 255, 255, 0, 255, 0, 0, 255, 255, 0, 0, 723. 0, 724. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 725. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 726. 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 727. 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 255, 728. 0, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 255, 255, 255, 0, 255, 729. 255, 0, 0, 730. 255, 255, 0, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 731. 0, 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 732. 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 733. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 734. 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 255, 255, 255, 735. 255, 255, 736. 0, 255, 255, 255, 255, 255, 0, 0, 255, 255, 255, 255, 0, 0, 255, 255, 0, 0, 737. 255, 738. 255, 0, 0, 0, 0, 0, 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 739. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 740. 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 741. 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 0, 255, 255, 255, 0, 0, 742. 255, 255, 0, 255, 255, 0, 255, 255, 255, 255, 255, 0, 255, 255, 255, 255, 743. 255, 0, 255, 744. 255, 255, 0, 255, 255, 255, 255, 0, 0, 0, 0, 0, 255, 0, 0, 0, 0, 0, 0, 745. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 746. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 747. 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 748. 255, 255, 255, 0, 0, 255, 255, 255, 255, 255, 0, 255, 255, 255, 255, 0, 0, 749. 255, 255, 750. 0, 255, 255, 255, 255, 255, 0, 255, 255, 0, 0, 255, 255, 0, 0, 0, 255, 255, 751. 0, 752. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 753. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 754. 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 755. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 0, 0, 255, 255, 255, 756. 255, 0, 0, 255, 255, 0, 255, 255, 255, 255, 255, 0, 255, 255, 0, 255, 255, 757. 255, 0, 758. 0, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 759. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 760. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 761. 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 762. 0, 0, 255, 0, 0, 255, 255, 0, 255, 255, 0, 255, 255, 255, 255, 255, 0, 255, 763. 255, 764. 0, 255, 255, 255, 0, 0, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 765. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 766. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 767. 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 768. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 255, 255, 0, 769. 255, 255, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 0, 0, 0, 0, 770. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 771. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 772. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 773. 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 774. 0, 0, 0, 0, 0, 255, 255, 0, 255, 255, 255, 255, 255, 0, 0, 0, 0, 255, 0, 775. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 776. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 777. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 778. 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 779. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 0, 0, 780. 0, 0, 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 781. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 782. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 783. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 784. 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 785. 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 786. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 787. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 788. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 789. 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 790. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 0, 0, 0, 0, 0, 791. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 792. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 793. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 794. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 795. 255, 255, 255, 255, 255, 255, 255, 255, 255, 0, 0, 0, 0, 0, 0, 0, 0, 255, 796. 255, 797. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 798. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 799. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 800. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 801. 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 255, 255, 255, 255, 255, 255, 255, 255, 255, 802. 255, 255, 0, 255, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 803. }; |

### 1.2.3 Compare the relative running times of the original edge detection program, with the multithread one.

|  |  |  |
| --- | --- | --- |
| **Run time** | **Original program time ns** | **Multithread program ns** |
| 1 | 73251 | 342517 |
| 2 | 61698 | 469085 |
| 3 | 62160 | 203127 |
| 4 | 61606 | 241242 |
| 5 | 66720 | 346862 |
| 6 | 299407 | 193769 |
| 7 | 61957 | 254527 |
| 8 | 62064 | 1598401 |
| 9 | 62564 | 196090 |
| 10 | 91841 | 247459 |
| **Mean Time in NS** | 90326.8 | 409307.9 |

**Original program**

Mean time in Nanosecond = 90326.8

Mean time in second = 9.03268

Mean time in Minutes = 1.50544667

**Multithread program**

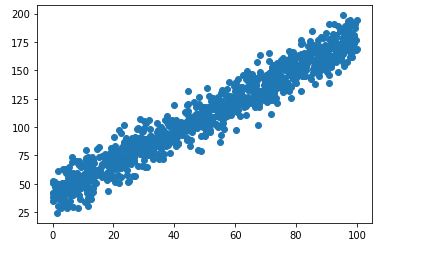
Mean time in Nanosecond = 409307.9

Mean time in second = 0.0004093079

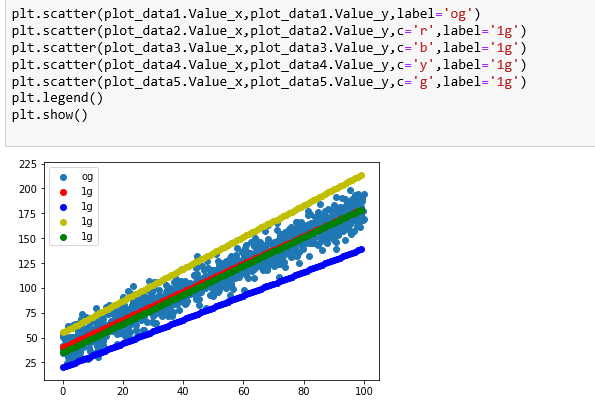
Mean time in Minutes = 0.00000682179833333347

## 1.3 Linear regression

### 1.3.1Do a scatter plot of your dataset and put it in your portfolio.

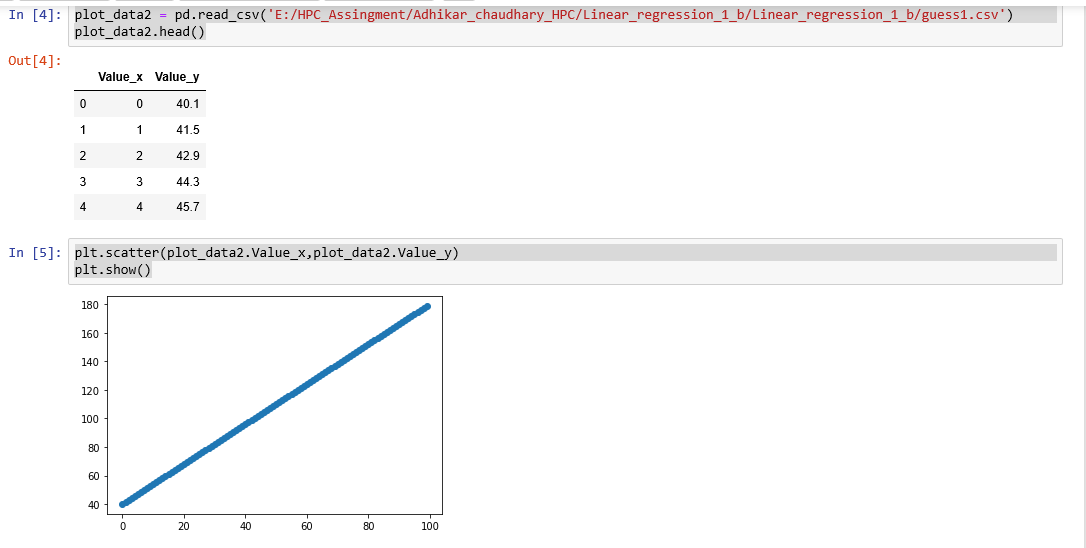


### 1.3.2 Have three guesses at the optimum values for m and c and plot them

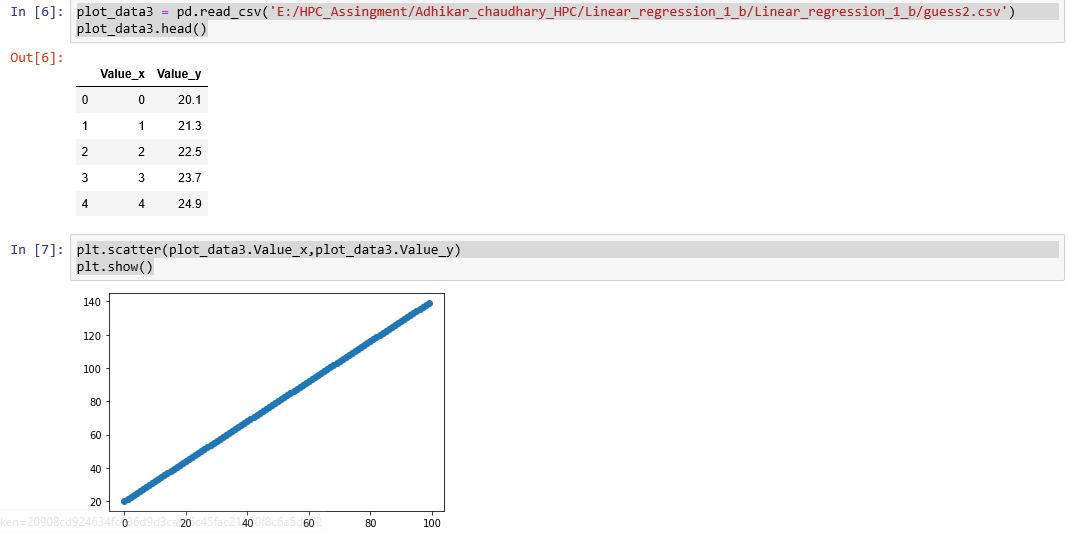


### 1.3.3 Run the program to see what solution it finds. Overlay the line that was found by the program on toa dataset scatter plot and comment on the solution.

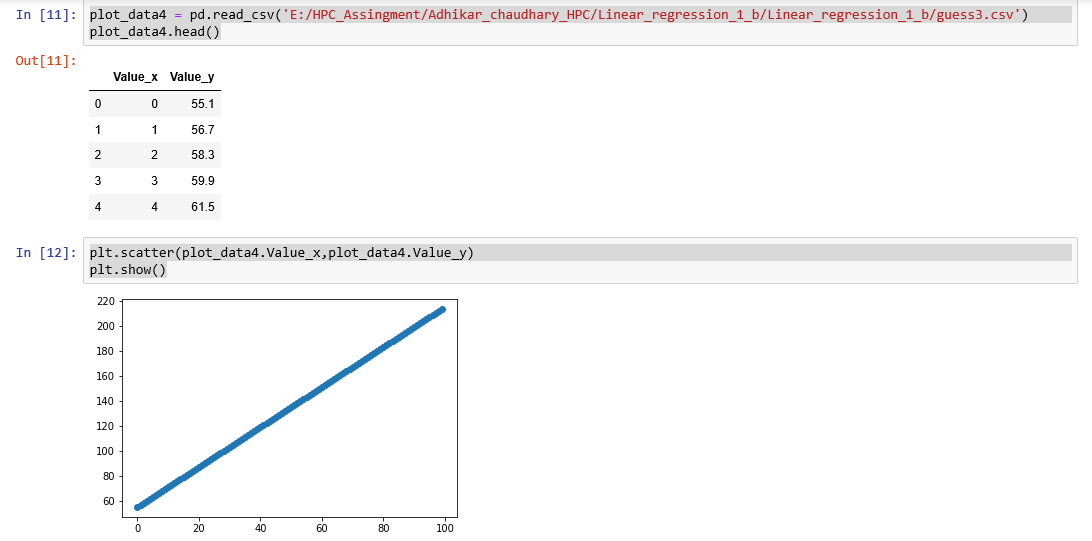
Guess 1



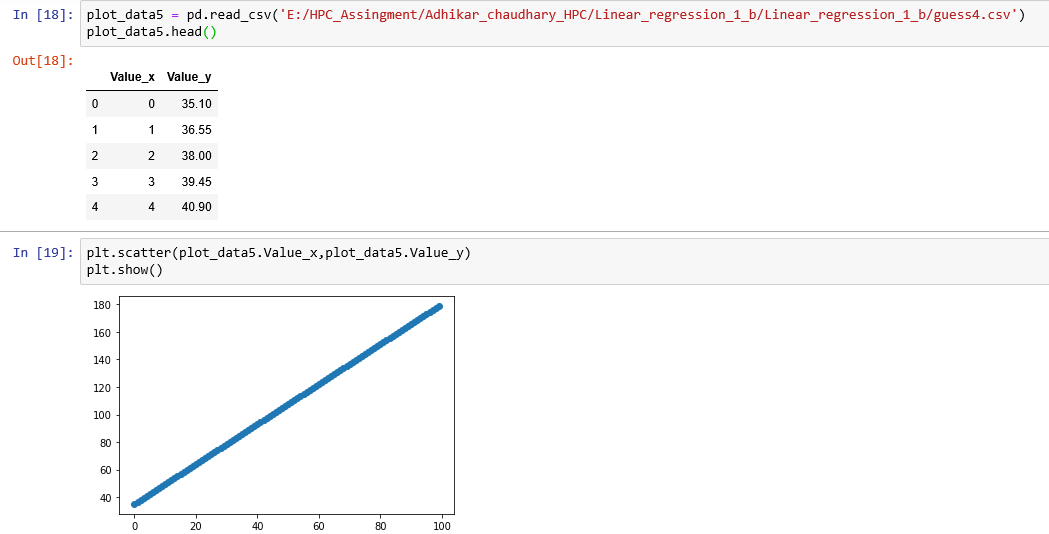
Guess 2



Guess 3



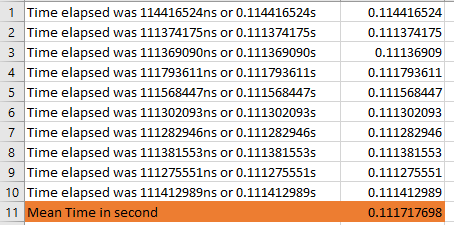
Original but I have kept values as a guess 4



### 1.3.4 Remove any extraneous printf statements from the program and find its mean running time.

|  |
| --- |
| 1. #include <stdio.h> 2. #include <math.h> 3. #include <time.h>  6. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 7. \* This program takes an initial estimate of m and c and finds the associated 8. \* rms error. It is then as a base to generate and evaluate 8 new estimates, 9. \* which are steps in different directions in m-c space. The best estimate is 10. \* then used as the base for another iteration of "generate and evaluate". This 11. \* continues until none of the new estimates are better than the base. This is 12. \* a gradient search for a minimum in mc-space. 13. \* 14. \* To compile: 15. \*   cc -o Lr18 Lr18.c -lm 16. \* 17. \* To run: 18. \*   ./Lr18 19. \* Name:1928720\_Adhikar\_chaudhary @University of wolverhamption 20. \* Dr Kevan Buckley, University of Wolverhampton, 2018 21. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ 23. **typedef** **struct** point\_t { 24. **double** x; 25. **double** y; 26. } point\_t; 28. **int** n\_data = 1000; 29. point\_t data[]; 31. **double** residual\_error(**double** x, **double** y, **double** m, **double** c) { 32. **double** e = (m \* x) + c - y; 33. **return** e \* e; 34. } 36. **double** rms\_error(**double** m, **double** c) { 37. **int** i; 38. **double** mean; 39. **double** error\_sum = 0; 41. **for**(i=0; i<n\_data; i++) { 42. error\_sum += residual\_error(data[i].x, data[i].y, m, c); 43. } 45. mean = error\_sum / n\_data; 47. **return** sqrt(mean); 48. } 49. **int** time\_difference(**struct** timespec \*start, **struct** timespec \*finish, 50. **long** **long** **int** \*difference) { 51. **long** **long** **int** ds =  finish->tv\_sec - start->tv\_sec; 52. **long** **long** **int** dn =  finish->tv\_nsec - start->tv\_nsec; 54. **if**(dn < 0 ) { 55. ds--; 56. dn += 1000000000; 57. } 58. \*difference = ds \* 1000000000 + dn; 59. **return** !(\*difference > 0); 60. } 62. **int** main() { 63. **int** i; 64. **double** bm = 1.3; 65. **double** bc = 10; 66. **double** be; 67. **double** dm[8]; 68. **double** dc[8]; 69. **double** e[8]; 70. **double** step = 0.01; 71. **double** best\_error = 999999999; 72. **int** best\_error\_i; 73. **int** minimum\_found = 0; 75. **struct** timespec start, finish; 76. **long** **long** **int** time\_elapsed; 77. clock\_gettime(CLOCK\_MONOTONIC, &start); 79. **double** om[] = {0,1,1, 1, 0,-1,-1,-1}; 80. **double** oc[] = {1,1,0,-1,-1,-1, 0, 1}; 82. be = rms\_error(bm, bc); 84. **while**(!minimum\_found) { 85. **for**(i=0;i<8;i++) { 86. dm[i] = bm + (om[i] \* step); 87. dc[i] = bc + (oc[i] \* step); 88. } 90. **for**(i=0;i<8;i++) { 91. e[i] = rms\_error(dm[i], dc[i]); 92. **if**(e[i] < best\_error) { 93. best\_error = e[i]; 94. best\_error\_i = i; 95. } 96. }  99. **if**(best\_error < be) { 100. be = best\_error; 101. bm = dm[best\_error\_i]; 102. bc = dc[best\_error\_i]; 103. } **else** { 104. minimum\_found = 1; 105. } 106. } 107. printf("minimum m,c is %lf,%lf with error %lf\n", bm, bc, be); 108. clock\_gettime(CLOCK\_MONOTONIC, &finish); 109. time\_difference(&start, &finish, &time\_elapsed); 110. printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed, 111. (time\_elapsed/1.0e9)); 113. **return** 0; 114. } 116. point\_t data[] = { 117. {87.72,160.36},{79.77,154.56},{74.88,144.94},{67.05,157.02}, 118. {73.75,121.36},{69.58,132.51},{67.21,143.57},{72.13,141.31}, 119. {87.64,144.77},{65.42,146.46},{83.09,144.82},{73.30,137.10}, 120. {76.75,151.67},{69.17,144.50},{87.61,171.62},{83.56,164.97}, 121. {75.57,137.55},{73.80,156.25},{75.75,130.90},{97.13,193.14}, 122. { 6.54,74.00},{54.31,100.05},{57.36,127.37},{93.54,172.18}, 123. {98.26,190.44},{87.27,149.69},{66.44,143.27},{98.24,171.56}, 124. {27.74,98.27},{10.51,57.95},{88.17,179.14},{22.66,63.64}, 125. {30.46,64.81},{65.24,135.15},{ 1.71,39.30},{91.29,171.76}, 126. {77.43,153.66},{68.29,132.07},{33.43,82.08},{60.40,119.17}, 127. {15.86,64.98},{61.96,126.18},{11.05,58.97},{23.99,67.74}, 128. {21.13,65.78},{34.79,97.87},{22.32,74.86},{78.29,136.48}, 129. {25.38,84.45},{94.49,169.91},{35.62,99.82},{58.20,127.53}, 130. {98.37,182.22},{93.79,168.74},{26.98,91.60},{30.55,103.77}, 131. { 5.86,44.79},{78.96,159.17},{93.19,158.82},{24.73,51.92}, 132. {94.91,186.12},{61.84,135.07},{ 2.60,37.48},{95.93,189.32}, 133. {95.39,157.13},{20.24,69.22},{29.93,71.36},{68.29,148.83}, 134. {36.68,92.16},{93.29,180.89},{30.53,75.37},{48.10,120.04}, 135. {83.17,153.60},{ 8.22,50.63},{ 8.76,50.80},{97.71,171.61}, 136. {13.28,46.71},{ 1.07,43.74},{11.16,71.21},{20.98,78.43}, 137. {21.89,50.58},{11.51,55.46},{ 3.22,43.94},{41.33,86.40}, 138. { 9.72,66.88},{10.91,51.56},{40.75,100.96},{81.92,161.87}, 139. {54.44,112.70},{11.35,62.04},{69.49,123.86},{50.51,96.47}, 140. {74.84,150.85},{77.35,150.52},{81.72,163.59},{11.65,37.53}, 141. {13.48,52.63},{79.02,166.24},{84.93,171.49},{59.13,117.54}, 142. {70.14,129.94},{58.90,118.01},{28.64,107.03},{43.71,112.86}, 143. {58.86,114.54},{78.67,160.78},{94.74,171.99},{70.31,122.40}, 144. {16.20,74.80},{58.59,110.98},{70.81,130.09},{ 7.86,46.69}, 145. {91.46,153.64},{34.05,87.81},{24.06,68.39},{66.23,134.24}, 146. {58.69,121.12},{ 0.18,38.19},{20.48,63.34},{63.79,125.30}, 147. {27.38,80.17},{39.12,100.73},{85.45,151.11},{48.89,92.37}, 148. {81.54,155.19},{20.52,55.22},{16.33,56.39},{82.69,158.63}, 149. {78.03,160.98},{ 3.36,62.70},{ 9.16,36.34},{ 5.56,50.25}, 150. {96.94,185.14},{46.54,107.56},{13.32,54.76},{21.63,58.16}, 151. {35.36,86.17},{92.63,175.36},{35.77,93.56},{24.99,73.63}, 152. {78.27,153.33},{44.20,89.01},{ 8.41,28.53},{22.39,65.89}, 153. { 4.01,63.33},{61.57,131.79},{39.85,95.92},{76.59,132.38}, 154. {49.60,117.48},{54.00,114.01},{92.20,160.82},{98.73,180.85}, 155. {56.18,108.40},{75.10,144.54},{36.91,105.85},{36.52,75.99}, 156. {47.65,79.93},{72.52,151.63},{50.78,112.17},{85.76,151.27}, 157. {14.41,51.07},{48.66,109.44},{50.39,104.20},{95.63,174.36}, 158. {69.74,140.10},{25.53,87.75},{67.15,141.26},{55.44,129.77}, 159. {20.44,65.95},{98.96,181.37},{47.04,111.25},{58.45,117.28}, 160. {57.79,123.58},{83.84,171.31},{61.97,131.09},{37.66,109.35}, 161. {65.32,124.56},{32.31,92.17},{64.08,140.31},{53.53,119.28}, 162. {46.02,102.80},{ 8.19,46.99},{75.24,134.02},{11.54,67.14}, 163. {37.35,94.89},{17.90,73.66},{68.74,127.84},{35.61,74.80}, 164. {12.89,61.03},{51.57,127.06},{69.44,137.16},{55.00,87.07}, 165. { 9.46,66.57},{ 7.32,50.65},{18.69,60.81},{91.80,157.00}, 166. {23.13,80.31},{79.77,160.31},{25.92,67.57},{88.84,157.95}, 167. {59.62,122.91},{20.02,74.74},{66.18,130.20},{22.33,67.57}, 168. {34.98,72.13},{27.88,80.57},{74.11,152.93},{63.45,123.91}, 169. {34.17,76.27},{59.99,137.07},{26.78,70.38},{30.99,79.08}, 170. { 3.26,42.81},{51.85,97.62},{38.00,89.05},{69.57,128.69}, 171. {55.77,102.44},{38.64,95.04},{31.94,80.16},{33.57,84.21}, 172. {49.34,115.16},{60.37,126.82},{48.79,79.20},{53.62,101.98}, 173. {44.49,120.58},{40.72,92.32},{62.30,134.92},{86.71,167.45}, 174. {28.92,90.49},{ 3.64,50.38},{ 2.98,51.03},{ 7.72,50.72}, 175. {36.01,88.98},{12.80,53.33},{18.18,59.93},{ 8.74,39.72}, 176. {93.05,157.51},{78.51,157.36},{ 9.32,51.91},{28.43,83.25}, 177. { 5.63,57.81},{ 5.75,49.78},{33.39,89.81},{98.06,183.34}, 178. {61.33,124.19},{90.28,167.83},{97.93,180.90},{24.74,74.19}, 179. {28.42,90.02},{67.13,126.96},{29.53,65.54},{28.34,82.58}, 180. { 5.75,35.03},{36.57,89.60},{18.70,68.23},{85.61,147.33}, 181. {90.19,148.59},{29.44,82.07},{95.34,164.13},{25.79,76.89}, 182. {18.18,55.48},{47.36,99.63},{ 9.34,53.88},{95.16,174.51}, 183. {79.94,148.24},{ 1.82,30.86},{75.83,138.77},{15.10,60.29}, 184. {34.45,91.55},{ 0.41,39.02},{12.90,64.51},{75.66,161.10}, 185. {40.13,85.95},{57.37,109.43},{79.64,155.61},{25.54,76.82}, 186. {33.13,80.14},{31.55,75.12},{ 0.93,51.83},{36.40,97.82}, 187. {55.60,112.28},{ 1.26,47.44},{73.54,125.72},{49.81,102.51}, 188. {97.31,170.44},{89.64,169.02},{91.89,169.61},{45.89,107.66}, 189. {99.90,168.98},{39.27,106.49},{63.61,118.52},{60.86,115.28}, 190. { 3.11,48.70},{61.64,119.97},{28.18,81.53},{ 5.20,43.98}, 191. {51.76,117.25},{57.63,127.76},{78.84,153.80},{10.54,48.23}, 192. {52.51,115.61},{ 4.69,28.30},{93.93,164.59},{69.70,141.08}, 193. {57.13,118.40},{23.45,86.44},{44.59,132.03},{31.91,98.03}, 194. {44.44,103.85},{ 9.46,53.83},{92.57,190.48},{36.29,95.15}, 195. {32.06,92.48},{86.16,138.27},{49.86,112.94},{96.14,181.82}, 196. {16.05,59.03},{52.13,102.89},{51.27,109.44},{67.94,117.86}, 197. {86.77,158.44},{74.20,143.83},{39.13,93.09},{66.00,137.66}, 198. {22.48,55.93},{54.90,116.64},{51.71,95.82},{36.07,85.96}, 199. { 5.38,59.94},{84.49,160.75},{28.93,82.85},{89.92,183.08}, 200. { 1.83,61.19},{26.71,73.35},{49.96,93.82},{13.56,68.73}, 201. {26.93,70.29},{85.19,165.22},{74.31,148.26},{44.90,107.17}, 202. {81.60,159.66},{86.68,150.41},{ 8.77,66.43},{75.18,159.06}, 203. { 8.86,45.78},{66.61,131.48},{ 5.80,40.46},{84.73,169.37}, 204. {35.34,85.87},{97.62,161.97},{15.22,73.73},{77.52,152.32}, 205. { 2.96,45.12},{60.66,127.28},{66.50,131.20},{72.85,141.01}, 206. {65.90,130.75},{43.44,101.16},{11.06,52.18},{14.77,62.27}, 207. {77.52,159.43},{47.99,126.74},{63.43,110.36},{50.26,113.37}, 208. {95.43,198.57},{24.53,57.71},{12.87,54.28},{63.79,135.32}, 209. {56.58,110.85},{59.10,121.23},{99.99,194.48},{34.56,83.22}, 210. {98.83,173.25},{ 7.73,70.43},{34.44,103.18},{96.37,169.25}, 211. {50.19,118.06},{84.66,175.92},{79.01,142.91},{99.12,174.11}, 212. {61.97,120.41},{81.93,146.96},{36.18,71.82},{ 3.47,51.04}, 213. { 0.07,52.79},{82.02,158.42},{37.02,94.32},{26.83,77.25}, 214. {87.90,166.44},{22.65,76.05},{ 8.80,53.89},{53.12,109.05}, 215. {64.69,147.19},{55.56,107.54},{98.42,175.01},{ 4.99,30.66}, 216. {63.23,122.97},{56.44,121.49},{ 8.58,69.79},{79.38,149.69}, 217. {55.64,122.77},{84.76,142.77},{29.21,85.68},{18.11,70.14}, 218. {33.15,85.38},{11.30,44.68},{83.67,163.51},{23.43,101.91}, 219. {29.59,81.85},{19.90,75.18},{26.05,89.95},{61.05,129.50}, 220. { 6.27,68.93},{96.95,191.69},{82.78,145.43},{73.84,136.65}, 221. {60.44,140.08},{12.67,65.86},{ 2.20,36.68},{ 6.16,54.50}, 222. {35.28,95.48},{83.01,152.33},{64.33,121.41},{91.72,162.34}, 223. {72.62,130.77},{55.31,121.78},{83.52,164.02},{ 1.45,24.37}, 224. {84.96,164.70},{ 6.85,59.62},{89.95,158.91},{57.41,133.29}, 225. { 6.13,58.72},{78.33,125.38},{65.15,121.67},{19.10,81.33}, 226. {17.43,58.99},{60.92,142.74},{69.37,142.70},{ 7.71,53.52}, 227. {38.81,101.59},{18.31,65.35},{41.05,90.89},{29.84,96.28}, 228. {27.65,88.04},{11.74,30.21},{72.14,150.80},{59.92,123.34}, 229. {20.59,51.65},{73.54,147.09},{25.04,52.81},{21.44,78.92}, 230. { 8.30,68.18},{38.27,103.50},{76.73,135.46},{13.41,42.84}, 231. { 9.77,50.17},{31.79,84.64},{11.63,47.87},{81.10,154.34}, 232. {32.88,86.39},{83.66,156.16},{42.84,101.97},{92.23,181.69}, 233. {56.62,128.67},{21.57,72.57},{28.42,76.81},{78.49,151.43}, 234. {34.76,87.12},{95.65,153.86},{48.99,114.03},{22.97,82.24}, 235. {96.82,167.98},{55.42,93.17},{59.22,121.38},{41.66,101.95}, 236. {77.91,166.28},{12.14,54.69},{91.24,171.96},{46.22,106.58}, 237. {98.54,161.56},{46.98,104.41},{60.09,131.79},{67.83,119.99}, 238. {41.09,85.10},{70.10,135.73},{36.99,91.78},{10.72,53.99}, 239. {50.06,91.80},{18.39,60.80},{26.53,85.06},{76.65,154.26}, 240. { 0.02,34.66},{37.56,104.13},{91.48,177.82},{ 0.31,41.81}, 241. {93.20,166.21},{ 6.94,43.44},{85.38,159.15},{90.74,139.02}, 242. {83.80,157.57},{39.34,92.96},{95.49,170.28},{77.55,145.57}, 243. {33.05,88.61},{ 4.28,43.15},{95.89,183.02},{36.18,97.42}, 244. {94.88,160.54},{ 5.18,55.47},{24.04,66.69},{50.93,118.02}, 245. {11.82,58.02},{ 4.55,62.84},{71.87,136.43},{ 1.64,46.82}, 246. {86.64,157.08},{76.78,135.11},{17.94,59.70},{41.81,96.43}, 247. {89.21,170.11},{75.89,139.66},{90.73,160.27},{45.39,101.75}, 248. {61.07,135.71},{ 7.53,48.91},{26.32,74.50},{89.44,164.11}, 249. {79.85,162.30},{55.91,104.02},{89.47,172.39},{88.09,154.40}, 250. {87.63,148.99},{24.43,74.77},{49.28,103.08},{86.49,165.82}, 251. {93.79,148.29},{93.25,170.91},{59.55,126.98},{ 9.24,64.46}, 252. {73.82,134.35},{76.07,152.94},{77.44,148.98},{87.01,161.01}, 253. {72.52,154.33},{21.77,97.29},{47.74,111.47},{17.34,68.45}, 254. {39.75,119.04},{84.78,160.15},{62.57,121.76},{70.20,146.15}, 255. { 7.08,50.89},{60.33,97.44},{29.90,89.84},{41.82,95.48}, 256. {38.27,101.32},{ 9.60,53.87},{84.69,166.21},{97.64,177.02}, 257. {73.96,145.64},{11.68,73.29},{31.64,75.20},{44.12,119.54}, 258. {29.91,99.48},{62.85,117.28},{65.55,123.70},{78.66,161.94}, 259. {71.06,158.16},{71.17,147.50},{12.49,63.11},{62.47,146.21}, 260. { 5.52,64.77},{19.39,81.84},{90.73,177.46},{46.94,101.25}, 261. {35.30,92.84},{25.31,80.86},{29.29,95.30},{79.68,160.63}, 262. {65.64,143.96},{81.97,173.79},{58.68,123.92},{37.35,94.26}, 263. {81.31,146.56},{10.82,34.36},{72.19,152.74},{96.81,157.39}, 264. {37.82,84.01},{26.02,85.45},{49.68,99.80},{63.61,134.18}, 265. {78.45,135.62},{28.06,99.20},{29.49,76.17},{38.73,90.19}, 266. {66.67,128.82},{ 4.14,43.63},{75.01,155.68},{12.38,39.83}, 267. {64.06,126.87},{77.44,154.01},{30.67,89.38},{30.38,85.81}, 268. {98.17,180.45},{72.62,131.67},{18.80,74.37},{56.99,112.44}, 269. {80.45,152.36},{43.87,96.62},{10.95,79.63},{ 8.13,48.84}, 270. {96.47,180.58},{57.99,139.71},{29.81,82.98},{ 7.69,59.04}, 271. {60.75,113.11},{61.26,127.28},{29.91,74.20},{72.81,130.29}, 272. {97.22,186.22},{16.10,67.14},{45.22,82.91},{59.90,136.15}, 273. {50.86,99.99},{40.09,89.91},{38.69,87.12},{38.22,86.34}, 274. {82.85,160.54},{44.59,114.27},{ 6.39,49.22},{53.02,118.97}, 275. {67.10,132.43},{87.17,167.48},{61.46,109.49},{79.66,163.17}, 276. {40.28,88.74},{81.76,164.45},{10.26,58.64},{14.58,81.13}, 277. {85.30,184.26},{64.06,132.71},{ 5.55,56.52},{96.97,187.38}, 278. {92.22,174.95},{42.45,100.51},{30.79,81.24},{ 4.25,61.71}, 279. {47.15,104.16},{35.87,86.39},{81.62,152.64},{42.46,95.25}, 280. {66.69,137.47},{33.21,84.65},{23.42,84.12},{99.30,187.76}, 281. {19.15,77.26},{17.74,70.35},{87.90,170.12},{47.01,118.00}, 282. {78.63,155.19},{92.38,163.60},{72.75,153.70},{79.92,138.69}, 283. {21.94,78.76},{55.51,120.91},{27.08,57.31},{12.83,45.59}, 284. {48.22,103.52},{35.64,87.26},{59.90,119.91},{50.05,110.55}, 285. { 0.23,41.68},{66.03,129.51},{42.67,95.15},{37.78,103.08}, 286. { 3.06,43.68},{53.80,102.89},{ 9.78,51.90},{94.94,185.83}, 287. {31.69,105.92},{70.50,123.84},{ 5.52,51.03},{ 0.93,47.63}, 288. {68.12,146.17},{ 6.86,51.21},{ 4.60,42.38},{72.98,138.03}, 289. {58.59,125.26},{40.21,88.92},{12.51,41.25},{31.12,65.03}, 290. {75.68,143.15},{74.02,141.52},{ 5.61,50.98},{82.39,162.02}, 291. {28.07,65.38},{71.22,145.28},{44.22,99.25},{72.03,123.62}, 292. {45.88,95.49},{76.37,136.85},{29.19,81.03},{63.46,142.45}, 293. {49.44,100.25},{81.71,132.07},{83.34,150.05},{38.88,93.91}, 294. {86.01,172.44},{51.32,110.43},{86.82,154.64},{70.02,140.53}, 295. {26.43,72.25},{34.48,91.08},{30.41,80.09},{24.77,80.43}, 296. { 8.14,53.39},{18.88,70.38},{26.90,73.96},{94.43,173.51}, 297. {24.45,62.18},{56.07,111.25},{66.96,136.85},{93.78,188.93}, 298. {75.18,144.91},{18.22,43.66},{97.70,170.91},{34.25,95.34}, 299. {12.16,53.60},{88.48,164.67},{81.58,176.86},{81.96,151.45}, 300. {50.13,109.26},{44.20,90.81},{52.84,121.91},{17.30,76.64}, 301. {53.60,120.26},{32.01,84.79},{72.56,149.71},{19.15,55.05}, 302. {78.26,164.32},{ 9.84,56.96},{ 2.48,50.11},{50.84,134.33}, 303. {90.65,164.89},{35.58,81.09},{72.54,151.27},{54.39,119.73}, 304. {44.15,105.72},{74.88,145.20},{86.66,158.64},{17.79,67.78}, 305. {54.84,115.87},{99.10,173.71},{93.02,174.29},{23.52,83.88}, 306. {19.56,68.81},{24.03,79.83},{11.73,35.38},{ 3.82,37.84}, 307. {61.92,130.48},{77.02,139.38},{91.29,161.65},{98.09,162.25}, 308. { 0.49,36.07},{75.44,138.86},{ 4.32,59.64},{79.99,143.62}, 309. {13.43,47.42},{44.44,110.46},{25.03,71.30},{71.95,147.86}, 310. {78.51,152.59},{ 3.43,34.40},{55.28,115.97},{88.77,165.45}, 311. {15.43,82.65},{99.09,179.00},{79.77,143.93},{52.73,116.02}, 312. {52.40,109.06},{37.24,83.30},{31.90,80.93},{68.13,127.42}, 313. {70.63,127.66},{55.84,132.35},{39.95,99.29},{ 6.84,40.38}, 314. {66.47,117.89},{20.53,83.00},{82.22,147.22},{23.74,73.03}, 315. {77.83,159.15},{11.29,64.73},{49.15,104.01},{52.54,105.95}, 316. {93.36,160.71},{51.35,106.68},{28.56,83.86},{78.27,147.91}, 317. { 0.25,50.56},{59.93,120.88},{ 8.58,49.04},{74.24,134.96}, 318. {51.22,98.10},{24.91,74.31},{87.07,160.74},{52.25,105.22}, 319. {91.43,152.78},{ 8.10,59.46},{94.97,178.64},{88.81,178.24}, 320. {88.45,150.89},{21.60,76.09},{70.62,122.85},{99.65,168.36}, 321. {73.32,142.58},{13.18,71.21},{37.26,88.09},{79.15,142.60}, 322. {20.05,91.09},{33.64,87.49},{21.84,64.04},{49.12,116.82}, 323. {52.57,125.37},{42.43,93.45},{22.54,94.46},{82.51,165.14}, 324. {77.64,132.17},{32.25,83.61},{10.77,55.92},{71.34,133.82}, 325. {60.85,127.38},{22.68,79.72},{30.62,77.71},{81.90,161.50}, 326. {10.22,47.19},{26.58,57.16},{43.66,113.00},{90.69,145.82}, 327. {12.64,58.91},{85.90,154.22},{18.03,53.36},{84.49,144.57}, 328. {87.51,169.54},{92.50,170.96},{51.99,123.08},{45.16,108.57}, 329. {71.40,137.44},{58.36,121.84},{76.06,143.46},{42.17,104.52}, 330. { 2.57,50.10},{11.44,41.77},{71.09,143.92},{88.92,151.09}, 331. {92.79,177.42},{90.72,157.64},{66.11,141.52},{ 2.33,38.80}, 332. {76.26,158.72},{76.52,150.07},{70.31,132.13},{52.77,119.85}, 333. {99.59,176.94},{ 8.16,56.11},{99.29,190.79},{25.00,77.22}, 334. {13.45,63.42},{17.35,70.05},{ 4.16,35.31},{86.57,152.50}, 335. {88.57,168.14},{67.96,123.67},{72.36,142.41},{10.95,73.06}, 336. {78.45,163.31},{71.69,139.46},{82.78,157.91},{80.14,161.51}, 337. {60.33,133.76},{ 9.44,46.00},{68.21,163.48},{30.78,88.27}, 338. {38.74,105.24},{19.52,62.94},{49.03,105.82},{76.01,138.95}, 339. {71.08,165.10},{49.97,108.11},{75.15,145.38},{ 5.20,62.33}, 340. {97.13,188.35},{87.18,176.82},{42.70,96.24},{62.98,126.92}, 341. {96.09,175.08},{90.77,190.99},{71.68,124.23},{15.67,61.12}, 342. {95.37,178.30},{40.64,83.70},{22.64,71.16},{30.22,105.64}, 343. {18.96,77.17},{56.47,98.36},{36.53,84.99},{13.11,73.50}, 344. {32.04,80.23},{72.49,135.67},{54.33,126.59},{13.54,51.56}, 345. {14.77,57.56},{24.09,90.04},{32.43,86.80},{ 3.82,43.03}, 346. {81.10,163.58},{45.39,96.22},{57.29,115.98},{76.10,151.06}, 347. { 7.74,56.38},{48.95,108.35},{40.07,101.13},{81.91,144.88}, 348. {64.47,124.53},{70.83,129.18},{ 7.05,44.06},{36.46,86.68}, 349. {32.53,89.07},{32.88,77.92},{ 6.62,29.48},{28.87,76.01}, 350. {37.36,90.05},{72.25,136.07},{81.47,173.47},{ 4.20,47.41}, 351. {98.64,166.84},{46.61,109.47},{45.38,88.61},{95.41,169.40}, 352. {66.63,122.42},{98.96,176.41},{77.60,166.70},{39.53,93.96}, 353. {73.29,138.23},{87.99,159.87},{34.35,91.01},{33.30,78.44}, 354. {29.29,78.05},{89.99,153.84},{ 3.90,31.79},{ 2.74,28.45}, 355. {74.07,144.41},{59.60,135.80},{83.19,154.17},{33.14,71.48}, 356. {71.18,127.25},{59.10,126.89},{14.88,60.57},{46.36,122.43}, 357. {97.68,166.02},{47.91,110.00},{94.43,185.03},{25.13,73.98}, 358. {30.66,83.04},{47.36,100.33},{20.03,57.00},{38.53,77.34}, 359. {53.29,122.85},{77.72,146.52},{23.42,84.64},{96.85,170.99}, 360. { 8.49,68.27},{71.67,127.11},{84.22,158.75},{35.25,87.63}, 361. {74.00,140.25},{32.42,91.48},{88.91,156.95},{88.11,163.18}, 362. {60.49,132.55},{63.94,149.75},{95.21,172.49},{ 2.14,37.70}, 363. { 2.33,36.29},{24.38,83.62},{87.11,162.19},{37.16,85.65}, 364. {81.57,152.86},{49.26,116.82},{54.72,108.56},{65.82,132.02}, 365. {10.93,47.63},{71.92,111.28},{12.67,36.23},{67.35,101.56}, 366. {86.25,169.18},{97.89,194.43},{40.63,106.58},{73.87,135.71} 367. }; |

**Mean Time**



# 2 CUDA

## 2.1 Password Cracking

|  |
| --- |
| 1. #include <stdio.h> 2. #include <cuda\_runtime\_api.h> 3. #include <time.h> 4. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  7. To Compile: 8. nvcc -o passwordcrack passwordcrack\_cuda.cu 9. To Run 10. ./passwordcrack 12. Dr Kevan Buckley, University of Wolverhampton, 2018 13. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/   17. \_\_device\_\_ **int** is\_a\_match(**char** \*attempt) { 18. **char** my\_password1[] = "ET8789"; 19. **char** my\_password2[] = "IR3475"; 20. **char** my\_password3[] = "SB4665"; 21. **char** my\_password4[] = "BV7870";  24. **char** \*a = attempt; 25. **char** \*d = attempt; 26. **char** \*h = attempt; 27. **char** \*i = attempt; 28. **char** \*p1 = my\_password1; 29. **char** \*p2 = my\_password2; 30. **char** \*p3 = my\_password3; 31. **char** \*p4 = my\_password4; 33. **while**(\*a == \*p1) { 34. **if**(\*a == '\0') 35. { 36. printf("Password: %s\n",my\_password1); 37. **break**; 38. } 40. a++; 41. p1++; 42. } 44. **while**(\*d == \*p2) { 45. **if**(\*d == '\0') 46. { 47. printf("Password: %s\n",my\_password2); 48. **break**; 49. } 51. d++; 52. p2++; 53. } 55. **while**(\*h == \*p3) { 56. **if**(\*h == '\0') 57. { 58. printf("Password: %s\n",my\_password3); 59. **break**; 60. } 62. h++; 63. p3++; 64. } 66. **while**(\*i == \*p4) { 67. **if**(\*i == '\0') 68. { 69. printf("Password: %s\n",my\_password4); 70. **return** 1; 71. } 73. i++; 74. p4++; 75. } 76. **return** 0; 78. } 80. \_\_global\_\_ **void**  kernel() { 81. **char** i1,i2,i3,i4; 83. **char** password[7]; 84. password[6] = '\0'; 86. **int** i = blockIdx.x+65; 87. **int** j = threadIdx.x+65; 88. **char** firstMatch = i; 89. **char** secondMatch = j; 91. password[0] = firstMatch; 92. password[1] = secondMatch; 93. **for**(i1='0'; i1<='9'; i1++){ 94. **for**(i2='0'; i2<='9'; i2++){ 95. **for**(i3='0'; i3<='9'; i3++){ 96. **for**(i4='0'; i4<='9'; i4++){ 97. password[2] = i1; 98. password[3] = i2; 99. password[4] = i3; 100. password[5] = i4; 101. **if**(is\_a\_match(password)) { 102. } 103. **else** { 105. } 106. } 107. } 108. } 109. } 110. } 112. **int** time\_difference(**struct** timespec \*start, 113. **struct** timespec \*finish, 114. **long** **long** **int** \*difference) { 115. **long** **long** **int** ds =  finish->tv\_sec - start->tv\_sec; 116. **long** **long** **int** dn =  finish->tv\_nsec - start->tv\_nsec; 117. **if**(dn < 0 ) { 118. ds--; 119. dn += 1000000000; 120. } 121. \*difference = ds \* 1000000000 + dn; 122. **return** !(\*difference > 0); 123. }  126. **int** main() { 128. **struct**  timespec start, finish; 129. **long** **long** **int** time\_elapsed; 130. clock\_gettime(CLOCK\_MONOTONIC, &start); 132. kernel <<<26,26>>>(); 133. cudaThreadSynchronize(); 135. clock\_gettime(CLOCK\_MONOTONIC, &finish); 136. time\_difference(&start, &finish, &time\_elapsed); 137. printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed, (time\_elapsed/1.0e9)); 139. **return** 0; 140. } |

CUDA mean running time comparing with original program, Multi thread program and CUDA version of password cracking.

|  |  |  |  |
| --- | --- | --- | --- |
| **Run time** | **Original** | **Multithread** | **CUDA** |
| 1 | 515327416109 | 253364837088 | 102675970 |
| 2 | 513725913738 | 254653411494 | 57992930 |
| 3 | 516503597715 | 257242191080 | 59011875 |
| 4 | 510567181476 | 266215610321 | 58552307 |
| 5 | 511684660337 | 260193453845 | 58289075 |
| 6 | 510520288292 | 269390773805 | 58617149 |
| 7 | 504789551079 | 258632131485 | 58269481 |
| 8 | 505053580512 | 258429777495 | 58215004 |
| 9 | 505854465487 | 258139711870 | 57994957 |
| 10 | 506524673766 | 260346290280 | 58087092 |
| **Mean** | 510055132851.1 Nano second | 259660818876.3  Nano second | 62770584  Nano second |

Result Analysis

* Original= 510055132851.1 Nano second which is 8.500918880851836 Minute.
* Multi thread=259660818876.3 Nano second which is 4.327680314605086 Minute.
* CUDA version= 62770584 Nano second which is 0.001046176400000021 Minute.

Let's evaluate the CUDA version with the original program and multi-thread program, as CUDA operates on parallel processing, GPU hosting several thousand cores where all data is processed by these cores. The ability to speed up depends on the parallelism of the program to use high cores to process so that the CUDA output is better than a multi-threading and original program. (Pradhan & Shah , 2019)

## 2.2 Image Processing

|  |
| --- |
| 1. #include <stdio.h> 2. #include <stdlib.h> 3. #include <time.h> 4. #include <GL/glut.h> 5. #include <GL/gl.h> 6. #include <malloc.h> 7. #include <signal.h> 8. #include <cuda\_runtime\_api.h> 9. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 10. //to compile nvcc -o Image\_cuda Image\_cuda.cu -lglut -lGL 11. // ./Image\_cuda > results.txt 12. Dr Kevan Buckley, University of Wolverhampton, 2018 13. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ 14. #define width 100 15. #define height 72 17. unsigned **char** results[width \* height]; 18. unsigned **char** image[] = { 0,0,0,0,0,0,0,0,0,0,0,0,0,255,0,0,0,255, 19. 255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0, 20. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 21. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 22. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 23. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 24. 255,0,255,255,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255, 25. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 26. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 27. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 28. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 29. 0,0,0,0,0,255,0,255,255,255,255,255,255,255,255,255,255,0,0, 30. 0,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0, 31. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 32. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 33. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 34. 0,0,0,0,0,0,0,0,0,255,255,0,255,255,255,255,255,255,255, 35. 255,255,255,255,255,255,255,255,0,0,0,0,0,0,255,255,255,255,255, 36. 255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 37. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 38. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 39. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255, 40. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 41. 0,0,0,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,0, 42. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 43. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 44. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 45. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 46. 255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,255,255,255,255, 47. 255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 48. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 49. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 50. 0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 51. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 52. 255,255,255,255,0,0,0,0,0,255,255,255,255,255,255,255,255,0,0, 53. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 54. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 55. 0,0,0,0,0,0,0,0,0,255,255,0,255,255,255,255,255,255,255, 56. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 57. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0, 58. 255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0, 59. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 60. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,255,255, 61. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 62. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 63. 255,255,255,255,255,0,0,0,0,0,255,255,255,255,255,255,255,255,255, 64. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 65. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 66. 255,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 67. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 68. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0, 69. 0,0,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0, 70. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 71. 0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 72. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 73. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 74. 255,255,255,255,255,255,255,255,255,0,0,0,0,0,255,255,255,255,255, 75. 255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 76. 0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255, 77. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,255, 78. 0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 79. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 80. 255,0,0,0,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0, 81. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,255,255, 82. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 83. 255,255,255,0,0,0,255,255,255,255,255,0,255,255,0,0,255,255,255, 84. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 85. 255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,255,255,255, 86. 255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 87. 255,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 88. 255,255,255,255,255,255,255,255,0,255,0,255,255,255,255,255,0,255,255, 89. 0,255,255,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 90. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 91. 255,255,0,0,0,0,0,255,255,255,0,0,0,0,0,0,0,0,0, 92. 0,0,0,0,0,255,255,0,255,255,255,255,255,255,255,255,255,255,255, 93. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,255,255,255,255, 94. 255,255,0,255,255,0,255,255,255,0,255,0,255,0,255,255,255,255,255, 95. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 96. 255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,0,0,0,0, 97. 0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255, 98. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 99. 0,255,255,255,255,255,255,0,255,255,0,255,255,255,0,0,0,255,255, 100. 0,0,0,255,0,255,255,0,255,255,255,255,255,255,255,255,255,255,255, 101. 255,255,255,255,255,255,255,255,255,255,0,255,255,255,255,255,255,0,255, 102. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,0,255,255, 103. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 104. 255,255,255,255,255,255,255,255,255,255,255,0,0,0,255,0,255,255,255, 105. 0,0,255,255,255,0,255,0,255,0,255,0,0,255,255,0,255,255,255, 106. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,0, 107. 0,255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 108. 0,255,0,255,255,255,255,255,255,255,255,255,255,255,0,255,255,255,255, 109. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 110. 255,0,0,0,255,0,255,255,255,255,0,0,255,255,0,0,255,0,255, 111. 255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 112. 255,255,0,0,0,0,255,255,0,255,0,0,0,0,0,0,0,0,0, 113. 0,0,0,0,0,0,255,0,255,255,255,255,255,255,255,255,255,255,0, 114. 0,255,0,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 115. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,255, 116. 0,0,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 117. 255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,0,0,0,0, 118. 0,0,0,0,0,0,0,0,0,0,255,255,0,255,255,255,255,255,255, 119. 255,255,255,255,0,255,255,0,0,0,255,0,0,0,0,0,0,0,0, 120. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 121. 255,255,255,255,255,0,255,255,0,255,0,0,255,255,255,255,255,255,255, 122. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,255,0,255, 123. 255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255, 124. 255,255,255,255,255,255,255,255,255,0,255,255,0,0,0,255,0,255,0, 125. 255,255,255,0,255,0,255,0,0,0,255,255,255,255,255,255,255,255,255, 126. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,255, 127. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 128. 0,255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 129. 0,255,0,255,255,255,255,255,255,255,255,255,255,255,0,255,255,0,0, 130. 0,0,255,255,0,255,255,255,0,255,255,255,0,255,0,255,0,0,0, 131. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 132. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 133. 255,255,255,255,255,255,255,255,0,255,0,0,0,0,0,0,0,0,0, 134. 0,0,0,0,0,0,255,0,255,255,255,255,255,255,255,255,255,255,255, 135. 255,255,255,255,0,0,255,255,255,0,255,255,255,0,0,0,255,255,0, 136. 0,0,255,255,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255, 137. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 138. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,0,0,0,0, 139. 0,0,0,0,0,0,0,0,0,0,0,255,0,255,255,255,255,255,255, 140. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,255,0, 141. 255,255,255,0,0,255,255,255,0,0,0,255,255,255,255,0,255,255,0, 142. 0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 143. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 144. 255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,255, 145. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 146. 255,255,255,255,0,0,0,255,0,255,255,255,0,0,255,255,255,255,255, 147. 255,0,255,255,0,255,255,0,255,0,255,0,255,255,255,255,255,255,255, 148. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 149. 255,255,255,0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0, 150. 0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 151. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,255,0,255, 152. 255,255,255,255,255,255,0,0,255,255,255,255,255,0,0,255,255,255,255, 153. 255,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 154. 255,255,255,255,255,255,255,255,0,255,255,0,0,0,0,0,0,0,0, 155. 0,0,0,0,0,0,255,255,255,255,255,0,255,255,0,255,0,0,255, 156. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 157. 255,255,255,255,255,255,255,255,255,255,0,255,0,0,0,255,255,0,0, 158. 0,255,255,255,255,255,255,0,0,0,0,0,255,255,255,255,255,0,255, 159. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,0,0,0, 160. 0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,0,0,0, 161. 0,255,0,255,0,255,255,0,0,255,255,255,255,255,255,255,255,255,255, 162. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0, 163. 0,0,0,0,255,0,255,255,255,255,255,255,0,255,0,255,255,255,255, 164. 0,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0, 165. 255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255, 166. 255,255,255,255,0,255,255,0,255,255,0,0,255,255,0,255,0,0,0, 167. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 168. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0, 169. 0,255,255,0,0,0,255,0,255,255,255,255,255,255,255,255,255,255,255, 170. 255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0, 171. 0,255,255,0,255,255,255,255,0,0,255,255,0,0,0,255,0,255,255, 172. 255,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 173. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 174. 255,255,255,255,0,0,255,0,255,255,0,255,0,0,0,255,255,255,255, 175. 255,255,255,255,255,255,255,255,0,255,255,0,0,0,0,0,0,0,0, 176. 0,0,0,0,0,0,255,255,0,255,255,255,255,0,255,255,255,0,255, 177. 0,255,0,255,255,255,0,0,255,255,255,255,255,255,255,255,255,255,255, 178. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 179. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,255,0,255, 180. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,0,0,0, 181. 0,0,0,0,0,0,0,0,0,0,0,255,255,0,255,255,255,255,255, 182. 255,255,255,0,0,0,255,0,255,255,0,255,0,255,255,0,255,255,255, 183. 255,255,255,255,0,255,255,0,0,255,0,0,255,255,255,255,255,255,255, 184. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 185. 255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0, 186. 255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255, 187. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,255,255,0,255, 188. 255,0,255,0,255,255,255,255,255,0,0,0,255,0,0,0,255,0,255, 189. 255,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 190. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 191. 255,255,255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0, 192. 0,0,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 193. 255,255,255,255,255,255,255,0,0,0,255,255,255,255,255,255,0,255,255, 194. 0,0,255,0,0,0,255,255,0,255,0,255,0,255,0,255,255,255,255, 195. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 196. 255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0, 197. 0,0,0,0,0,0,0,255,0,255,255,255,255,255,255,255,255,255,255, 198. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 199. 0,0,0,255,0,0,255,255,0,255,0,255,0,0,0,0,255,0,255, 200. 0,255,0,0,255,0,0,0,255,255,255,255,255,255,255,255,255,255,255, 201. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0, 202. 0,0,0,0,0,0,0,0,0,0,0,255,255,0,255,255,255,255,255, 203. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 204. 255,255,255,255,255,255,255,255,255,0,0,255,255,0,255,255,0,0,0, 205. 0,255,255,0,255,0,0,0,0,0,255,255,255,0,255,255,255,255,255, 206. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 207. 255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0, 208. 255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,255,255,255,255, 209. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 210. 0,0,255,0,255,255,0,255,0,255,0,0,255,0,0,255,255,0,0, 211. 255,255,255,0,255,0,255,0,0,0,255,255,255,255,255,255,255,255,255, 212. 255,255,255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0, 213. 0,0,255,255,255,255,255,255,255,255,255,255,255,0,255,255,0,255,0, 214. 0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 215. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,0,255,255,0, 216. 255,255,255,255,255,255,255,255,0,255,0,255,0,255,255,255,0,255,255, 217. 0,255,255,255,255,255,255,255,255,0,255,0,0,0,0,0,0,0,0, 218. 0,0,0,0,0,0,0,255,0,255,255,255,255,255,255,255,255,255,0, 219. 255,255,0,0,255,255,255,0,0,255,255,0,255,255,255,255,255,255,255, 220. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 221. 255,255,255,255,0,255,0,255,0,255,255,255,255,0,255,0,255,0,0, 222. 255,255,0,255,0,0,255,0,255,255,255,255,255,255,0,255,0,0,0, 223. 0,0,0,0,0,0,0,0,0,0,0,0,255,0,255,255,255,255,255, 224. 255,255,255,255,0,255,255,0,0,255,255,255,0,0,255,0,0,255,255, 225. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 226. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 227. 255,0,255,0,255,255,255,0,0,0,255,255,0,255,255,255,255,255,255, 228. 0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0, 229. 255,255,255,255,255,255,255,255,255,0,255,255,0,255,255,255,255,0,0, 230. 255,0,255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255, 231. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 232. 255,255,255,255,255,255,0,255,0,255,255,255,0,255,255,0,255,0,0, 233. 0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0, 234. 0,0,255,255,0,255,255,255,255,255,255,255,255,255,255,255,0,255,255, 235. 0,255,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,0,255, 236. 255,255,255,0,0,255,255,0,255,255,255,255,255,255,255,255,255,255,255, 237. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,0,0, 238. 255,0,255,0,255,255,255,255,255,0,255,255,0,0,0,0,0,0,0, 239. 0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255, 240. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,255,255, 241. 255,255,255,0,0,255,255,255,0,0,255,0,0,255,255,0,255,255,0, 242. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 243. 255,255,255,255,255,255,255,255,0,0,0,255,255,255,0,255,0,0,0, 244. 0,0,0,0,0,0,0,0,0,0,0,0,255,0,255,255,255,255,255, 245. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 246. 255,0,0,0,255,255,255,255,0,255,255,255,0,255,255,255,0,0,255, 247. 255,0,255,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 248. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 249. 0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,0, 250. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 251. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,255,0,255, 252. 255,255,0,0,0,255,0,255,0,0,255,255,255,255,255,255,255,0,255, 253. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 254. 255,255,255,255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0, 255. 0,0,0,255,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255, 256. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 257. 255,255,255,255,255,255,255,0,255,255,255,0,0,0,0,255,255,255,255, 258. 0,255,255,0,255,255,0,255,0,255,255,255,255,255,255,255,255,255,255, 259. 255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0, 260. 0,0,0,0,0,0,0,255,255,0,0,0,0,0,0,0,0,0,0, 261. 0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 262. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,255,0,0,255, 263. 0,255,255,255,255,0,255,255,0,0,0,0,255,0,255,255,255,0,255, 264. 0,255,255,0,255,255,255,255,255,255,255,255,255,255,0,255,255,0,0, 265. 0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,0,0,0, 266. 0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255, 267. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 268. 255,255,255,255,255,255,255,255,255,255,0,255,255,0,255,255,0,255,0, 269. 255,0,0,0,255,0,255,0,0,255,255,255,255,255,255,255,255,255,255, 270. 0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255, 271. 0,0,0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 272. 0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 273. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0, 274. 255,255,0,255,0,255,0,255,0,255,0,0,0,255,255,255,255,255,255, 275. 255,255,255,255,255,0,255,255,0,0,0,0,0,0,0,0,0,0,0, 276. 0,0,0,255,0,0,0,0,255,0,0,0,0,255,255,0,0,0,0, 277. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255, 278. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 279. 255,255,255,255,255,255,255,255,255,0,255,0,255,0,255,0,0,0,255, 280. 255,255,255,255,255,255,255,255,255,255,0,255,0,0,0,0,0,0,0, 281. 0,0,0,0,0,0,0,0,255,0,0,0,0,255,0,0,0,0,255, 282. 255,0,255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0, 283. 0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255, 284. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 285. 255,0,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0, 286. 0,0,0,0,0,0,0,0,0,0,0,0,0,255,0,0,0,0,255, 287. 255,255,0,0,255,0,255,255,0,0,255,0,0,0,255,255,0,255,255, 288. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255, 289. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 290. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 291. 0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255, 292. 0,0,0,255,255,255,255,255,255,255,255,255,255,0,255,255,0,0,255, 293. 255,255,0,255,255,0,0,255,255,0,0,0,0,0,0,0,0,0,0, 294. 0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255, 295. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 296. 255,255,255,255,255,0,255,255,0,0,0,0,0,0,0,0,0,0,0, 297. 0,0,0,255,255,0,0,0,255,255,0,255,255,255,255,255,255,255,0, 298. 255,255,0,0,255,255,255,0,255,255,255,255,255,0,0,255,0,0,0, 299. 255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 300. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 301. 255,255,255,255,255,255,255,255,255,255,0,255,255,0,0,0,0,0,0, 302. 0,0,0,0,0,0,0,0,255,255,0,0,0,255,255,255,255,255,255, 303. 255,0,255,255,0,255,255,0,255,255,0,255,0,255,255,255,255,255,0, 304. 0,255,0,0,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0, 305. 0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255, 306. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,0,0, 307. 0,0,0,0,0,0,0,0,0,0,0,0,0,255,0,0,0,0,0, 308. 0,255,255,0,255,255,255,255,255,0,255,255,0,255,0,0,255,0,255, 309. 255,255,255,0,0,255,255,0,0,255,255,0,0,0,0,0,0,0,255, 310. 0,0,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 311. 0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 312. 255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255, 313. 0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,255,255,255,255, 314. 0,0,255,0,255,255,255,255,0,0,255,255,0,0,255,255,0,0,0, 315. 0,0,0,255,255,0,0,0,255,0,0,0,255,0,255,0,0,0,0, 316. 0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255, 317. 255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0, 318. 0,0,0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 319. 0,0,0,255,0,0,0,255,0,0,255,255,0,0,0,255,255,0,0, 320. 255,255,255,255,0,0,0,0,255,255,0,0,0,255,255,255,255,255,0, 321. 255,255,0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 322. 0,0,0,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,0, 323. 0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,0,0,0, 324. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,255, 325. 255,255,255,255,255,255,255,0,0,0,0,0,0,255,255,255,255,0,255, 326. 255,255,255,0,255,255,0,255,255,255,0,255,0,0,255,255,0,0,0, 327. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0, 328. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255, 329. 255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0, 330. 0,0,0,0,0,255,255,255,255,255,255,255,255,255,0,0,0,0,255, 331. 0,255,255,255,255,255,255,255,0,255,255,255,255,255,0,255,255,0,0, 332. 255,255,0,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0, 333. 0,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 334. 0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,0,0, 335. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255, 336. 0,0,0,0,255,255,255,255,255,255,255,255,255,0,255,255,255,255,255, 337. 0,255,255,255,255,255,0,0,255,255,255,255,0,0,255,255,0,0,255, 338. 255,0,0,0,0,0,0,255,0,0,0,0,0,0,0,0,0,0,0, 339. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255, 340. 255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0, 341. 0,0,0,0,0,0,0,0,0,255,255,255,255,0,255,255,255,0,0, 342. 255,255,0,255,255,0,255,255,255,255,255,0,255,255,255,255,255,0,255, 343. 255,255,0,255,255,255,255,0,0,0,0,0,255,0,0,0,0,0,0, 344. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 345. 0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255, 346. 255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 347. 255,255,255,0,0,255,255,255,255,255,0,255,255,255,255,0,0,255,255, 348. 0,255,255,255,255,255,0,255,255,0,0,255,255,0,0,0,255,255,0, 349. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 350. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 351. 0,0,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0, 352. 0,0,0,0,0,0,0,0,0,0,255,255,255,255,0,0,255,255,255, 353. 255,0,0,255,255,0,255,255,255,255,255,0,255,255,0,255,255,255,0, 354. 0,0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 355. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 356. 0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255, 357. 255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 358. 0,0,255,0,0,255,255,0,255,255,0,255,255,255,255,255,0,255,255, 359. 0,255,255,255,0,0,0,255,255,0,0,0,0,0,0,0,0,0,0, 360. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 361. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 362. 0,0,0,0,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0, 363. 0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,255,255,0, 364. 255,255,0,255,255,0,0,0,0,0,0,0,255,255,0,0,0,0,0, 365. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 366. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 367. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255, 368. 255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 369. 0,0,0,0,0,255,255,0,255,255,255,255,255,0,0,0,0,255,0, 370. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 371. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 372. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 373. 0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,0,0,0, 374. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,0,0, 375. 0,0,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 376. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 377. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 378. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255, 379. 255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0, 380. 0,0,0,0,0,0,0,255,255,0,0,0,0,0,0,0,0,0,0, 381. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 382. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 383. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 384. 0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,0,0, 385. 0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,0,0,0, 386. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 387. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 388. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 389. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255, 390. 255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,255,255, 391. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 392. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 393. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 394. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 395. 0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255, 396. 255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 397. }; 399. \_\_global\_\_ **void** detect\_edges(unsigned **char** \*in, unsigned **char** \*out) { 400. **int** i = (blockIdx.x \* 72) + threadIdx.x; 401. **int** x, y; // the pixel of interest 402. **int** a, d, h, k; // the pixels adjacent to x,y used for the calculation 403. **int** r; // the result of calculate 405. y = i / width; 406. x = i - (width \* y); 408. **if** (x == 0 || y == 0 || x == width - 1 || y == height - 1) { 409. out[i] = 0; 410. } **else** { 411. a = i + width; 412. d = i - 1; 413. h = i + 1; 414. k = i - width; 416. r = (in[i] \* 4) + (in[a] \* -1) + (in[d] \* -1) + (in[h] \* -1) 417. + (in[k] \* -1); 419. **if** (r > 0) { // if the result is positive this is an edge pixel 420. out[i] = 255; 421. } **else** { 422. out[i] = 0; 423. } 424. } 425. }  428. **void** tidy\_and\_exit() { 429. exit(0); 430. } 432. **void** sigint\_callback(**int** signal\_number){ 433. printf("\nInterrupt from keyboard\n"); 434. tidy\_and\_exit(); 435. } 437. **static** **void** display() { 438. glClear(GL\_COLOR\_BUFFER\_BIT); 439. glRasterPos4i(-1, -1, 0, 1); 440. glDrawPixels(width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, image); 441. glRasterPos4i(0, -1, 0, 1); 442. glDrawPixels(width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, results); 443. glFlush(); 444. } 446. **static** **void** key\_pressed(unsigned **char** key, **int** x, **int** y) { 447. **switch**(key){ 448. **case** 27: 449. tidy\_and\_exit(); 450. **break**; 451. **default**: 452. printf("\nPress escape to exit\n"); 453. **break**; 454. } 455. } 457. **int** time\_difference(**struct** timespec \*start, **struct** timespec \*finish, 458. **long** **long** **int** \*difference) { 459. **long** **long** **int** ds =  finish->tv\_sec - start->tv\_sec; 460. **long** **long** **int** dn =  finish->tv\_nsec - start->tv\_nsec; 462. **if**(dn < 0 ) { 463. ds--; 464. dn += 1000000000; 465. } 466. \*difference = ds \* 1000000000 + dn; 467. **return** !(\*difference > 0); 468. } 470. **int** main(**int** argc, **char** \*\*argv) { 472. unsigned **char** \*d\_results; 473. unsigned **char** \*d\_image; 475. cudaMalloc((**void**\*\*)&d\_image, **sizeof**(unsigned **char**) \* (width \* height)); 477. cudaMalloc((**void**\*\*)&d\_results, **sizeof**(unsigned **char**) \* (width \* height)); 479. cudaMemcpy(d\_image, &image, **sizeof**(unsigned **char**) \* (width \* height), cudaMemcpyHostToDevice); 480. signal(SIGINT, sigint\_callback);  483. **struct** timespec start, finish; 484. **long** **long** **int** time\_elapsed; 486. clock\_gettime(CLOCK\_MONOTONIC, &start); 487. detect\_edges<<<100,72>>>(d\_image, d\_results); 488. cudaThreadSynchronize(); 490. cudaMemcpy(&results, d\_results, **sizeof**(unsigned **char**) \* (width \* height), cudaMemcpyDeviceToHost);   494. clock\_gettime(CLOCK\_MONOTONIC, &finish); 495. time\_difference(&start, &finish, &time\_elapsed); 496. printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed, 497. (time\_elapsed/1.0e9)); 498. cudaFree(&d\_image); 499. cudaFree(&d\_results); 501. glutInit(&argc, argv); 502. glutInitWindowSize(width \* 2,height); 503. glutInitDisplayMode(GLUT\_SINGLE | GLUT\_LUMINANCE); 505. glutCreateWindow("6CS005 Image Progessing Courework"); 506. glutDisplayFunc(display); 507. glutKeyboardFunc(key\_pressed); 508. glClearColor(0.0, 1.0, 0.0, 1.0); 510. glutMainLoop(); 512. tidy\_and\_exit(); 514. **return** 0; 515. } |

**Compare the mean running time of the CUDA version with the original program and multithreaded versions. Comment of the effect of invoking more threads than cores.**

|  |  |  |  |
| --- | --- | --- | --- |
| Run time | Original | Multithread | CUDA |
| 1 | 73251 | 342517 | 228796 |
| 2 | 61698 | 469085 | 381101 |
| 3 | 62160 | 203127 | 1370830 |
| 4 | 61606 | 241242 | 185344 |
| 5 | 66720 | 346862 | 405813 |
| 6 | 299407 | 193769 | 186620 |
| 7 | 61957 | 254527 | 227852 |
| 8 | 62064 | 1598401 | 178031 |
| 9 | 62564 | 196090 | 184438 |
| 10 | 91841 | 247459 | 209565 |
| Mean NS | 90326.8 | 409307.9 | 355839 |

**Result analysis**

Original program takes 90326.8 Nano second and program by using multi thread takes 409307.9 Nano second and CUDA takes 355839 second which means cuda is taking less time compare to multi thread. CUDA is a parallel processing platform which was is developed by NVDIA.CUDA Increases computing performance by connecting GPU power.

## 2.3 Linear Regression

|  |
| --- |
| 1. #include <stdio.h> 2. #include <math.h> 3. #include <time.h> 4. #include <unistd.h> 5. #include <cuda\_runtime\_api.h> 6. #include <errno.h> 7. #include <unistd.h> 8. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 9. To compile: nvcc -o Cuda\_Lr Cuda\_Lr.cu -lm 10. To run: ./Cuda\_Lr 11. Name:1928720\_Adhikar\_chaudhary 12. @University of wolverhamption 13. Dr Kevan Buckley, University of Wolverhampton, 2018 14. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ 15. **typedef** **struct** point\_t { 16. **double** x; 17. **double** y; 18. } point\_t; 20. **int** n\_data = 1000; 21. \_\_device\_\_ **int** d\_n\_data = 1000;  24. point\_t data[] = { 25. {87.72,160.36},{79.77,154.56},{74.88,144.94},{67.05,157.02}, 26. {73.75,121.36},{69.58,132.51},{67.21,143.57},{72.13,141.31}, 27. {87.64,144.77},{65.42,146.46},{83.09,144.82},{73.30,137.10}, 28. {76.75,151.67},{69.17,144.50},{87.61,171.62},{83.56,164.97}, 29. {75.57,137.55},{73.80,156.25},{75.75,130.90},{97.13,193.14}, 30. { 6.54,74.00},{54.31,100.05},{57.36,127.37},{93.54,172.18}, 31. {98.26,190.44},{87.27,149.69},{66.44,143.27},{98.24,171.56}, 32. {27.74,98.27},{10.51,57.95},{88.17,179.14},{22.66,63.64}, 33. {30.46,64.81},{65.24,135.15},{ 1.71,39.30},{91.29,171.76}, 34. {77.43,153.66},{68.29,132.07},{33.43,82.08},{60.40,119.17}, 35. {15.86,64.98},{61.96,126.18},{11.05,58.97},{23.99,67.74}, 36. {21.13,65.78},{34.79,97.87},{22.32,74.86},{78.29,136.48}, 37. {25.38,84.45},{94.49,169.91},{35.62,99.82},{58.20,127.53}, 38. {98.37,182.22},{93.79,168.74},{26.98,91.60},{30.55,103.77}, 39. { 5.86,44.79},{78.96,159.17},{93.19,158.82},{24.73,51.92}, 40. {94.91,186.12},{61.84,135.07},{ 2.60,37.48},{95.93,189.32}, 41. {95.39,157.13},{20.24,69.22},{29.93,71.36},{68.29,148.83}, 42. {36.68,92.16},{93.29,180.89},{30.53,75.37},{48.10,120.04}, 43. {83.17,153.60},{ 8.22,50.63},{ 8.76,50.80},{97.71,171.61}, 44. {13.28,46.71},{ 1.07,43.74},{11.16,71.21},{20.98,78.43}, 45. {21.89,50.58},{11.51,55.46},{ 3.22,43.94},{41.33,86.40}, 46. { 9.72,66.88},{10.91,51.56},{40.75,100.96},{81.92,161.87}, 47. {54.44,112.70},{11.35,62.04},{69.49,123.86},{50.51,96.47}, 48. {74.84,150.85},{77.35,150.52},{81.72,163.59},{11.65,37.53}, 49. {13.48,52.63},{79.02,166.24},{84.93,171.49},{59.13,117.54}, 50. {70.14,129.94},{58.90,118.01},{28.64,107.03},{43.71,112.86}, 51. {58.86,114.54},{78.67,160.78},{94.74,171.99},{70.31,122.40}, 52. {16.20,74.80},{58.59,110.98},{70.81,130.09},{ 7.86,46.69}, 53. {91.46,153.64},{34.05,87.81},{24.06,68.39},{66.23,134.24}, 54. {58.69,121.12},{ 0.18,38.19},{20.48,63.34},{63.79,125.30}, 55. {27.38,80.17},{39.12,100.73},{85.45,151.11},{48.89,92.37}, 56. {81.54,155.19},{20.52,55.22},{16.33,56.39},{82.69,158.63}, 57. {78.03,160.98},{ 3.36,62.70},{ 9.16,36.34},{ 5.56,50.25}, 58. {96.94,185.14},{46.54,107.56},{13.32,54.76},{21.63,58.16}, 59. {35.36,86.17},{92.63,175.36},{35.77,93.56},{24.99,73.63}, 60. {78.27,153.33},{44.20,89.01},{ 8.41,28.53},{22.39,65.89}, 61. { 4.01,63.33},{61.57,131.79},{39.85,95.92},{76.59,132.38}, 62. {49.60,117.48},{54.00,114.01},{92.20,160.82},{98.73,180.85}, 63. {56.18,108.40},{75.10,144.54},{36.91,105.85},{36.52,75.99}, 64. {47.65,79.93},{72.52,151.63},{50.78,112.17},{85.76,151.27}, 65. {14.41,51.07},{48.66,109.44},{50.39,104.20},{95.63,174.36}, 66. {69.74,140.10},{25.53,87.75},{67.15,141.26},{55.44,129.77}, 67. {20.44,65.95},{98.96,181.37},{47.04,111.25},{58.45,117.28}, 68. {57.79,123.58},{83.84,171.31},{61.97,131.09},{37.66,109.35}, 69. {65.32,124.56},{32.31,92.17},{64.08,140.31},{53.53,119.28}, 70. {46.02,102.80},{ 8.19,46.99},{75.24,134.02},{11.54,67.14}, 71. {37.35,94.89},{17.90,73.66},{68.74,127.84},{35.61,74.80}, 72. {12.89,61.03},{51.57,127.06},{69.44,137.16},{55.00,87.07}, 73. { 9.46,66.57},{ 7.32,50.65},{18.69,60.81},{91.80,157.00}, 74. {23.13,80.31},{79.77,160.31},{25.92,67.57},{88.84,157.95}, 75. {59.62,122.91},{20.02,74.74},{66.18,130.20},{22.33,67.57}, 76. {34.98,72.13},{27.88,80.57},{74.11,152.93},{63.45,123.91}, 77. {34.17,76.27},{59.99,137.07},{26.78,70.38},{30.99,79.08}, 78. { 3.26,42.81},{51.85,97.62},{38.00,89.05},{69.57,128.69}, 79. {55.77,102.44},{38.64,95.04},{31.94,80.16},{33.57,84.21}, 80. {49.34,115.16},{60.37,126.82},{48.79,79.20},{53.62,101.98}, 81. {44.49,120.58},{40.72,92.32},{62.30,134.92},{86.71,167.45}, 82. {28.92,90.49},{ 3.64,50.38},{ 2.98,51.03},{ 7.72,50.72}, 83. {36.01,88.98},{12.80,53.33},{18.18,59.93},{ 8.74,39.72}, 84. {93.05,157.51},{78.51,157.36},{ 9.32,51.91},{28.43,83.25}, 85. { 5.63,57.81},{ 5.75,49.78},{33.39,89.81},{98.06,183.34}, 86. {61.33,124.19},{90.28,167.83},{97.93,180.90},{24.74,74.19}, 87. {28.42,90.02},{67.13,126.96},{29.53,65.54},{28.34,82.58}, 88. { 5.75,35.03},{36.57,89.60},{18.70,68.23},{85.61,147.33}, 89. {90.19,148.59},{29.44,82.07},{95.34,164.13},{25.79,76.89}, 90. {18.18,55.48},{47.36,99.63},{ 9.34,53.88},{95.16,174.51}, 91. {79.94,148.24},{ 1.82,30.86},{75.83,138.77},{15.10,60.29}, 92. {34.45,91.55},{ 0.41,39.02},{12.90,64.51},{75.66,161.10}, 93. {40.13,85.95},{57.37,109.43},{79.64,155.61},{25.54,76.82}, 94. {33.13,80.14},{31.55,75.12},{ 0.93,51.83},{36.40,97.82}, 95. {55.60,112.28},{ 1.26,47.44},{73.54,125.72},{49.81,102.51}, 96. {97.31,170.44},{89.64,169.02},{91.89,169.61},{45.89,107.66}, 97. {99.90,168.98},{39.27,106.49},{63.61,118.52},{60.86,115.28}, 98. { 3.11,48.70},{61.64,119.97},{28.18,81.53},{ 5.20,43.98}, 99. {51.76,117.25},{57.63,127.76},{78.84,153.80},{10.54,48.23}, 100. {52.51,115.61},{ 4.69,28.30},{93.93,164.59},{69.70,141.08}, 101. {57.13,118.40},{23.45,86.44},{44.59,132.03},{31.91,98.03}, 102. {44.44,103.85},{ 9.46,53.83},{92.57,190.48},{36.29,95.15}, 103. {32.06,92.48},{86.16,138.27},{49.86,112.94},{96.14,181.82}, 104. {16.05,59.03},{52.13,102.89},{51.27,109.44},{67.94,117.86}, 105. {86.77,158.44},{74.20,143.83},{39.13,93.09},{66.00,137.66}, 106. {22.48,55.93},{54.90,116.64},{51.71,95.82},{36.07,85.96}, 107. { 5.38,59.94},{84.49,160.75},{28.93,82.85},{89.92,183.08}, 108. { 1.83,61.19},{26.71,73.35},{49.96,93.82},{13.56,68.73}, 109. {26.93,70.29},{85.19,165.22},{74.31,148.26},{44.90,107.17}, 110. {81.60,159.66},{86.68,150.41},{ 8.77,66.43},{75.18,159.06}, 111. { 8.86,45.78},{66.61,131.48},{ 5.80,40.46},{84.73,169.37}, 112. {35.34,85.87},{97.62,161.97},{15.22,73.73},{77.52,152.32}, 113. { 2.96,45.12},{60.66,127.28},{66.50,131.20},{72.85,141.01}, 114. {65.90,130.75},{43.44,101.16},{11.06,52.18},{14.77,62.27}, 115. {77.52,159.43},{47.99,126.74},{63.43,110.36},{50.26,113.37}, 116. {95.43,198.57},{24.53,57.71},{12.87,54.28},{63.79,135.32}, 117. {56.58,110.85},{59.10,121.23},{99.99,194.48},{34.56,83.22}, 118. {98.83,173.25},{ 7.73,70.43},{34.44,103.18},{96.37,169.25}, 119. {50.19,118.06},{84.66,175.92},{79.01,142.91},{99.12,174.11}, 120. {61.97,120.41},{81.93,146.96},{36.18,71.82},{ 3.47,51.04}, 121. { 0.07,52.79},{82.02,158.42},{37.02,94.32},{26.83,77.25}, 122. {87.90,166.44},{22.65,76.05},{ 8.80,53.89},{53.12,109.05}, 123. {64.69,147.19},{55.56,107.54},{98.42,175.01},{ 4.99,30.66}, 124. {63.23,122.97},{56.44,121.49},{ 8.58,69.79},{79.38,149.69}, 125. {55.64,122.77},{84.76,142.77},{29.21,85.68},{18.11,70.14}, 126. {33.15,85.38},{11.30,44.68},{83.67,163.51},{23.43,101.91}, 127. {29.59,81.85},{19.90,75.18},{26.05,89.95},{61.05,129.50}, 128. { 6.27,68.93},{96.95,191.69},{82.78,145.43},{73.84,136.65}, 129. {60.44,140.08},{12.67,65.86},{ 2.20,36.68},{ 6.16,54.50}, 130. {35.28,95.48},{83.01,152.33},{64.33,121.41},{91.72,162.34}, 131. {72.62,130.77},{55.31,121.78},{83.52,164.02},{ 1.45,24.37}, 132. {84.96,164.70},{ 6.85,59.62},{89.95,158.91},{57.41,133.29}, 133. { 6.13,58.72},{78.33,125.38},{65.15,121.67},{19.10,81.33}, 134. {17.43,58.99},{60.92,142.74},{69.37,142.70},{ 7.71,53.52}, 135. {38.81,101.59},{18.31,65.35},{41.05,90.89},{29.84,96.28}, 136. {27.65,88.04},{11.74,30.21},{72.14,150.80},{59.92,123.34}, 137. {20.59,51.65},{73.54,147.09},{25.04,52.81},{21.44,78.92}, 138. { 8.30,68.18},{38.27,103.50},{76.73,135.46},{13.41,42.84}, 139. { 9.77,50.17},{31.79,84.64},{11.63,47.87},{81.10,154.34}, 140. {32.88,86.39},{83.66,156.16},{42.84,101.97},{92.23,181.69}, 141. {56.62,128.67},{21.57,72.57},{28.42,76.81},{78.49,151.43}, 142. {34.76,87.12},{95.65,153.86},{48.99,114.03},{22.97,82.24}, 143. {96.82,167.98},{55.42,93.17},{59.22,121.38},{41.66,101.95}, 144. {77.91,166.28},{12.14,54.69},{91.24,171.96},{46.22,106.58}, 145. {98.54,161.56},{46.98,104.41},{60.09,131.79},{67.83,119.99}, 146. {41.09,85.10},{70.10,135.73},{36.99,91.78},{10.72,53.99}, 147. {50.06,91.80},{18.39,60.80},{26.53,85.06},{76.65,154.26}, 148. { 0.02,34.66},{37.56,104.13},{91.48,177.82},{ 0.31,41.81}, 149. {93.20,166.21},{ 6.94,43.44},{85.38,159.15},{90.74,139.02}, 150. {83.80,157.57},{39.34,92.96},{95.49,170.28},{77.55,145.57}, 151. {33.05,88.61},{ 4.28,43.15},{95.89,183.02},{36.18,97.42}, 152. {94.88,160.54},{ 5.18,55.47},{24.04,66.69},{50.93,118.02}, 153. {11.82,58.02},{ 4.55,62.84},{71.87,136.43},{ 1.64,46.82}, 154. {86.64,157.08},{76.78,135.11},{17.94,59.70},{41.81,96.43}, 155. {89.21,170.11},{75.89,139.66},{90.73,160.27},{45.39,101.75}, 156. {61.07,135.71},{ 7.53,48.91},{26.32,74.50},{89.44,164.11}, 157. {79.85,162.30},{55.91,104.02},{89.47,172.39},{88.09,154.40}, 158. {87.63,148.99},{24.43,74.77},{49.28,103.08},{86.49,165.82}, 159. {93.79,148.29},{93.25,170.91},{59.55,126.98},{ 9.24,64.46}, 160. {73.82,134.35},{76.07,152.94},{77.44,148.98},{87.01,161.01}, 161. {72.52,154.33},{21.77,97.29},{47.74,111.47},{17.34,68.45}, 162. {39.75,119.04},{84.78,160.15},{62.57,121.76},{70.20,146.15}, 163. { 7.08,50.89},{60.33,97.44},{29.90,89.84},{41.82,95.48}, 164. {38.27,101.32},{ 9.60,53.87},{84.69,166.21},{97.64,177.02}, 165. {73.96,145.64},{11.68,73.29},{31.64,75.20},{44.12,119.54}, 166. {29.91,99.48},{62.85,117.28},{65.55,123.70},{78.66,161.94}, 167. {71.06,158.16},{71.17,147.50},{12.49,63.11},{62.47,146.21}, 168. { 5.52,64.77},{19.39,81.84},{90.73,177.46},{46.94,101.25}, 169. {35.30,92.84},{25.31,80.86},{29.29,95.30},{79.68,160.63}, 170. {65.64,143.96},{81.97,173.79},{58.68,123.92},{37.35,94.26}, 171. {81.31,146.56},{10.82,34.36},{72.19,152.74},{96.81,157.39}, 172. {37.82,84.01},{26.02,85.45},{49.68,99.80},{63.61,134.18}, 173. {78.45,135.62},{28.06,99.20},{29.49,76.17},{38.73,90.19}, 174. {66.67,128.82},{ 4.14,43.63},{75.01,155.68},{12.38,39.83}, 175. {64.06,126.87},{77.44,154.01},{30.67,89.38},{30.38,85.81}, 176. {98.17,180.45},{72.62,131.67},{18.80,74.37},{56.99,112.44}, 177. {80.45,152.36},{43.87,96.62},{10.95,79.63},{ 8.13,48.84}, 178. {96.47,180.58},{57.99,139.71},{29.81,82.98},{ 7.69,59.04}, 179. {60.75,113.11},{61.26,127.28},{29.91,74.20},{72.81,130.29}, 180. {97.22,186.22},{16.10,67.14},{45.22,82.91},{59.90,136.15}, 181. {50.86,99.99},{40.09,89.91},{38.69,87.12},{38.22,86.34}, 182. {82.85,160.54},{44.59,114.27},{ 6.39,49.22},{53.02,118.97}, 183. {67.10,132.43},{87.17,167.48},{61.46,109.49},{79.66,163.17}, 184. {40.28,88.74},{81.76,164.45},{10.26,58.64},{14.58,81.13}, 185. {85.30,184.26},{64.06,132.71},{ 5.55,56.52},{96.97,187.38}, 186. {92.22,174.95},{42.45,100.51},{30.79,81.24},{ 4.25,61.71}, 187. {47.15,104.16},{35.87,86.39},{81.62,152.64},{42.46,95.25}, 188. {66.69,137.47},{33.21,84.65},{23.42,84.12},{99.30,187.76}, 189. {19.15,77.26},{17.74,70.35},{87.90,170.12},{47.01,118.00}, 190. {78.63,155.19},{92.38,163.60},{72.75,153.70},{79.92,138.69}, 191. {21.94,78.76},{55.51,120.91},{27.08,57.31},{12.83,45.59}, 192. {48.22,103.52},{35.64,87.26},{59.90,119.91},{50.05,110.55}, 193. { 0.23,41.68},{66.03,129.51},{42.67,95.15},{37.78,103.08}, 194. { 3.06,43.68},{53.80,102.89},{ 9.78,51.90},{94.94,185.83}, 195. {31.69,105.92},{70.50,123.84},{ 5.52,51.03},{ 0.93,47.63}, 196. {68.12,146.17},{ 6.86,51.21},{ 4.60,42.38},{72.98,138.03}, 197. {58.59,125.26},{40.21,88.92},{12.51,41.25},{31.12,65.03}, 198. {75.68,143.15},{74.02,141.52},{ 5.61,50.98},{82.39,162.02}, 199. {28.07,65.38},{71.22,145.28},{44.22,99.25},{72.03,123.62}, 200. {45.88,95.49},{76.37,136.85},{29.19,81.03},{63.46,142.45}, 201. {49.44,100.25},{81.71,132.07},{83.34,150.05},{38.88,93.91}, 202. {86.01,172.44},{51.32,110.43},{86.82,154.64},{70.02,140.53}, 203. {26.43,72.25},{34.48,91.08},{30.41,80.09},{24.77,80.43}, 204. { 8.14,53.39},{18.88,70.38},{26.90,73.96},{94.43,173.51}, 205. {24.45,62.18},{56.07,111.25},{66.96,136.85},{93.78,188.93}, 206. {75.18,144.91},{18.22,43.66},{97.70,170.91},{34.25,95.34}, 207. {12.16,53.60},{88.48,164.67},{81.58,176.86},{81.96,151.45}, 208. {50.13,109.26},{44.20,90.81},{52.84,121.91},{17.30,76.64}, 209. {53.60,120.26},{32.01,84.79},{72.56,149.71},{19.15,55.05}, 210. {78.26,164.32},{ 9.84,56.96},{ 2.48,50.11},{50.84,134.33}, 211. {90.65,164.89},{35.58,81.09},{72.54,151.27},{54.39,119.73}, 212. {44.15,105.72},{74.88,145.20},{86.66,158.64},{17.79,67.78}, 213. {54.84,115.87},{99.10,173.71},{93.02,174.29},{23.52,83.88}, 214. {19.56,68.81},{24.03,79.83},{11.73,35.38},{ 3.82,37.84}, 215. {61.92,130.48},{77.02,139.38},{91.29,161.65},{98.09,162.25}, 216. { 0.49,36.07},{75.44,138.86},{ 4.32,59.64},{79.99,143.62}, 217. {13.43,47.42},{44.44,110.46},{25.03,71.30},{71.95,147.86}, 218. {78.51,152.59},{ 3.43,34.40},{55.28,115.97},{88.77,165.45}, 219. {15.43,82.65},{99.09,179.00},{79.77,143.93},{52.73,116.02}, 220. {52.40,109.06},{37.24,83.30},{31.90,80.93},{68.13,127.42}, 221. {70.63,127.66},{55.84,132.35},{39.95,99.29},{ 6.84,40.38}, 222. {66.47,117.89},{20.53,83.00},{82.22,147.22},{23.74,73.03}, 223. {77.83,159.15},{11.29,64.73},{49.15,104.01},{52.54,105.95}, 224. {93.36,160.71},{51.35,106.68},{28.56,83.86},{78.27,147.91}, 225. { 0.25,50.56},{59.93,120.88},{ 8.58,49.04},{74.24,134.96}, 226. {51.22,98.10},{24.91,74.31},{87.07,160.74},{52.25,105.22}, 227. {91.43,152.78},{ 8.10,59.46},{94.97,178.64},{88.81,178.24}, 228. {88.45,150.89},{21.60,76.09},{70.62,122.85},{99.65,168.36}, 229. {73.32,142.58},{13.18,71.21},{37.26,88.09},{79.15,142.60}, 230. {20.05,91.09},{33.64,87.49},{21.84,64.04},{49.12,116.82}, 231. {52.57,125.37},{42.43,93.45},{22.54,94.46},{82.51,165.14}, 232. {77.64,132.17},{32.25,83.61},{10.77,55.92},{71.34,133.82}, 233. {60.85,127.38},{22.68,79.72},{30.62,77.71},{81.90,161.50}, 234. {10.22,47.19},{26.58,57.16},{43.66,113.00},{90.69,145.82}, 235. {12.64,58.91},{85.90,154.22},{18.03,53.36},{84.49,144.57}, 236. {87.51,169.54},{92.50,170.96},{51.99,123.08},{45.16,108.57}, 237. {71.40,137.44},{58.36,121.84},{76.06,143.46},{42.17,104.52}, 238. { 2.57,50.10},{11.44,41.77},{71.09,143.92},{88.92,151.09}, 239. {92.79,177.42},{90.72,157.64},{66.11,141.52},{ 2.33,38.80}, 240. {76.26,158.72},{76.52,150.07},{70.31,132.13},{52.77,119.85}, 241. {99.59,176.94},{ 8.16,56.11},{99.29,190.79},{25.00,77.22}, 242. {13.45,63.42},{17.35,70.05},{ 4.16,35.31},{86.57,152.50}, 243. {88.57,168.14},{67.96,123.67},{72.36,142.41},{10.95,73.06}, 244. {78.45,163.31},{71.69,139.46},{82.78,157.91},{80.14,161.51}, 245. {60.33,133.76},{ 9.44,46.00},{68.21,163.48},{30.78,88.27}, 246. {38.74,105.24},{19.52,62.94},{49.03,105.82},{76.01,138.95}, 247. {71.08,165.10},{49.97,108.11},{75.15,145.38},{ 5.20,62.33}, 248. {97.13,188.35},{87.18,176.82},{42.70,96.24},{62.98,126.92}, 249. {96.09,175.08},{90.77,190.99},{71.68,124.23},{15.67,61.12}, 250. {95.37,178.30},{40.64,83.70},{22.64,71.16},{30.22,105.64}, 251. {18.96,77.17},{56.47,98.36},{36.53,84.99},{13.11,73.50}, 252. {32.04,80.23},{72.49,135.67},{54.33,126.59},{13.54,51.56}, 253. {14.77,57.56},{24.09,90.04},{32.43,86.80},{ 3.82,43.03}, 254. {81.10,163.58},{45.39,96.22},{57.29,115.98},{76.10,151.06}, 255. { 7.74,56.38},{48.95,108.35},{40.07,101.13},{81.91,144.88}, 256. {64.47,124.53},{70.83,129.18},{ 7.05,44.06},{36.46,86.68}, 257. {32.53,89.07},{32.88,77.92},{ 6.62,29.48},{28.87,76.01}, 258. {37.36,90.05},{72.25,136.07},{81.47,173.47},{ 4.20,47.41}, 259. {98.64,166.84},{46.61,109.47},{45.38,88.61},{95.41,169.40}, 260. {66.63,122.42},{98.96,176.41},{77.60,166.70},{39.53,93.96}, 261. {73.29,138.23},{87.99,159.87},{34.35,91.01},{33.30,78.44}, 262. {29.29,78.05},{89.99,153.84},{ 3.90,31.79},{ 2.74,28.45}, 263. {74.07,144.41},{59.60,135.80},{83.19,154.17},{33.14,71.48}, 264. {71.18,127.25},{59.10,126.89},{14.88,60.57},{46.36,122.43}, 265. {97.68,166.02},{47.91,110.00},{94.43,185.03},{25.13,73.98}, 266. {30.66,83.04},{47.36,100.33},{20.03,57.00},{38.53,77.34}, 267. {53.29,122.85},{77.72,146.52},{23.42,84.64},{96.85,170.99}, 268. { 8.49,68.27},{71.67,127.11},{84.22,158.75},{35.25,87.63}, 269. {74.00,140.25},{32.42,91.48},{88.91,156.95},{88.11,163.18}, 270. {60.49,132.55},{63.94,149.75},{95.21,172.49},{ 2.14,37.70}, 271. { 2.33,36.29},{24.38,83.62},{87.11,162.19},{37.16,85.65}, 272. {81.57,152.86},{49.26,116.82},{54.72,108.56},{65.82,132.02}, 273. {10.93,47.63},{71.92,111.28},{12.67,36.23},{67.35,101.56}, 274. {86.25,169.18},{97.89,194.43},{40.63,106.58},{73.87,135.71} 275. }; 276. **double** residual\_error(**double** x, **double** y, **double** m, **double** c) { 277. **double** e = (m \* x) + c - y; 278. **return** e \* e; 279. } 281. \_\_device\_\_ **double** d\_residual\_error(**double** x, **double** y, **double** m, **double** c) { 282. **double** e = (m \* x) + c - y; 283. **return** e \* e; 284. } 286. **double** rms\_error(**double** m, **double** c) { 287. **int** i; 288. **double** mean; 289. **double** error\_sum = 0; 291. **for**(i=0; i<n\_data; i++) { 292. error\_sum += residual\_error(data[i].x, data[i].y, m, c); 293. } 295. mean = error\_sum / n\_data; 297. **return** sqrt(mean); 298. } 300. \_\_global\_\_ **void** d\_rms\_error(**double** \*m, **double** \*c, **double** \*error\_sum\_arr, point\_t \*d\_data) { 302. **int** i = threadIdx.x + blockIdx.x \* blockDim.x; 304. error\_sum\_arr[i] = d\_residual\_error(d\_data[i].x, d\_data[i].y, \*m, \*c); 305. } 307. **int** time\_difference(**struct** timespec \*start, **struct** timespec \*finish, 308. **long** **long** **int** \*difference) { 309. **long** **long** **int** ds =  finish->tv\_sec - start->tv\_sec; 310. **long** **long** **int** dn =  finish->tv\_nsec - start->tv\_nsec; 312. **if**(dn < 0 ) { 313. ds--; 314. dn += 1000000000; 315. } 316. \*difference = ds \* 1000000000 + dn; 317. **return** !(\*difference > 0); 318. } 320. **int** main() { 321. **int** i; 322. **double** bm = 1.3; 323. **double** bc = 10; 324. **double** be; 325. **double** dm[8]; 326. **double** dc[8]; 327. **double** e[8]; 328. **double** step = 0.01; 329. **double** best\_error = 999999999; 330. **int** best\_error\_i; 331. **int** minimum\_found = 0; 333. **double** om[] = {0,1,1, 1, 0,-1,-1,-1}; 334. **double** oc[] = {1,1,0,-1,-1,-1, 0, 1}; 336. **struct** timespec start, finish; 337. **long** **long** **int** time\_elapsed;  340. clock\_gettime(CLOCK\_MONOTONIC, &start); 342. cudaError\_t error;  345. **double** \*d\_dm; 346. **double** \*d\_dc; 347. **double** \*d\_error\_sum\_arr; 348. point\_t \*d\_data; 350. be = rms\_error(bm, bc);  353. error = cudaMalloc(&d\_dm, (**sizeof**(**double**) \* 8)); 354. **if**(error){ 355. fprintf(stderr, "cudaMalloc on d\_dm returned %d %s\n", error, 356. cudaGetErrorString(error)); 357. exit(1); 358. }  361. error = cudaMalloc(&d\_dc, (**sizeof**(**double**) \* 8)); 362. **if**(error){ 363. fprintf(stderr, "cudaMalloc on d\_dc returned %d %s\n", error, 364. cudaGetErrorString(error)); 365. exit(1); 366. }  369. error = cudaMalloc(&d\_error\_sum\_arr, (**sizeof**(**double**) \* 1000)); 370. **if**(error){ 371. fprintf(stderr, "cudaMalloc on d\_error\_sum\_arr returned %d %s\n", error, 372. cudaGetErrorString(error)); 373. exit(1); 374. }  377. error = cudaMalloc(&d\_data, **sizeof**(data)); 378. **if**(error){ 379. fprintf(stderr, "cudaMalloc on d\_data returned %d %s\n", error, 380. cudaGetErrorString(error)); 381. exit(1); 382. } 384. **while**(!minimum\_found) { 385. **for**(i=0;i<8;i++) { 386. dm[i] = bm + (om[i] \* step); 387. dc[i] = bc + (oc[i] \* step); 388. }  391. error = cudaMemcpy(d\_dm, dm, (**sizeof**(**double**) \* 8), cudaMemcpyHostToDevice); 392. **if**(error){ 393. fprintf(stderr, "cudaMemcpy to d\_dm returned %d %s\n", error, 394. cudaGetErrorString(error)); 395. }  398. error = cudaMemcpy(d\_dc, dc, (**sizeof**(**double**) \* 8), cudaMemcpyHostToDevice); 399. **if**(error){ 400. fprintf(stderr, "cudaMemcpy to d\_dc returned %d %s\n", error, 401. cudaGetErrorString(error)); 402. }  405. error = cudaMemcpy(d\_data, data, **sizeof**(data), cudaMemcpyHostToDevice); 406. **if**(error){ 407. fprintf(stderr, "cudaMemcpy to d\_data returned %d %s\n", error, 408. cudaGetErrorString(error)); 409. } 411. **for**(i=0;i<8;i++) { 413. **double** h\_error\_sum\_arr[1000]; 414. **double** error\_sum\_total; 415. **double** error\_sum\_mean; 416. d\_rms\_error <<<100,10>>>(&d\_dm[i], &d\_dc[i], d\_error\_sum\_arr, d\_data); 417. cudaThreadSynchronize(); 418. error = cudaMemcpy(&h\_error\_sum\_arr, d\_error\_sum\_arr, (**sizeof**(**double**) \* 1000), cudaMemcpyDeviceToHost); 419. **if**(error){ 420. fprintf(stderr, "cudaMemcpy to error\_sum returned %d %s\n", error, 421. cudaGetErrorString(error)); 422. } 423. **for**(**int** j=0; j<n\_data; j++) { 424. error\_sum\_total += h\_error\_sum\_arr[j]; 425. } 427. error\_sum\_mean = error\_sum\_total / n\_data; 428. e[i] = sqrt(error\_sum\_mean); 430. **if**(e[i] < best\_error) { 431. best\_error = e[i]; 432. best\_error\_i = i; 433. } 435. error\_sum\_total = 0; 436. }  439. **if**(best\_error < be) { 440. be = best\_error; 441. bm = dm[best\_error\_i]; 442. bc = dc[best\_error\_i]; 443. } **else** { 444. minimum\_found = 1; 445. } 446. } 448. error = cudaFree(d\_dm); 449. **if**(error){ 450. fprintf(stderr, "cudaFree on d\_dm returned %d %s\n", error, 451. cudaGetErrorString(error)); 452. exit(1); 453. } 455. error = cudaFree(d\_dc); 456. **if**(error){ 457. fprintf(stderr, "cudaFree on d\_dc returned %d %s\n", error, 458. cudaGetErrorString(error)); 459. exit(1); 460. } 462. error = cudaFree(d\_data); 463. **if**(error){ 464. fprintf(stderr, "cudaFree on d\_data returned %d %s\n", error, 465. cudaGetErrorString(error)); 466. exit(1); 467. } 469. error = cudaFree(d\_error\_sum\_arr); 470. **if**(error){ 471. fprintf(stderr, "cudaFree on d\_error\_sum\_arr returned %d %s\n", error, 472. cudaGetErrorString(error)); 473. exit(1); 474. } 476. printf("minimum m,c is %lf,%lf with error %lf\n", bm, bc, be); 478. clock\_gettime(CLOCK\_MONOTONIC, &finish); 480. time\_difference(&start, &finish, &time\_elapsed); 482. printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed, 483. (time\_elapsed/1.0e9)); 485. **return** 0; 486. } |

**Mean time comparing with original, multithread and cuda version.**

|  |  |  |  |
| --- | --- | --- | --- |
| Run time | Original | Multithread | CUDA |
| 1 |  |  | 665680182 |
| 2 |  |  | 673435697 |
| 3 |  |  | 647885453 |
| 4 |  |  | 647003877 |
| 5 |  |  | 647861207 |
| 6 |  |  | 645532050 |
| 7 |  |  | 645304800 |
| 8 |  |  | 648976199 |
| 9 |  |  | 644777904 |
| 10 |  |  | 648446846 |
|  |  |  | 651490421.5 |

# 3 MPI

## 3.1 Password cracking

|  |
| --- |
| 1. #include <stdlib.h> 2. #include <stdio.h> 3. #include <string.h> 4. #include <time.h> 5. #include <crypt.h> 6. #include <mpi.h> 8. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 9. To compile: mpicc -o Mpi\_pw Password\_mpi.c -lrt -lcrypt 10. To run 3 processes on this computer:mpirun -n 3 ./Mpi\_pw 11. Name:1928720\_Adhikar\_chaudhary 12. @University of wolverhamption 13. Dr Kevan Buckley, University of Wolverhampton, 2018 14. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/   18. **int** n\_passwords = 4; 19. pthread\_t t1, t2; 20. **char** \*encrypted\_passwords[] = { 22. "$6$KB$0G24VuNaA9ApVG4z8LkI/OOr9a54nBfzgQjbebhqBZxMHNg0HiYYf1Lx/HcGg6q1nnOSArPtZYbGy7yc5V.wP/", 23. "$6$KB$uTdNob3WgiJel6BANUV7Yc94ra1yxDGopfCSu4zRuiZwNHzqNbgTkbtFOXNZQWB0q5ItG9tr3iI4Fq5Cp5tTA.", 24. "$6$KB$Wq1dzyVEN2NNJvf6v0XFA/Nsx8HDDmwiX5RVNrPfPK6.ciNVJKEUG1urR0K2wrsDy2xQxbqrbmM5QmH.x6Im71", 25. "$6$KB$wbRHzUBWMCWXZDeiA0VAnCjZGiFEeK/RME1eHhYp5Uc3nGLJzrGrmeYYftsFLImL.8JMhFDbnRetMR7KLM0qh."  28. }; 30. **void** substr(**char** \*dest, **char** \*src, **int** start, **int** length){ 31. memcpy(dest, src + start, length); 32. \*(dest + length) = '\0'; 33. } 35. **void** kernel\_function1(**char** \*salt\_and\_encrypted){ 36. **int** x, y, z; 37. **char** salt[7]; 39. **char** plain[7]; 40. **char** \*enc; 41. **int** count = 0; 43. substr(salt, salt\_and\_encrypted, 0, 6); 45. **for**(x='A'; x<='M'; x++){ 46. **for**(y='A'; y<='Z'; y++){ 47. **for**(z=0; z<=99; z++){ 48. printf("Instance 1:"); 49. sprintf(plain, "%c%c%02d",x, y, z); 50. enc = (**char** \*) crypt(plain, salt); 51. count++; 52. **if**(strcmp(salt\_and\_encrypted, enc) == 0){ 53. printf("#%-8d%s %s\n", count, plain, enc); 54. } **else** { 55. printf(" %-8d%s %s\n", count, plain, enc); 56. } 57. } 58. } 59. } 60. printf("%d solutions explored\n", count); 61. } 62. **void** kernel\_function2(**char** \*salt\_and\_encrypted){ 63. **int** x, y, z; 64. **char** salt[7]; 66. **char** plain[7]; 67. **char** \*enc; 68. **int** count = 0; 70. substr(salt, salt\_and\_encrypted, 0, 6); 72. **for**(x='N'; x<='Z'; x++){ 73. **for**(y='A'; y<='Z'; y++){ 74. **for**(z=0; z<=99; z++){ 75. printf("Instance 2:"); 76. sprintf(plain, "%c%c%02d",x, y, z); 77. enc = (**char** \*) crypt(plain, salt); 78. count++; 79. **if**(strcmp(salt\_and\_encrypted, enc) == 0){ 80. printf("#%-8d%s %s\n", count, plain, enc); 81. } **else** { 82. printf(" %-8d%s %s\n", count, plain, enc); 83. } 84. } 85. } 86. } 87. printf("%d solutions explored\n", count); 88. }  91. **int** time\_difference(**struct** timespec \*start, **struct** timespec \*finish, 92. **long** **long** **int** \*difference) { 93. **long** **long** **int** ds =  finish->tv\_sec - start->tv\_sec; 94. **long** **long** **int** dn =  finish->tv\_nsec - start->tv\_nsec; 96. **if**(dn < 0 ) { 97. ds--; 98. dn += 1000000000; 99. } 100. \*difference = ds \* 1000000000 + dn; 101. **return** !(\*difference > 0); 102. } 104. **int** main(**int** argc, **char**\*\* argv) { 105. **struct** timespec start, finish; 106. **long** **long** **int** time\_elapsed; 108. clock\_gettime(CLOCK\_MONOTONIC, &start);  111. **int** size, rank; 112. **int** i; 114. MPI\_Init(NULL, NULL); 115. MPI\_Comm\_size(MPI\_COMM\_WORLD, &size); 116. MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank); 117. **if**(size != 3) { 118. **if**(rank == 0) { 119. printf("3 processes running\n"); 120. } 121. } **else** { 122. **if**(rank ==0){ 124. **int** x; 126. MPI\_Send(&x, 1, MPI\_INT, 1, 0, MPI\_COMM\_WORLD); 127. MPI\_Send(&x, 1, MPI\_INT, 2, 0, MPI\_COMM\_WORLD);   131. } **else** **if**(rank==1){ 132. **int** number; 133. MPI\_Recv(&number, 1, MPI\_INT, 0, 0, MPI\_COMM\_WORLD, 134. MPI\_STATUS\_IGNORE); 135. **for**(i=0;i<n\_passwords;i<i++) { 136. kernel\_function1(encrypted\_passwords[i]); 137. } 138. } 139. **else**{ 140. **int** number; 141. MPI\_Recv(&number, 1, MPI\_INT, 0, 0, MPI\_COMM\_WORLD, 142. MPI\_STATUS\_IGNORE); 143. **for**(i=0;i<n\_passwords;i<i++) { 144. kernel\_function2(encrypted\_passwords[i]); 145. } 146. } 147. } 148. MPI\_Finalize(); 149. clock\_gettime(CLOCK\_MONOTONIC, &finish); 150. time\_difference(&start, &finish, &time\_elapsed); 151. printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed, 152. (time\_elapsed/1.0e9)); 154. **return** 0; 155. } |

**Insert a table that shows running times for the original and MPI versions**

|  |  |  |
| --- | --- | --- |
| Run time | Original time Nano second | MPI Time Nano second |
| 1 | 515327416109 | 573711456 |
| 2 | 513725913738 | 153355679 |
| 3 | 516503597715 | 141454960 |
| 4 | 510567181476 | 144999512 |
| 5 | 511684660337 | 146447144 |
| 6 | 510520288292 | 145109899 |
| 7 | 504789551079 | 144568517 |
| 8 | 505053580512 | 144401323 |
| 9 | 505854465487 | 143879801 |
| 10 | 506524673766 | 143623426 |
| Mean | 510055132851.1 Nano second | 188155171.7 Nano Second |

510055132851.1 Nano second = 510.055132851099984 Second

188155171.7 Nano Second = 0.1881551717 second

**Result Analysis**

## 3.2 Image processing

|  |
| --- |
| 1. #include <mpi.h> 2. #include <stdio.h> 3. #include <stdlib.h> 4. #include <time.h> 5. #include <GL/glut.h> 6. #include <GL/gl.h> 7. #include <malloc.h> 8. #include <signal.h> 9. #include <math.h> 10. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 11. To compile adapt the code below wo match your filenames: 12. mpicc -o Mpi\_Image\_processing Mpi\_Image\_processing.c -lglut -lGL -lm 14. To run 15. mpirun -n 5 -quiet ./Mpi\_Image\_processing 17. Name:1928720\_Adhikar\_chaudhary 18. @University of wolverhamption 20. \* Dr Kevan Buckley, University of Wolverhampton, 2018 21. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  24. #define width 100 25. #define height 72 27. **int** first\_index, last\_index; 28. unsigned **char** image[], results[width \* height]; 30. **void** detect\_edges(unsigned **char** \*in, unsigned **char** \*out) { 31. **int** i; 32. **int** n\_pixels = width \* height; 34. **for**(i=0;i<n\_pixels;i++) { 35. **int** x, y; // the pixel of interest 36. **int** a, d, s, h; // the pixels adjacent to x,y used for the calculation 37. **int** r; // the result of calculate 39. y = i / width; 40. x = i - (width \* y); 42. **if** (x == 0 || y == 0 || x == width - 1 || y == height - 1) { 43. results[i] = 0; 44. } **else** { 45. a = i + width; 46. d = i - 1; 47. s = i + 1; 48. h = i - width; 50. r = (in[i] \* 4) + (in[a] \* -1) + (in[d] \* -1) + (in[s] \* -1) 51. + (in[h] \* -1); 53. **if** (r > 0) { // if the result is positive this is an edge pixel 54. out[i] = 255; 55. } **else** { 56. out[i] = 0; 57. } 58. } 59. } 60. } 62. **void** tidy\_and\_exit() { 63. exit(0); 64. } 66. **void** sigint\_callback(**int** signal\_number){ 67. printf("\nInterrupt from keyboard\n"); 68. tidy\_and\_exit(); 69. } 71. **static** **void** display() { 72. glClear(GL\_COLOR\_BUFFER\_BIT); 73. glRasterPos4i(-1, -1, 0, 1); 74. glDrawPixels(width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, image); 75. glRasterPos4i(0, -1, 0, 1); 76. glDrawPixels(width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, results); 77. glFlush(); 78. } 80. **static** **void** key\_pressed(unsigned **char** key, **int** x, **int** y) { 81. **switch**(key){ 82. **case** 27: // escape 83. tidy\_and\_exit(); 84. **break**; 86. **case** 'e': // press e to exit 87. tidy\_and\_exit(); 88. **break**; 90. **default**: 91. printf("\nPress escape or 'e' to exit\n"); 92. **break**; 93. } 94. } 96. **int** time\_difference(**struct** timespec \*start, **struct** timespec \*finish, 97. **long** **long** **int** \*difference) { 98. **long** **long** **int** ds =  finish->tv\_sec - start->tv\_sec; 99. **long** **long** **int** dn =  finish->tv\_nsec - start->tv\_nsec; 101. **if**(dn < 0 ) { 102. ds--; 103. dn += 1000000000; 104. } 105. \*difference = ds \* 1000000000 + dn; 106. **return** !(\*difference > 0); 107. } 108. **int** main(**int** argc, **char** \*\*argv) { 109. signal(SIGINT, sigint\_callback); 111. **int** size, rank; 113. MPI\_Init(NULL, NULL); 114. MPI\_Comm\_size(MPI\_COMM\_WORLD, &size); 115. MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank); 116. **if**(size != 5) { 117. **if**(rank == 0) { 118. printf("Five process\n"); 119. } 120. } **else** { 121. **if**(rank ==0){ 122. **struct** timespec start, finish; 123. **long** **long** **int** time\_elapsed; 124. clock\_gettime(CLOCK\_MONOTONIC, &start); 125. MPI\_Send(&results[0], 1800, MPI\_UNSIGNED\_CHAR, 1, 0, MPI\_COMM\_WORLD); 126. MPI\_Send(&results[1800], 1800, MPI\_UNSIGNED\_CHAR, 2, 0, MPI\_COMM\_WORLD); 127. MPI\_Send(&results[3600], 1800, MPI\_UNSIGNED\_CHAR, 3, 0, MPI\_COMM\_WORLD); 128. MPI\_Send(&results[5400], 1800, MPI\_UNSIGNED\_CHAR, 4, 0, MPI\_COMM\_WORLD); 130. MPI\_Recv(&results[0], 1800, MPI\_UNSIGNED\_CHAR, 1, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE); 131. MPI\_Recv(&results[1800], 1800,MPI\_UNSIGNED\_CHAR, 2, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE); 132. MPI\_Recv(&results[3600], 1800,MPI\_UNSIGNED\_CHAR, 3, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE); 133. MPI\_Recv(&results[5400], 1800,MPI\_UNSIGNED\_CHAR, 4, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE); 135. clock\_gettime(CLOCK\_MONOTONIC, &finish); 136. time\_difference(&start, &finish, &time\_elapsed); 137. printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed,(time\_elapsed/1.0e9)); 139. glutInit(&argc, argv); 140. glutInitWindowSize(width \* 2,height); 141. glutInitDisplayMode(GLUT\_SINGLE | GLUT\_LUMINANCE); 143. glutCreateWindow("6CS005 Image Progessing Courework"); 144. glutDisplayFunc(display); 145. glutKeyboardFunc(key\_pressed); 146. glClearColor(0.0, 1.0, 0.0, 1.0); 148. glutMainLoop(); 150. tidy\_and\_exit();  153. } **else** { 154. **if**(rank == 1){ 156. first\_index = 0; 157. last\_index = 1799; 159. MPI\_Recv(&results[0], 1800, MPI\_UNSIGNED\_CHAR, 0, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE); 160. detect\_edges(image, results); 161. MPI\_Send(&results[0], 1800, MPI\_UNSIGNED\_CHAR, 0, 0, MPI\_COMM\_WORLD); 162. } 163. **else** **if**(rank == 2){ 164. first\_index = 1800; 165. last\_index = 3599; 167. MPI\_Recv(&results[1800], 1800, MPI\_UNSIGNED\_CHAR, 0, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE); 168. detect\_edges(image, results); 169. MPI\_Send(&results[1800], 1800, MPI\_UNSIGNED\_CHAR, 0, 0, MPI\_COMM\_WORLD); 170. } 171. **else** **if**(rank == 3){ 172. first\_index = 3600; 173. last\_index = 5399; 175. MPI\_Recv(&results[3600], 1800, MPI\_UNSIGNED\_CHAR, 0, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE); 176. detect\_edges(image, results); 177. MPI\_Send(&results[3600], 1800, MPI\_UNSIGNED\_CHAR, 0, 0, MPI\_COMM\_WORLD); 179. } 180. **else** **if**(rank == 4){ 181. first\_index = 5400; 182. last\_index = 7199; 184. MPI\_Recv(&results[5400], 1800, MPI\_UNSIGNED\_CHAR, 0, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE); 185. detect\_edges(image, results); 186. MPI\_Send(&results[5400], 1800, MPI\_UNSIGNED\_CHAR, 0, 0, MPI\_COMM\_WORLD); 187. } 188. } 189. } 190. MPI\_Finalize(); 191. **return** 0; 192. } 194. unsigned **char** image[] = {0,0,0,0,0,0,0,0,0,0,0,0,0,255,0,0,0,255, 195. 255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0, 196. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 197. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 198. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 199. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 200. 255,0,255,255,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255, 201. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 202. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 203. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 204. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 205. 0,0,0,0,0,255,0,255,255,255,255,255,255,255,255,255,255,0,0, 206. 0,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0, 207. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 208. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 209. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 210. 0,0,0,0,0,0,0,0,0,255,255,0,255,255,255,255,255,255,255, 211. 255,255,255,255,255,255,255,255,0,0,0,0,0,0,255,255,255,255,255, 212. 255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 213. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 214. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 215. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255, 216. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 217. 0,0,0,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,0, 218. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 219. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 220. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 221. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 222. 255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,255,255,255,255, 223. 255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 224. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 225. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 226. 0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 227. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 228. 255,255,255,255,0,0,0,0,0,255,255,255,255,255,255,255,255,0,0, 229. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 230. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 231. 0,0,0,0,0,0,0,0,0,255,255,0,255,255,255,255,255,255,255, 232. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 233. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0, 234. 255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0, 235. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 236. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,255,255, 237. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 238. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 239. 255,255,255,255,255,0,0,0,0,0,255,255,255,255,255,255,255,255,255, 240. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 241. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 242. 255,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 243. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 244. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0, 245. 0,0,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0, 246. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 247. 0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 248. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 249. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 250. 255,255,255,255,255,255,255,255,255,0,0,0,0,0,255,255,255,255,255, 251. 255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 252. 0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255, 253. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,255, 254. 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, 256. 255,0,0,0,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0, 257. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,255,255, 258. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 259. 255,255,255,0,0,0,255,255,255,255,255,0,255,255,0,0,255,255,255, 260. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 261. 255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,255,255,255, 262. 255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 263. 255,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 264. 255,255,255,255,255,255,255,255,0,255,0,255,255,255,255,255,0,255,255, 265. 0,255,255,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 266. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 267. 255,255,0,0,0,0,0,255,255,255,0,0,0,0,0,0,0,0,0, 268. 0,0,0,0,0,255,255,0,255,255,255,255,255,255,255,255,255,255,255, 269. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,255,255,255,255, 270. 255,255,0,255,255,0,255,255,255,0,255,0,255,0,255,255,255,255,255, 271. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 272. 255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,0,0,0,0, 273. 0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255, 274. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 275. 0,255,255,255,255,255,255,0,255,255,0,255,255,255,0,0,0,255,255, 276. 0,0,0,255,0,255,255,0,255,255,255,255,255,255,255,255,255,255,255, 277. 255,255,255,255,255,255,255,255,255,255,0,255,255,255,255,255,255,0,255, 278. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,0,255,255, 279. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 280. 255,255,255,255,255,255,255,255,255,255,255,0,0,0,255,0,255,255,255, 281. 0,0,255,255,255,0,255,0,255,0,255,0,0,255,255,0,255,255,255, 282. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,0, 283. 0,255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 284. 0,255,0,255,255,255,255,255,255,255,255,255,255,255,0,255,255,255,255, 285. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 286. 255,0,0,0,255,0,255,255,255,255,0,0,255,255,0,0,255,0,255, 287. 255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 288. 255,255,0,0,0,0,255,255,0,255,0,0,0,0,0,0,0,0,0, 289. 0,0,0,0,0,0,255,0,255,255,255,255,255,255,255,255,255,255,0, 290. 0,255,0,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 291. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,255, 292. 0,0,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 293. 255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,0,0,0,0, 294. 0,0,0,0,0,0,0,0,0,0,255,255,0,255,255,255,255,255,255, 295. 255,255,255,255,0,255,255,0,0,0,255,0,0,0,0,0,0,0,0, 296. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 297. 255,255,255,255,255,0,255,255,0,255,0,0,255,255,255,255,255,255,255, 298. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,255,0,255, 299. 255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255, 300. 255,255,255,255,255,255,255,255,255,0,255,255,0,0,0,255,0,255,0, 301. 255,255,255,0,255,0,255,0,0,0,255,255,255,255,255,255,255,255,255, 302. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,255, 303. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 304. 0,255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 305. 0,255,0,255,255,255,255,255,255,255,255,255,255,255,0,255,255,0,0, 306. 0,0,255,255,0,255,255,255,0,255,255,255,0,255,0,255,0,0,0, 307. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 308. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 309. 255,255,255,255,255,255,255,255,0,255,0,0,0,0,0,0,0,0,0, 310. 0,0,0,0,0,0,255,0,255,255,255,255,255,255,255,255,255,255,255, 311. 255,255,255,255,0,0,255,255,255,0,255,255,255,0,0,0,255,255,0, 312. 0,0,255,255,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255, 313. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 314. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,0,0,0,0, 315. 0,0,0,0,0,0,0,0,0,0,0,255,0,255,255,255,255,255,255, 316. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,255,0, 317. 255,255,255,0,0,255,255,255,0,0,0,255,255,255,255,0,255,255,0, 318. 0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 319. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 320. 255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,255, 321. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 322. 255,255,255,255,0,0,0,255,0,255,255,255,0,0,255,255,255,255,255, 323. 255,0,255,255,0,255,255,0,255,0,255,0,255,255,255,255,255,255,255, 324. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 325. 255,255,255,0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0, 326. 0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 327. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,255,0,255, 328. 255,255,255,255,255,255,0,0,255,255,255,255,255,0,0,255,255,255,255, 329. 255,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 330. 255,255,255,255,255,255,255,255,0,255,255,0,0,0,0,0,0,0,0, 331. 0,0,0,0,0,0,255,255,255,255,255,0,255,255,0,255,0,0,255, 332. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 333. 255,255,255,255,255,255,255,255,255,255,0,255,0,0,0,255,255,0,0, 334. 0,255,255,255,255,255,255,0,0,0,0,0,255,255,255,255,255,0,255, 335. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,0,0,0, 336. 0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,0,0,0, 337. 0,255,0,255,0,255,255,0,0,255,255,255,255,255,255,255,255,255,255, 338. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0, 339. 0,0,0,0,255,0,255,255,255,255,255,255,0,255,0,255,255,255,255, 340. 0,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0, 341. 255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255, 342. 255,255,255,255,0,255,255,0,255,255,0,0,255,255,0,255,0,0,0, 343. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 344. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0, 345. 0,255,255,0,0,0,255,0,255,255,255,255,255,255,255,255,255,255,255, 346. 255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0, 347. 0,255,255,0,255,255,255,255,0,0,255,255,0,0,0,255,0,255,255, 348. 255,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 349. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 350. 255,255,255,255,0,0,255,0,255,255,0,255,0,0,0,255,255,255,255, 351. 255,255,255,255,255,255,255,255,0,255,255,0,0,0,0,0,0,0,0, 352. 0,0,0,0,0,0,255,255,0,255,255,255,255,0,255,255,255,0,255, 353. 0,255,0,255,255,255,0,0,255,255,255,255,255,255,255,255,255,255,255, 354. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 355. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,255,0,255, 356. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,0,0,0, 357. 0,0,0,0,0,0,0,0,0,0,0,255,255,0,255,255,255,255,255, 358. 255,255,255,0,0,0,255,0,255,255,0,255,0,255,255,0,255,255,255, 359. 255,255,255,255,0,255,255,0,0,255,0,0,255,255,255,255,255,255,255, 360. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 361. 255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,0, 362. 255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255, 363. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,255,255,0,255, 364. 255,0,255,0,255,255,255,255,255,0,0,0,255,0,0,0,255,0,255, 365. 255,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 366. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 367. 255,255,255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0, 368. 0,0,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 369. 255,255,255,255,255,255,255,0,0,0,255,255,255,255,255,255,0,255,255, 370. 0,0,255,0,0,0,255,255,0,255,0,255,0,255,0,255,255,255,255, 371. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 372. 255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0, 373. 0,0,0,0,0,0,0,255,0,255,255,255,255,255,255,255,255,255,255, 374. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 375. 0,0,0,255,0,0,255,255,0,255,0,255,0,0,0,0,255,0,255, 376. 0,255,0,0,255,0,0,0,255,255,255,255,255,255,255,255,255,255,255, 377. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0, 378. 0,0,0,0,0,0,0,0,0,0,0,255,255,0,255,255,255,255,255, 379. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 380. 255,255,255,255,255,255,255,255,255,0,0,255,255,0,255,255,0,0,0, 381. 0,255,255,0,255,0,0,0,0,0,255,255,255,0,255,255,255,255,255, 382. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 383. 255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0, 384. 255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,255,255,255,255, 385. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 386. 0,0,255,0,255,255,0,255,0,255,0,0,255,0,0,255,255,0,0, 387. 255,255,255,0,255,0,255,0,0,0,255,255,255,255,255,255,255,255,255, 388. 255,255,255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0, 389. 0,0,255,255,255,255,255,255,255,255,255,255,255,0,255,255,0,255,0, 390. 0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 391. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,0,255,255,0, 392. 255,255,255,255,255,255,255,255,0,255,0,255,0,255,255,255,0,255,255, 393. 0,255,255,255,255,255,255,255,255,0,255,0,0,0,0,0,0,0,0, 394. 0,0,0,0,0,0,0,255,0,255,255,255,255,255,255,255,255,255,0, 395. 255,255,0,0,255,255,255,0,0,255,255,0,255,255,255,255,255,255,255, 396. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 397. 255,255,255,255,0,255,0,255,0,255,255,255,255,0,255,0,255,0,0, 398. 255,255,0,255,0,0,255,0,255,255,255,255,255,255,0,255,0,0,0, 399. 0,0,0,0,0,0,0,0,0,0,0,0,255,0,255,255,255,255,255, 400. 255,255,255,255,0,255,255,0,0,255,255,255,0,0,255,0,0,255,255, 401. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 402. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 403. 255,0,255,0,255,255,255,0,0,0,255,255,0,255,255,255,255,255,255, 404. 0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0, 405. 255,255,255,255,255,255,255,255,255,0,255,255,0,255,255,255,255,0,0, 406. 255,0,255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255, 407. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 408. 255,255,255,255,255,255,0,255,0,255,255,255,0,255,255,0,255,0,0, 409. 0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0, 410. 0,0,255,255,0,255,255,255,255,255,255,255,255,255,255,255,0,255,255, 411. 0,255,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,0,255, 412. 255,255,255,0,0,255,255,0,255,255,255,255,255,255,255,255,255,255,255, 413. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,0,0, 414. 255,0,255,0,255,255,255,255,255,0,255,255,0,0,0,0,0,0,0, 415. 0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255, 416. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,255,255, 417. 255,255,255,0,0,255,255,255,0,0,255,0,0,255,255,0,255,255,0, 418. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 419. 255,255,255,255,255,255,255,255,0,0,0,255,255,255,0,255,0,0,0, 420. 0,0,0,0,0,0,0,0,0,0,0,0,255,0,255,255,255,255,255, 421. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 422. 255,0,0,0,255,255,255,255,0,255,255,255,0,255,255,255,0,0,255, 423. 255,0,255,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 424. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 425. 0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,0, 426. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 427. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,255,0,255, 428. 255,255,0,0,0,255,0,255,0,0,255,255,255,255,255,255,255,0,255, 429. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 430. 255,255,255,255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0, 431. 0,0,0,255,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255, 432. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 433. 255,255,255,255,255,255,255,0,255,255,255,0,0,0,0,255,255,255,255, 434. 0,255,255,0,255,255,0,255,0,255,255,255,255,255,255,255,255,255,255, 435. 255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0, 436. 0,0,0,0,0,0,0,255,255,0,0,0,0,0,0,0,0,0,0, 437. 0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 438. 255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,255,0,0,255, 439. 0,255,255,255,255,0,255,255,0,0,0,0,255,0,255,255,255,0,255, 440. 0,255,255,0,255,255,255,255,255,255,255,255,255,255,0,255,255,0,0, 441. 0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,0,0,0, 442. 0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255, 443. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 444. 255,255,255,255,255,255,255,255,255,255,0,255,255,0,255,255,0,255,0, 445. 255,0,0,0,255,0,255,0,0,255,255,255,255,255,255,255,255,255,255, 446. 0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255, 447. 0,0,0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 448. 0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 449. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0, 450. 255,255,0,255,0,255,0,255,0,255,0,0,0,255,255,255,255,255,255, 451. 255,255,255,255,255,0,255,255,0,0,0,0,0,0,0,0,0,0,0, 452. 0,0,0,255,0,0,0,0,255,0,0,0,0,255,255,0,0,0,0, 453. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255, 454. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 455. 255,255,255,255,255,255,255,255,255,0,255,0,255,0,255,0,0,0,255, 456. 255,255,255,255,255,255,255,255,255,255,0,255,0,0,0,0,0,0,0, 457. 0,0,0,0,0,0,0,0,255,0,0,0,0,255,0,0,0,0,255, 458. 255,0,255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0, 459. 0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255, 460. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 461. 255,0,255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0, 462. 0,0,0,0,0,0,0,0,0,0,0,0,0,255,0,0,0,0,255, 463. 255,255,0,0,255,0,255,255,0,0,255,0,0,0,255,255,0,255,255, 464. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255, 465. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 466. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 467. 0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255, 468. 0,0,0,255,255,255,255,255,255,255,255,255,255,0,255,255,0,0,255, 469. 255,255,0,255,255,0,0,255,255,0,0,0,0,0,0,0,0,0,0, 470. 0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255, 471. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 472. 255,255,255,255,255,0,255,255,0,0,0,0,0,0,0,0,0,0,0, 473. 0,0,0,255,255,0,0,0,255,255,0,255,255,255,255,255,255,255,0, 474. 255,255,0,0,255,255,255,0,255,255,255,255,255,0,0,255,0,0,0, 475. 255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 476. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 477. 255,255,255,255,255,255,255,255,255,255,0,255,255,0,0,0,0,0,0, 478. 0,0,0,0,0,0,0,0,255,255,0,0,0,255,255,255,255,255,255, 479. 255,0,255,255,0,255,255,0,255,255,0,255,0,255,255,255,255,255,0, 480. 0,255,0,0,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0, 481. 0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255, 482. 255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,255,0,0, 483. 0,0,0,0,0,0,0,0,0,0,0,0,0,255,0,0,0,0,0, 484. 0,255,255,0,255,255,255,255,255,0,255,255,0,255,0,0,255,0,255, 485. 255,255,255,0,0,255,255,0,0,255,255,0,0,0,0,0,0,0,255, 486. 0,0,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 487. 0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255, 488. 255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255, 489. 0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,255,255,255,255, 490. 0,0,255,0,255,255,255,255,0,0,255,255,0,0,255,255,0,0,0, 491. 0,0,0,255,255,0,0,0,255,0,0,0,255,0,255,0,0,0,0, 492. 0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255, 493. 255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0, 494. 0,0,0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 495. 0,0,0,255,0,0,0,255,0,0,255,255,0,0,0,255,255,0,0, 496. 255,255,255,255,0,0,0,0,255,255,0,0,0,255,255,255,255,255,0, 497. 255,255,0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 498. 0,0,0,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,0, 499. 0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,0,0,0, 500. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,255, 501. 255,255,255,255,255,255,255,0,0,0,0,0,0,255,255,255,255,0,255, 502. 255,255,255,0,255,255,0,255,255,255,0,255,0,0,255,255,0,0,0, 503. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0, 504. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255, 505. 255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0, 506. 0,0,0,0,0,255,255,255,255,255,255,255,255,255,0,0,0,0,255, 507. 0,255,255,255,255,255,255,255,0,255,255,255,255,255,0,255,255,0,0, 508. 255,255,0,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0, 509. 0,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 510. 0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,0,0, 511. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255, 512. 0,0,0,0,255,255,255,255,255,255,255,255,255,0,255,255,255,255,255, 513. 0,255,255,255,255,255,0,0,255,255,255,255,0,0,255,255,0,0,255, 514. 255,0,0,0,0,0,0,255,0,0,0,0,0,0,0,0,0,0,0, 515. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255, 516. 255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0, 517. 0,0,0,0,0,0,0,0,0,255,255,255,255,0,255,255,255,0,0, 518. 255,255,0,255,255,0,255,255,255,255,255,0,255,255,255,255,255,0,255, 519. 255,255,0,255,255,255,255,0,0,0,0,0,255,0,0,0,0,0,0, 520. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 521. 0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255, 522. 255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 523. 255,255,255,0,0,255,255,255,255,255,0,255,255,255,255,0,0,255,255, 524. 0,255,255,255,255,255,0,255,255,0,0,255,255,0,0,0,255,255,0, 525. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 526. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 527. 0,0,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0, 528. 0,0,0,0,0,0,0,0,0,0,255,255,255,255,0,0,255,255,255, 529. 255,0,0,255,255,0,255,255,255,255,255,0,255,255,0,255,255,255,0, 530. 0,0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 531. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 532. 0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255, 533. 255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 534. 0,0,255,0,0,255,255,0,255,255,0,255,255,255,255,255,0,255,255, 535. 0,255,255,255,0,0,0,255,255,0,0,0,0,0,0,0,0,0,0, 536. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 537. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 538. 0,0,0,0,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0, 539. 0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,255,255,0, 540. 255,255,0,255,255,0,0,0,0,0,0,0,255,255,0,0,0,0,0, 541. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 542. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 543. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255, 544. 255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 545. 0,0,0,0,0,255,255,0,255,255,255,255,255,0,0,0,0,255,0, 546. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 547. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 548. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 549. 0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,0,0,0, 550. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,0,0, 551. 0,0,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 552. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 553. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 554. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255, 555. 255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0, 556. 0,0,0,0,0,0,0,255,255,0,0,0,0,0,0,0,0,0,0, 557. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 558. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 559. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 560. 0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,0,0, 561. 0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,0,0,0, 562. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 563. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 564. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 565. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255, 566. 255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,255,255, 567. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 568. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 569. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 570. 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0, 571. 0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255, 572. 255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0 573. }; |

Mean time calculation of Image processing

|  |  |  |
| --- | --- | --- |
| Run time | Original Time in Nano second | MPI time in nano second |
| 1 | 73251 | 709502 |
| 2 | 61698 | 3905668 |
| 3 | 62160 | 5592521 |
| 4 | 61606 | 831325 |
| 5 | 66720 | 2225413 |
| 6 | 299407 | 944255 |
| 7 | 61957 | 8179203 |
| 8 | 62064 | 986019 |
| 9 | 62564 | 9734165 |
| 10 | 91841 | 833920 |
| Mean | 90326.8 NS = 9.03268e-5 | 3394199.1 NS = 0.0033941991 S |

**Result analysis of Image processing**

The average run time of the original image processing time is 9.03268e-5 seconds and the run time of the MPI program is 0.0033941991 seconds. Here, after analyzing the result of the original program and the MPI program, the MPI is relatively faster than the original program reason behind that is In MPI, the master assigns work to the other processors so that they divide the time to the other workers so that the program is faster than the other .Here in the Edge Detection program at the time that 4 MPI instances are used to make it faster.

## 3.3 Linear Regression

|  |
| --- |
| 1. #include <stdio.h> 2. #include <math.h> 3. #include <time.h> 4. #include <mpi.h> 5. #include <malloc.h> 6. #include <sys/stat.h> 7. #include <stdlib.h> 8. #include <stdio.h> 9. #include <math.h> 11. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 12. \* This program takes an initial estimate of m and c and finds the associated 13. \* rms error. It is then as a base to generate and evaluate 8 new estimates, 14. \* which are steps in different directions in m-c space. The best estimate is 15. \* then used as the base for another iteration of "generate and evaluate". This 16. \* continues until none of the new estimates are better than the base. This is 17. \* a gradient search for a minimum in mc-space. 18. \* 19. \* To compile: 20. \*     mpicc -o Lr\_Mpi Lr\_Mpi.c -lm 21. \* 22. \* To run: 23. \*   mpirun -n 9 -quiet ./Lr\_Mpi 24. \*Name:1928720\_Adhikar\_chaudhary 25. \*@University of wolverhamption 26. \* 27. \* Dr Kevan Buckley, University of Wolverhampton, 2018 28. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/ 30. **typedef** **struct** point\_t 31. { 32. **double** x; 33. **double** y; 34. } point\_t; 36. **int** n\_data = 1000; 37. point\_t data[]; 39. **double** residual\_error (**double** x, **double** y, **double** m, **double** c) 40. { 41. **double** e = (m \* x) + c - y; 42. **return** e \* e; 43. } 45. **double** rms\_error (**double** m, **double** c) 46. { 47. **int** i; 48. **double** mean; 49. **double** error\_sum = 0; 51. **for** (i = 0; i < n\_data; i++) 52. { 53. error\_sum += residual\_error (data[i].x, data[i].y, m, c); 54. } 56. mean = error\_sum / n\_data; 58. **return** sqrt (mean); 59. } 60. **int** time\_difference(**struct** timespec \*start, **struct** timespec \*finish, 61. **long** **long** **int** \*difference) { 62. **long** **long** **int** ds =  finish->tv\_sec - start->tv\_sec; 63. **long** **long** **int** dn =  finish->tv\_nsec - start->tv\_nsec; 65. **if**(dn < 0 ) { 66. ds--; 67. dn += 1000000000; 68. } 69. \*difference = ds \* 1000000000 + dn; 70. **return** !(\*difference > 0); 71. } 72. **int** main () {  75. **struct** timespec start, finish; 76. **long** **long** **int** time\_elapsed; 77. clock\_gettime(CLOCK\_MONOTONIC, &start);  80. **int** rank, size; 81. **int** i; 82. **double** bm = 1.3; 83. **double** bc = 10; 84. **double** be; 85. **double** dm[8]; 86. **double** dc[8]; 87. **double** e[8]; 88. **double** step = 0.01; 89. **double** best\_error = 999999999; 90. **int** best\_error\_i; 91. **int** minimum\_found = 0; 92. **double** pError = 0; 93. **double** baseMC[2]; 95. **double** om[] = { 0, 1, 1, 1, 0, -1, -1, -1 }; 96. **double** oc[] = { 1, 1, 0, -1, -1, -1, 0, 1 };  99. MPI\_Init (NULL, NULL); 100. MPI\_Comm\_size (MPI\_COMM\_WORLD, &size); 101. MPI\_Comm\_rank (MPI\_COMM\_WORLD, &rank); 103. be = rms\_error (bm, bc); 105. **if** (size != 9) 106. { 107. **if** (rank == 0) 108. { 109. printf 110. ("Nine processes.\n"); 111. **return** 0; 112. } 113. } 115. **while** (!minimum\_found) 116. { 118. **if** (rank != 0) 119. { 120. i = rank - 1; 121. dm[i] = bm + (om[i] \* step); 122. dc[i] = bc + (oc[i] \* step); 123. pError = rms\_error (dm[i], dc[i]); 125. MPI\_Send (&pError, 1, MPI\_DOUBLE, 0, 0, MPI\_COMM\_WORLD); 126. MPI\_Send (&dm[i], 1, MPI\_DOUBLE, 0, 0, MPI\_COMM\_WORLD); 127. MPI\_Send (&dc[i], 1, MPI\_DOUBLE, 0, 0, MPI\_COMM\_WORLD);  130. MPI\_Recv (&bm, 1, MPI\_DOUBLE, 0, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE); 131. MPI\_Recv (&bc, 1, MPI\_DOUBLE, 0, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE); 132. MPI\_Recv (&minimum\_found, 1, MPI\_INT, 0, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE); 133. } 134. **else** 135. { 136. **for** (i = 1; i < size; i++) 137. { 138. MPI\_Recv (&pError, 1, MPI\_DOUBLE, i, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE); 139. MPI\_Recv (&dm[i-1], 1, MPI\_DOUBLE, i, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE); 140. MPI\_Recv (&dc[i-1], 1, MPI\_DOUBLE, i, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE); 141. **if** (pError < best\_error) 142. { 143. best\_error = pError; 144. best\_error\_i = i - 1; 146. } 147. } 148. **if** (best\_error < be) 149. { 150. be = best\_error; 151. bm = dm[best\_error\_i]; 152. bc = dc[best\_error\_i]; 153. } 154. **else** 155. { 156. minimum\_found = 1; 157. } 159. **for** (i = 1; i < size; i++) 160. { 161. MPI\_Send (&bm, 1, MPI\_DOUBLE, i, 0, MPI\_COMM\_WORLD); 162. MPI\_Send (&bc, 1, MPI\_DOUBLE, i, 0, MPI\_COMM\_WORLD); 163. MPI\_Send (&minimum\_found, 1, MPI\_INT, i, 0, MPI\_COMM\_WORLD); 164. } 165. } 166. } 168. **if**(rank==0) { 169. printf ("minimum m,c is %lf,%lf with error %lf\n", bm, bc, be); 170. clock\_gettime(CLOCK\_MONOTONIC, &finish); 171. time\_difference(&start, &finish, &time\_elapsed); 172. printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed, 173. (time\_elapsed/1.0e9)); 174. } 176. MPI\_Finalize(); 177. **return** 0; 179. } 180. point\_t data[] = { 181. {87.72,160.36},{79.77,154.56},{74.88,144.94},{67.05,157.02}, 182. {73.75,121.36},{69.58,132.51},{67.21,143.57},{72.13,141.31}, 183. {87.64,144.77},{65.42,146.46},{83.09,144.82},{73.30,137.10}, 184. {76.75,151.67},{69.17,144.50},{87.61,171.62},{83.56,164.97}, 185. {75.57,137.55},{73.80,156.25},{75.75,130.90},{97.13,193.14}, 186. { 6.54,74.00},{54.31,100.05},{57.36,127.37},{93.54,172.18}, 187. {98.26,190.44},{87.27,149.69},{66.44,143.27},{98.24,171.56}, 188. {27.74,98.27},{10.51,57.95},{88.17,179.14},{22.66,63.64}, 189. {30.46,64.81},{65.24,135.15},{ 1.71,39.30},{91.29,171.76}, 190. {77.43,153.66},{68.29,132.07},{33.43,82.08},{60.40,119.17}, 191. {15.86,64.98},{61.96,126.18},{11.05,58.97},{23.99,67.74}, 192. {21.13,65.78},{34.79,97.87},{22.32,74.86},{78.29,136.48}, 193. {25.38,84.45},{94.49,169.91},{35.62,99.82},{58.20,127.53}, 194. {98.37,182.22},{93.79,168.74},{26.98,91.60},{30.55,103.77}, 195. { 5.86,44.79},{78.96,159.17},{93.19,158.82},{24.73,51.92}, 196. {94.91,186.12},{61.84,135.07},{ 2.60,37.48},{95.93,189.32}, 197. {95.39,157.13},{20.24,69.22},{29.93,71.36},{68.29,148.83}, 198. {36.68,92.16},{93.29,180.89},{30.53,75.37},{48.10,120.04}, 199. {83.17,153.60},{ 8.22,50.63},{ 8.76,50.80},{97.71,171.61}, 200. {13.28,46.71},{ 1.07,43.74},{11.16,71.21},{20.98,78.43}, 201. {21.89,50.58},{11.51,55.46},{ 3.22,43.94},{41.33,86.40}, 202. { 9.72,66.88},{10.91,51.56},{40.75,100.96},{81.92,161.87}, 203. {54.44,112.70},{11.35,62.04},{69.49,123.86},{50.51,96.47}, 204. {74.84,150.85},{77.35,150.52},{81.72,163.59},{11.65,37.53}, 205. {13.48,52.63},{79.02,166.24},{84.93,171.49},{59.13,117.54}, 206. {70.14,129.94},{58.90,118.01},{28.64,107.03},{43.71,112.86}, 207. {58.86,114.54},{78.67,160.78},{94.74,171.99},{70.31,122.40}, 208. {16.20,74.80},{58.59,110.98},{70.81,130.09},{ 7.86,46.69}, 209. {91.46,153.64},{34.05,87.81},{24.06,68.39},{66.23,134.24}, 210. {58.69,121.12},{ 0.18,38.19},{20.48,63.34},{63.79,125.30}, 211. {27.38,80.17},{39.12,100.73},{85.45,151.11},{48.89,92.37}, 212. {81.54,155.19},{20.52,55.22},{16.33,56.39},{82.69,158.63}, 213. {78.03,160.98},{ 3.36,62.70},{ 9.16,36.34},{ 5.56,50.25}, 214. {96.94,185.14},{46.54,107.56},{13.32,54.76},{21.63,58.16}, 215. {35.36,86.17},{92.63,175.36},{35.77,93.56},{24.99,73.63}, 216. {78.27,153.33},{44.20,89.01},{ 8.41,28.53},{22.39,65.89}, 217. { 4.01,63.33},{61.57,131.79},{39.85,95.92},{76.59,132.38}, 218. {49.60,117.48},{54.00,114.01},{92.20,160.82},{98.73,180.85}, 219. {56.18,108.40},{75.10,144.54},{36.91,105.85},{36.52,75.99}, 220. {47.65,79.93},{72.52,151.63},{50.78,112.17},{85.76,151.27}, 221. {14.41,51.07},{48.66,109.44},{50.39,104.20},{95.63,174.36}, 222. {69.74,140.10},{25.53,87.75},{67.15,141.26},{55.44,129.77}, 223. {20.44,65.95},{98.96,181.37},{47.04,111.25},{58.45,117.28}, 224. {57.79,123.58},{83.84,171.31},{61.97,131.09},{37.66,109.35}, 225. {65.32,124.56},{32.31,92.17},{64.08,140.31},{53.53,119.28}, 226. {46.02,102.80},{ 8.19,46.99},{75.24,134.02},{11.54,67.14}, 227. {37.35,94.89},{17.90,73.66},{68.74,127.84},{35.61,74.80}, 228. {12.89,61.03},{51.57,127.06},{69.44,137.16},{55.00,87.07}, 229. { 9.46,66.57},{ 7.32,50.65},{18.69,60.81},{91.80,157.00}, 230. {23.13,80.31},{79.77,160.31},{25.92,67.57},{88.84,157.95}, 231. {59.62,122.91},{20.02,74.74},{66.18,130.20},{22.33,67.57}, 232. {34.98,72.13},{27.88,80.57},{74.11,152.93},{63.45,123.91}, 233. {34.17,76.27},{59.99,137.07},{26.78,70.38},{30.99,79.08}, 234. { 3.26,42.81},{51.85,97.62},{38.00,89.05},{69.57,128.69}, 235. {55.77,102.44},{38.64,95.04},{31.94,80.16},{33.57,84.21}, 236. {49.34,115.16},{60.37,126.82},{48.79,79.20},{53.62,101.98}, 237. {44.49,120.58},{40.72,92.32},{62.30,134.92},{86.71,167.45}, 238. {28.92,90.49},{ 3.64,50.38},{ 2.98,51.03},{ 7.72,50.72}, 239. {36.01,88.98},{12.80,53.33},{18.18,59.93},{ 8.74,39.72}, 240. {93.05,157.51},{78.51,157.36},{ 9.32,51.91},{28.43,83.25}, 241. { 5.63,57.81},{ 5.75,49.78},{33.39,89.81},{98.06,183.34}, 242. {61.33,124.19},{90.28,167.83},{97.93,180.90},{24.74,74.19}, 243. {28.42,90.02},{67.13,126.96},{29.53,65.54},{28.34,82.58}, 244. { 5.75,35.03},{36.57,89.60},{18.70,68.23},{85.61,147.33}, 245. {90.19,148.59},{29.44,82.07},{95.34,164.13},{25.79,76.89}, 246. {18.18,55.48},{47.36,99.63},{ 9.34,53.88},{95.16,174.51}, 247. {79.94,148.24},{ 1.82,30.86},{75.83,138.77},{15.10,60.29}, 248. {34.45,91.55},{ 0.41,39.02},{12.90,64.51},{75.66,161.10}, 249. {40.13,85.95},{57.37,109.43},{79.64,155.61},{25.54,76.82}, 250. {33.13,80.14},{31.55,75.12},{ 0.93,51.83},{36.40,97.82}, 251. {55.60,112.28},{ 1.26,47.44},{73.54,125.72},{49.81,102.51}, 252. {97.31,170.44},{89.64,169.02},{91.89,169.61},{45.89,107.66}, 253. {99.90,168.98},{39.27,106.49},{63.61,118.52},{60.86,115.28}, 254. { 3.11,48.70},{61.64,119.97},{28.18,81.53},{ 5.20,43.98}, 255. {51.76,117.25},{57.63,127.76},{78.84,153.80},{10.54,48.23}, 256. {52.51,115.61},{ 4.69,28.30},{93.93,164.59},{69.70,141.08}, 257. {57.13,118.40},{23.45,86.44},{44.59,132.03},{31.91,98.03}, 258. {44.44,103.85},{ 9.46,53.83},{92.57,190.48},{36.29,95.15}, 259. {32.06,92.48},{86.16,138.27},{49.86,112.94},{96.14,181.82}, 260. {16.05,59.03},{52.13,102.89},{51.27,109.44},{67.94,117.86}, 261. {86.77,158.44},{74.20,143.83},{39.13,93.09},{66.00,137.66}, 262. {22.48,55.93},{54.90,116.64},{51.71,95.82},{36.07,85.96}, 263. { 5.38,59.94},{84.49,160.75},{28.93,82.85},{89.92,183.08}, 264. { 1.83,61.19},{26.71,73.35},{49.96,93.82},{13.56,68.73}, 265. {26.93,70.29},{85.19,165.22},{74.31,148.26},{44.90,107.17}, 266. {81.60,159.66},{86.68,150.41},{ 8.77,66.43},{75.18,159.06}, 267. { 8.86,45.78},{66.61,131.48},{ 5.80,40.46},{84.73,169.37}, 268. {35.34,85.87},{97.62,161.97},{15.22,73.73},{77.52,152.32}, 269. { 2.96,45.12},{60.66,127.28},{66.50,131.20},{72.85,141.01}, 270. {65.90,130.75},{43.44,101.16},{11.06,52.18},{14.77,62.27}, 271. {77.52,159.43},{47.99,126.74},{63.43,110.36},{50.26,113.37}, 272. {95.43,198.57},{24.53,57.71},{12.87,54.28},{63.79,135.32}, 273. {56.58,110.85},{59.10,121.23},{99.99,194.48},{34.56,83.22}, 274. {98.83,173.25},{ 7.73,70.43},{34.44,103.18},{96.37,169.25}, 275. {50.19,118.06},{84.66,175.92},{79.01,142.91},{99.12,174.11}, 276. {61.97,120.41},{81.93,146.96},{36.18,71.82},{ 3.47,51.04}, 277. { 0.07,52.79},{82.02,158.42},{37.02,94.32},{26.83,77.25}, 278. {87.90,166.44},{22.65,76.05},{ 8.80,53.89},{53.12,109.05}, 279. {64.69,147.19},{55.56,107.54},{98.42,175.01},{ 4.99,30.66}, 280. {63.23,122.97},{56.44,121.49},{ 8.58,69.79},{79.38,149.69}, 281. {55.64,122.77},{84.76,142.77},{29.21,85.68},{18.11,70.14}, 282. {33.15,85.38},{11.30,44.68},{83.67,163.51},{23.43,101.91}, 283. {29.59,81.85},{19.90,75.18},{26.05,89.95},{61.05,129.50}, 284. { 6.27,68.93},{96.95,191.69},{82.78,145.43},{73.84,136.65}, 285. {60.44,140.08},{12.67,65.86},{ 2.20,36.68},{ 6.16,54.50}, 286. {35.28,95.48},{83.01,152.33},{64.33,121.41},{91.72,162.34}, 287. {72.62,130.77},{55.31,121.78},{83.52,164.02},{ 1.45,24.37}, 288. {84.96,164.70},{ 6.85,59.62},{89.95,158.91},{57.41,133.29}, 289. { 6.13,58.72},{78.33,125.38},{65.15,121.67},{19.10,81.33}, 290. {17.43,58.99},{60.92,142.74},{69.37,142.70},{ 7.71,53.52}, 291. {38.81,101.59},{18.31,65.35},{41.05,90.89},{29.84,96.28}, 292. {27.65,88.04},{11.74,30.21},{72.14,150.80},{59.92,123.34}, 293. {20.59,51.65},{73.54,147.09},{25.04,52.81},{21.44,78.92}, 294. { 8.30,68.18},{38.27,103.50},{76.73,135.46},{13.41,42.84}, 295. { 9.77,50.17},{31.79,84.64},{11.63,47.87},{81.10,154.34}, 296. {32.88,86.39},{83.66,156.16},{42.84,101.97},{92.23,181.69}, 297. {56.62,128.67},{21.57,72.57},{28.42,76.81},{78.49,151.43}, 298. {34.76,87.12},{95.65,153.86},{48.99,114.03},{22.97,82.24}, 299. {96.82,167.98},{55.42,93.17},{59.22,121.38},{41.66,101.95}, 300. {77.91,166.28},{12.14,54.69},{91.24,171.96},{46.22,106.58}, 301. {98.54,161.56},{46.98,104.41},{60.09,131.79},{67.83,119.99}, 302. {41.09,85.10},{70.10,135.73},{36.99,91.78},{10.72,53.99}, 303. {50.06,91.80},{18.39,60.80},{26.53,85.06},{76.65,154.26}, 304. { 0.02,34.66},{37.56,104.13},{91.48,177.82},{ 0.31,41.81}, 305. {93.20,166.21},{ 6.94,43.44},{85.38,159.15},{90.74,139.02}, 306. {83.80,157.57},{39.34,92.96},{95.49,170.28},{77.55,145.57}, 307. {33.05,88.61},{ 4.28,43.15},{95.89,183.02},{36.18,97.42}, 308. {94.88,160.54},{ 5.18,55.47},{24.04,66.69},{50.93,118.02}, 309. {11.82,58.02},{ 4.55,62.84},{71.87,136.43},{ 1.64,46.82}, 310. {86.64,157.08},{76.78,135.11},{17.94,59.70},{41.81,96.43}, 311. {89.21,170.11},{75.89,139.66},{90.73,160.27},{45.39,101.75}, 312. {61.07,135.71},{ 7.53,48.91},{26.32,74.50},{89.44,164.11}, 313. {79.85,162.30},{55.91,104.02},{89.47,172.39},{88.09,154.40}, 314. {87.63,148.99},{24.43,74.77},{49.28,103.08},{86.49,165.82}, 315. {93.79,148.29},{93.25,170.91},{59.55,126.98},{ 9.24,64.46}, 316. {73.82,134.35},{76.07,152.94},{77.44,148.98},{87.01,161.01}, 317. {72.52,154.33},{21.77,97.29},{47.74,111.47},{17.34,68.45}, 318. {39.75,119.04},{84.78,160.15},{62.57,121.76},{70.20,146.15}, 319. { 7.08,50.89},{60.33,97.44},{29.90,89.84},{41.82,95.48}, 320. {38.27,101.32},{ 9.60,53.87},{84.69,166.21},{97.64,177.02}, 321. {73.96,145.64},{11.68,73.29},{31.64,75.20},{44.12,119.54}, 322. {29.91,99.48},{62.85,117.28},{65.55,123.70},{78.66,161.94}, 323. {71.06,158.16},{71.17,147.50},{12.49,63.11},{62.47,146.21}, 324. { 5.52,64.77},{19.39,81.84},{90.73,177.46},{46.94,101.25}, 325. {35.30,92.84},{25.31,80.86},{29.29,95.30},{79.68,160.63}, 326. {65.64,143.96},{81.97,173.79},{58.68,123.92},{37.35,94.26}, 327. {81.31,146.56},{10.82,34.36},{72.19,152.74},{96.81,157.39}, 328. {37.82,84.01},{26.02,85.45},{49.68,99.80},{63.61,134.18}, 329. {78.45,135.62},{28.06,99.20},{29.49,76.17},{38.73,90.19}, 330. {66.67,128.82},{ 4.14,43.63},{75.01,155.68},{12.38,39.83}, 331. {64.06,126.87},{77.44,154.01},{30.67,89.38},{30.38,85.81}, 332. {98.17,180.45},{72.62,131.67},{18.80,74.37},{56.99,112.44}, 333. {80.45,152.36},{43.87,96.62},{10.95,79.63},{ 8.13,48.84}, 334. {96.47,180.58},{57.99,139.71},{29.81,82.98},{ 7.69,59.04}, 335. {60.75,113.11},{61.26,127.28},{29.91,74.20},{72.81,130.29}, 336. {97.22,186.22},{16.10,67.14},{45.22,82.91},{59.90,136.15}, 337. {50.86,99.99},{40.09,89.91},{38.69,87.12},{38.22,86.34}, 338. {82.85,160.54},{44.59,114.27},{ 6.39,49.22},{53.02,118.97}, 339. {67.10,132.43},{87.17,167.48},{61.46,109.49},{79.66,163.17}, 340. {40.28,88.74},{81.76,164.45},{10.26,58.64},{14.58,81.13}, 341. {85.30,184.26},{64.06,132.71},{ 5.55,56.52},{96.97,187.38}, 342. {92.22,174.95},{42.45,100.51},{30.79,81.24},{ 4.25,61.71}, 343. {47.15,104.16},{35.87,86.39},{81.62,152.64},{42.46,95.25}, 344. {66.69,137.47},{33.21,84.65},{23.42,84.12},{99.30,187.76}, 345. {19.15,77.26},{17.74,70.35},{87.90,170.12},{47.01,118.00}, 346. {78.63,155.19},{92.38,163.60},{72.75,153.70},{79.92,138.69}, 347. {21.94,78.76},{55.51,120.91},{27.08,57.31},{12.83,45.59}, 348. {48.22,103.52},{35.64,87.26},{59.90,119.91},{50.05,110.55}, 349. { 0.23,41.68},{66.03,129.51},{42.67,95.15},{37.78,103.08}, 350. { 3.06,43.68},{53.80,102.89},{ 9.78,51.90},{94.94,185.83}, 351. {31.69,105.92},{70.50,123.84},{ 5.52,51.03},{ 0.93,47.63}, 352. {68.12,146.17},{ 6.86,51.21},{ 4.60,42.38},{72.98,138.03}, 353. {58.59,125.26},{40.21,88.92},{12.51,41.25},{31.12,65.03}, 354. {75.68,143.15},{74.02,141.52},{ 5.61,50.98},{82.39,162.02}, 355. {28.07,65.38},{71.22,145.28},{44.22,99.25},{72.03,123.62}, 356. {45.88,95.49},{76.37,136.85},{29.19,81.03},{63.46,142.45}, 357. {49.44,100.25},{81.71,132.07},{83.34,150.05},{38.88,93.91}, 358. {86.01,172.44},{51.32,110.43},{86.82,154.64},{70.02,140.53}, 359. {26.43,72.25},{34.48,91.08},{30.41,80.09},{24.77,80.43}, 360. { 8.14,53.39},{18.88,70.38},{26.90,73.96},{94.43,173.51}, 361. {24.45,62.18},{56.07,111.25},{66.96,136.85},{93.78,188.93}, 362. {75.18,144.91},{18.22,43.66},{97.70,170.91},{34.25,95.34}, 363. {12.16,53.60},{88.48,164.67},{81.58,176.86},{81.96,151.45}, 364. {50.13,109.26},{44.20,90.81},{52.84,121.91},{17.30,76.64}, 365. {53.60,120.26},{32.01,84.79},{72.56,149.71},{19.15,55.05}, 366. {78.26,164.32},{ 9.84,56.96},{ 2.48,50.11},{50.84,134.33}, 367. {90.65,164.89},{35.58,81.09},{72.54,151.27},{54.39,119.73}, 368. {44.15,105.72},{74.88,145.20},{86.66,158.64},{17.79,67.78}, 369. {54.84,115.87},{99.10,173.71},{93.02,174.29},{23.52,83.88}, 370. {19.56,68.81},{24.03,79.83},{11.73,35.38},{ 3.82,37.84}, 371. {61.92,130.48},{77.02,139.38},{91.29,161.65},{98.09,162.25}, 372. { 0.49,36.07},{75.44,138.86},{ 4.32,59.64},{79.99,143.62}, 373. {13.43,47.42},{44.44,110.46},{25.03,71.30},{71.95,147.86}, 374. {78.51,152.59},{ 3.43,34.40},{55.28,115.97},{88.77,165.45}, 375. {15.43,82.65},{99.09,179.00},{79.77,143.93},{52.73,116.02}, 376. {52.40,109.06},{37.24,83.30},{31.90,80.93},{68.13,127.42}, 377. {70.63,127.66},{55.84,132.35},{39.95,99.29},{ 6.84,40.38}, 378. {66.47,117.89},{20.53,83.00},{82.22,147.22},{23.74,73.03}, 379. {77.83,159.15},{11.29,64.73},{49.15,104.01},{52.54,105.95}, 380. {93.36,160.71},{51.35,106.68},{28.56,83.86},{78.27,147.91}, 381. { 0.25,50.56},{59.93,120.88},{ 8.58,49.04},{74.24,134.96}, 382. {51.22,98.10},{24.91,74.31},{87.07,160.74},{52.25,105.22}, 383. {91.43,152.78},{ 8.10,59.46},{94.97,178.64},{88.81,178.24}, 384. {88.45,150.89},{21.60,76.09},{70.62,122.85},{99.65,168.36}, 385. {73.32,142.58},{13.18,71.21},{37.26,88.09},{79.15,142.60}, 386. {20.05,91.09},{33.64,87.49},{21.84,64.04},{49.12,116.82}, 387. {52.57,125.37},{42.43,93.45},{22.54,94.46},{82.51,165.14}, 388. {77.64,132.17},{32.25,83.61},{10.77,55.92},{71.34,133.82}, 389. {60.85,127.38},{22.68,79.72},{30.62,77.71},{81.90,161.50}, 390. {10.22,47.19},{26.58,57.16},{43.66,113.00},{90.69,145.82}, 391. {12.64,58.91},{85.90,154.22},{18.03,53.36},{84.49,144.57}, 392. {87.51,169.54},{92.50,170.96},{51.99,123.08},{45.16,108.57}, 393. {71.40,137.44},{58.36,121.84},{76.06,143.46},{42.17,104.52}, 394. { 2.57,50.10},{11.44,41.77},{71.09,143.92},{88.92,151.09}, 395. {92.79,177.42},{90.72,157.64},{66.11,141.52},{ 2.33,38.80}, 396. {76.26,158.72},{76.52,150.07},{70.31,132.13},{52.77,119.85}, 397. {99.59,176.94},{ 8.16,56.11},{99.29,190.79},{25.00,77.22}, 398. {13.45,63.42},{17.35,70.05},{ 4.16,35.31},{86.57,152.50}, 399. {88.57,168.14},{67.96,123.67},{72.36,142.41},{10.95,73.06}, 400. {78.45,163.31},{71.69,139.46},{82.78,157.91},{80.14,161.51}, 401. {60.33,133.76},{ 9.44,46.00},{68.21,163.48},{30.78,88.27}, 402. {38.74,105.24},{19.52,62.94},{49.03,105.82},{76.01,138.95}, 403. {71.08,165.10},{49.97,108.11},{75.15,145.38},{ 5.20,62.33}, 404. {97.13,188.35},{87.18,176.82},{42.70,96.24},{62.98,126.92}, 405. {96.09,175.08},{90.77,190.99},{71.68,124.23},{15.67,61.12}, 406. {95.37,178.30},{40.64,83.70},{22.64,71.16},{30.22,105.64}, 407. {18.96,77.17},{56.47,98.36},{36.53,84.99},{13.11,73.50}, 408. {32.04,80.23},{72.49,135.67},{54.33,126.59},{13.54,51.56}, 409. {14.77,57.56},{24.09,90.04},{32.43,86.80},{ 3.82,43.03}, 410. {81.10,163.58},{45.39,96.22},{57.29,115.98},{76.10,151.06}, 411. { 7.74,56.38},{48.95,108.35},{40.07,101.13},{81.91,144.88}, 412. {64.47,124.53},{70.83,129.18},{ 7.05,44.06},{36.46,86.68}, 413. {32.53,89.07},{32.88,77.92},{ 6.62,29.48},{28.87,76.01}, 414. {37.36,90.05},{72.25,136.07},{81.47,173.47},{ 4.20,47.41}, 415. {98.64,166.84},{46.61,109.47},{45.38,88.61},{95.41,169.40}, 416. {66.63,122.42},{98.96,176.41},{77.60,166.70},{39.53,93.96}, 417. {73.29,138.23},{87.99,159.87},{34.35,91.01},{33.30,78.44}, 418. {29.29,78.05},{89.99,153.84},{ 3.90,31.79},{ 2.74,28.45}, 419. {74.07,144.41},{59.60,135.80},{83.19,154.17},{33.14,71.48}, 420. {71.18,127.25},{59.10,126.89},{14.88,60.57},{46.36,122.43}, 421. {97.68,166.02},{47.91,110.00},{94.43,185.03},{25.13,73.98}, 422. {30.66,83.04},{47.36,100.33},{20.03,57.00},{38.53,77.34}, 423. {53.29,122.85},{77.72,146.52},{23.42,84.64},{96.85,170.99}, 424. { 8.49,68.27},{71.67,127.11},{84.22,158.75},{35.25,87.63}, 425. {74.00,140.25},{32.42,91.48},{88.91,156.95},{88.11,163.18}, 426. {60.49,132.55},{63.94,149.75},{95.21,172.49},{ 2.14,37.70}, 427. { 2.33,36.29},{24.38,83.62},{87.11,162.19},{37.16,85.65}, 428. {81.57,152.86},{49.26,116.82},{54.72,108.56},{65.82,132.02}, 429. {10.93,47.63},{71.92,111.28},{12.67,36.23},{67.35,101.56}, 430. {86.25,169.18},{97.89,194.43},{40.63,106.58},{73.87,135.71} 431. }; |

**Mean Time analysis**

|  |  |  |
| --- | --- | --- |
| **Run Time** | **Original Time in second** | **MPI in second** |
| 1 |  | 1.07123663 |
| 2 |  | 0.673524714 |
| 3 |  | 0.955319488 |
| 4 |  | 1.009382439 |
| 5 |  | 0.859765587 |
| 6 |  | 0.797444912 |
| 7 |  | 1.991035077 |
| 8 |  | 1.014209037 |
| 9 |  | 0.863873347 |
| 10 |  | 1.043207169 |
| Mean |  | 1.02789984 |