 МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

УНИВЕРСИТЕТ

Институт №8

«Компьютерные науки и прикладная математика» Кафедра №806

«Виртуальная/дополнительная реальность

и искусственный интеллект»

Отчет по лабораторной работе № 1,

по учебной дисциплине

«Параллельные и распределенные вычисления»

Выполнил:

Студент 1-го курса

Гр. М80-114M-22

Гордеев Н. М.

.

(подпись, дата) .

Принял:

Семенов С. А.

.

(подпись, дата) .

Москва 2022

Оглавление

[1. Постановка задачи 2](#_Toc122001949)

[2. Метод решения 2](#_Toc122001950)

[3. Результат работы программы 2](#_Toc122001951)

[4. Вывод: 3](#_Toc122001952)

[5. Код программы 4](#_Toc122001953)

# 1. Постановка задачи

Вариант № 5. Задать массив чисел от 1 до 512 и инвертировать его

# 2. Метод решения

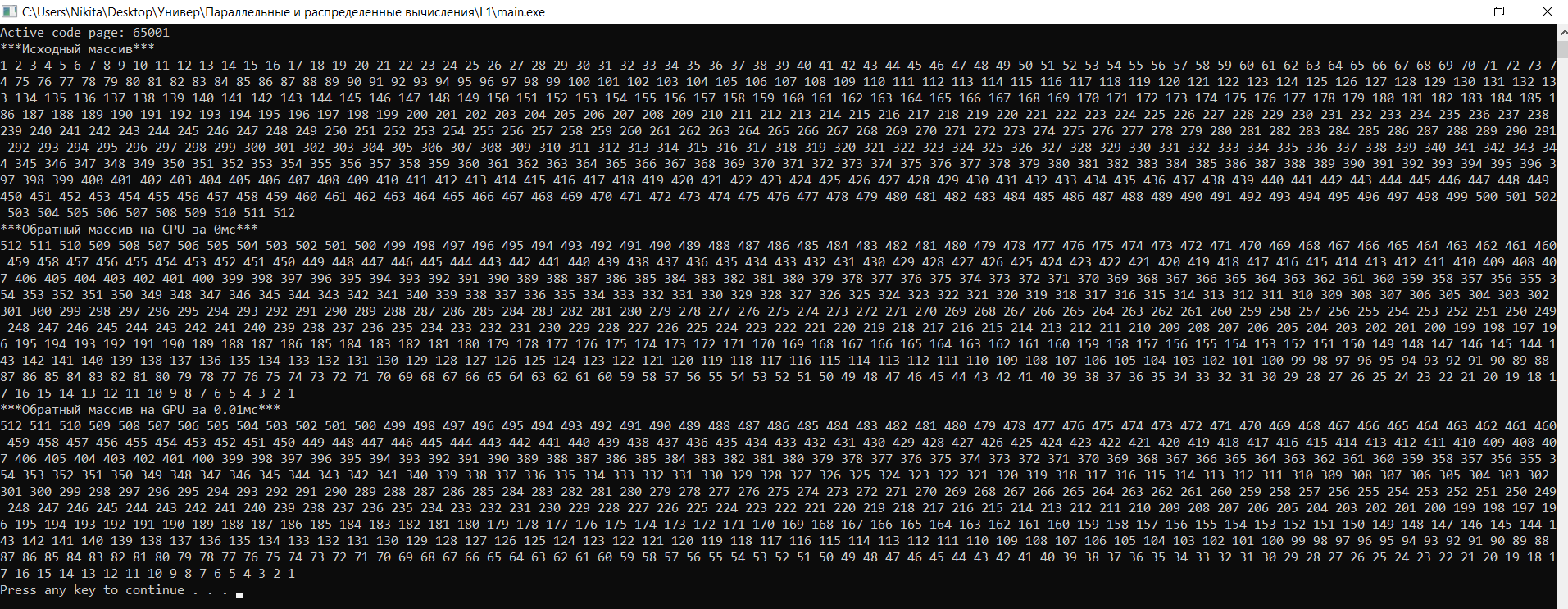
На процессоре

В цикле от 0 до n/2 поменять местами i-ый и n-i-тый элемент

На видеокарте

Запустить n/2 блоков с ссылкой на этот массив и поменять местами элемент равный своему индексу и n-свой индекс элемент.

# 3. Результат работы программы



Active code page: 65001 \*\*\*Исходный массив\*\*\* 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 \*\*\*Обратный массив на CPU за 0мс\*\*\* 512 511 510 509 508 507 506 505 504 503 502 501 500 499 498 497 496 495 494 493 492 491 490 489 488 487 486 485 484 483 482 481 480 479 478 477 476 475 474 473 472 471 470 469 468 467 466 465 464 463 462 461 460 459 458 457 456 455 454 453 452 451 450 449 448 447 446 445 444 443 442 441 440 439 438 437 436 435 434 433 432 431 430 429 428 427 426 425 424 423 422 421 420 419 418 417 416 415 414 413 412 411 410 409 408 407 406 405 404 403 402 401 400 399 398 397 396 395 394 393 392 391 390 389 388 387 386 385 384 383 382 381 380 379 378 377 376 375 374 373 372 371 370 369 368 367 366 365 364 363 362 361 360 359 358 357 356 355 354 353 352 351 350 349 348 347 346 345 344 343 342 341 340 339 338 337 336 335 334 333 332 331 330 329 328 327 326 325 324 323 322 321 320 319 318 317 316 315 314 313 312 311 310 309 308 307 306 305 304 303 302 301 300 299 298 297 296 295 294 293 292 291 290 289 288 287 286 285 284 283 282 281 280 279 278 277 276 275 274 273 272 271 270 269 268 267 266 265 264 263 262 261 260 259 258 257 256 255 254 253 252 251 250 249 248 247 246 245 244 243 242 241 240 239 238 237 236 235 234 233 232 231 230 229 228 227 226 225 224 223 222 221 220 219 218 217 216 215 214 213 212 211 210 209 208 207 206 205 204 203 202 201 200 199 198 197 196 195 194 193 192 191 190 189 188 187 186 185 184 183 182 181 180 179 178 177 176 175 174 173 172 171 170 169 168 167 166 165 164 163 162 161 160 159 158 157 156 155 154 153 152 151 150 149 148 147 146 145 144 143 142 141 140 139 138 137 136 135 134 133 132 131 130 129 128 127 126 125 124 123 122 121 120 119 118 117 116 115 114 113 112 111 110 109 108 107 106 105 104 103 102 101 100 99 98 97 96 95 94 93 92 91 90 89 88 87 86 85 84 83 82 81 80 79 78 77 76 75 74 73 72 71 70 69 68 67 66 65 64 63 62 61 60 59 58 57 56 55 54 53 52 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 \*\*\*Обратный массив на GPU за 0.01мс\*\*\* 512 511 510 509 508 507 506 505 504 503 502 501 500 499 498 497 496 495 494 493 492 491 490 489 488 487 486 485 484 483 482 481 480 479 478 477 476 475 474 473 472 471 470 469 468 467 466 465 464 463 462 461 460 459 458 457 456 455 454 453 452 451 450 449 448 447 446 445 444 443 442 441 440 439 438 437 436 435 434 433 432 431 430 429 428 427 426 425 424 423 422 421 420 419 418 417 416 415 414 413 412 411 410 409 408 407 406 405 404 403 402 401 400 399 398 397 396 395 394 393 392 391 390 389 388 387 386 385 384 383 382 381 380 379 378 377 376 375 374 373 372 371 370 369 368 367 366 365 364 363 362 361 360 359 358 357 356 355 354 353 352 351 350 349 348 347 346 345 344 343 342 341 340 339 338 337 336 335 334 333 332 331 330 329 328 327 326 325 324 323 322 321 320 319 318 317 316 315 314 313 312 311 310 309 308 307 306 305 304 303 302 301 300 299 298 297 296 295 294 293 292 291 290 289 288 287 286 285 284 283 282 281 280 279 278 277 276 275 274 273 272 271 270 269 268 267 266 265 264 263 262 261 260 259 258 257 256 255 254 253 252 251 250 249 248 247 246 245 244 243 242 241 240 239 238 237 236 235 234 233 232 231 230 229 228 227 226 225 224 223 222 221 220 219 218 217 216 215 214 213 212 211 210 209 208 207 206 205 204 203 202 201 200 199 198 197 196 195 194 193 192 191 190 189 188 187 186 185 184 183 182 181 180 179 178 177 176 175 174 173 172 171 170 169 168 167 166 165 164 163 162 161 160 159 158 157 156 155 154 153 152 151 150 149 148 147 146 145 144 143 142 141 140 139 138 137 136 135 134 133 132 131 130 129 128 127 126 125 124 123 122 121 120 119 118 117 116 115 114 113 112 111 110 109 108 107 106 105 104 103 102 101 100 99 98 97 96 95 94 93 92 91 90 89 88 87 86 85 84 83 82 81 80 79 78 77 76 75 74 73 72 71 70 69 68 67 66 65 64 63 62 61 60 59 58 57 56 55 54 53 52 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 Press any key to continue . . .

# 4. Вывод:

Познакомился с основами программирования на cuda. Время работы на видеокарте больше чем на процессоре. Нет смысла использовать cuda для задач такой маленькой размерности.

# 5. Код программы

#include "cuda\_runtime.h"      // CUDA runtime

#include "device\_launch\_parameters.h"

#include <stdio.h>

#include <stdlib.h>

#include <ctime>

#define N 512

// Фуккция, которая выполняется на GPU

\_\_global\_\_ void add( int \*mass){

    int tid = blockIdx.x; // Получить собственный индекс

    if (tid < N){

        int t = mass[tid];

        mass[tid] = mass[N - tid - 1];

        mass[N - tid - 1] = t;

    }

}

int main(){

    system("chcp 65001"); // Для Русских букв в консоли

    // Инициализация переменных

    int mass[N];

    int mass\_GPU[N];

    int mass\_CPU[N];

    for (int i = 0; i < N; i++){

        mass[i] = i+1;

        mass\_CPU[i] = i+1;

    }

    printf("\*\*\*Исходный массив\*\*\*\n");

    for (int i = 0; i < N; i++){

        printf("%i ",mass[i]);

    }

    printf("\n");

    int start\_CPU, time\_CPU;

    start\_CPU = clock();

    //Обратный массив на CPU

    for (int i = 0; i < int(N/2); i++){

        int t = mass\_CPU[i];

        mass\_CPU[i] = mass\_CPU[N - i - 1];

        mass\_CPU[N - i - 1] = t;

    }

    time\_CPU = clock() - start\_CPU;

    printf("\*\*\*Обратный массив на CPU за %iмс\*\*\*\n", time\_CPU);

    for (int i = 0; i < N; i++){

        printf("%i ",mass\_CPU[i]);

    }

    printf("\n");

    int \*dev\_mass; // Адрес масива на GPU

    // Выделить память на GPU и сохранить её адрес в переменную

    cudaMalloc( (void\*\*)&dev\_mass, N \* sizeof(int));

    // Копировать данные по адресу mass, размером N\*sizeof(int) в адрес dev\_mass. Копирование с устройства на GPU

    cudaMemcpy(dev\_mass, mass, N \* sizeof(int), cudaMemcpyHostToDevice);

    // Чото для времени GPU

    cudaEvent\_t start\_GPU, stop\_GPU;

    float gpuTime = 0.0f;

    cudaEventCreate ( &start\_GPU );

    cudaEventCreate ( &stop\_GPU );

    cudaEventRecord ( start\_GPU, 0 );

        // Запуск функции. N/2 потоков

        add<<<int(N/2),1>>> (dev\_mass);

    // Чото для времени GPU

    cudaEventRecord ( stop\_GPU, 0 );

    cudaEventSynchronize ( stop\_GPU );

    cudaEventElapsedTime ( &gpuTime, start\_GPU, stop\_GPU );

    // Копировать данные по адресу dev\_mass, размером N\*sizeof(int) в адрес mass\_GPU. Копирование с GPU на устройство

    cudaMemcpy(mass\_GPU, dev\_mass, N \* sizeof(int), cudaMemcpyDeviceToHost);

    printf("\*\*\*Обратный массив на GPU за %.2fмс\*\*\*\n", gpuTime);

    for (int i = 0; i < N; i++){

        printf("%i ",mass\_GPU[i]);

    }

    printf("\n");

    // Отчистить память

    cudaFree(dev\_mass);

    system("pause");

    return 0;

}