Московский Авиационный Институт

(Национальный Исследовательский Университет)

Институт №8 “Компьютерные науки и прикладная математика”

Кафедра №806 “Вычислительная математика и программирование”

# Лабораторная работа №2 по курсу

# «Операционные системы»

Группа: М8О-210Б-23

Студент: Сетраков Ф.С.

Преподаватель: Бахарев В.Д. (ФИИТ)

Оценка: _____

Дата: 12.12.24

Москва, 2024

# Постановка задачи

**Вариант 5.**

**Составить программу на языке Си, обрабатывающую данные в многопоточном режиме. При обработки использовать стандартные средства создания потоков операционной системы (Windows/Unix). Ограничение максимального количества потоков, работающих в один момент времени, должно быть задано ключом запуска вашей программы. Так же необходимо уметь продемонстрировать количество потоков, используемое вашей программой с помощью стандартных средств операционной системы. В отчете привести исследование зависимости ускорения и эффективности алгоритма от входных данных и количества потоков. Получившиеся результаты необходимо объяснить.**

**Отсортировать массив целых чисел при помощи чётной-нечётной сортировки Бетчера.**

# Общий метод и алгоритм решения

Использованные системные вызовы:

- write; – записывает данные в файл.
- read - читает данные из файла
- pthread_create - содаёт поток
- pthread_join - ожидание множества потоков
- pthread_mutex_init - создание мьютекса
- pthread_mutex_destroy - уничтожение мьютекса
- pthread_mutex_lock - блокировка мьютекса
- pthread_mutex_unlock - разброкировка мьютекса
- gettimeofday - получение текущего времени (для бенчмарка)

**В лабораторной работе я использовал собственную реализацию вывода строки в stdout (iolib/std_out). Которая использует системный вызов write.**

**В задании требуется реализовать чётную-нечётную сортировку Бетчера. Идея данный алгоритм схож с сортировкой слиянием, но в отличие от неё использует чётно-нечётные перестановки. Идея алгоритма в том, что мы последовательно рассматриваем подмассивы разных размеров, как и в сортировке слиянием, мы будем их сливать (при помощи операции compare and exchange), но в отличии от сортировки слиянием, здесь мы используем итеративный подход вместо рекурсии и вместо разбиения массива посередине, мы разбиваем массив на чётную половину и нечётную. Сам алгоритм можно описать при помощи псевдокода следующим образом:**

```
for p = 1, 2, 4, 8, ... # пока  p < n

  for k = p, p/2, p/4, p/8, ... # пока >= 1

    for j = mod(k,p) to (n-1-k) с шагом 2k

      for i = 0 to min(k-1, n-j-k-1) с единичным шагом

        if floor((i+j) / (p*2)) == floor((i+j+k) / (p*2))

          compare and sort elements (i+j) and (i+j+k)
```

**Для того чтобы распараллелить данный алгоритм, вынесем цикл с переменной j в отдельную функцию, плюс будем вычислять отступ с учётом кол-ва потоков и номером текущего потока, таким образом. Для того, чтобы избежать race condition при обращении к исходному вектору, будем использовать мьютекс. Будем использовать вызов pthread_join для ожидания списка потоков.**

Код программы

**main.cpp**

```
##include <vector>
#include <pthread.h>
#include <mutex>
#include <sys/time.h>
#include "iolib/std.h"

pthread_mutex_t mutex;

struct ThreadData
{
	std::vector<int> *sequence;
	int n;
	int p;
	int k;
	int threadId;
	int numThreads;
};

void compare_and_exchange(int &a, int &b)
{
	if (b < a)
	{
	std::swap(a, b);
	}
}

void *thread_function(void *arg)
{
	ThreadData *data = (ThreadData *)arg;
	std::vector<int> &sequence = *data->sequence;
	int n = data->n;
	int p = data->p;
	int k = data->k;
	int threadId = data->threadId;
	int numThreads = data->numThreads;

	for (int j = k % p + threadId * 2 * k; j <= n - 1 - k; j += 2 * k * numThreads)
	{
	for (int i = 0; i <= std::min(k - 1, n - j - k - 1); ++i)
	{
		pthread_mutex_lock(&mutex);
		if (std::floor((i + j) / (p * 2)) == std::floor((i + j + k) / (p * 2)))
		{
		compare_and_exchange(sequence[i + j], sequence[i + j + k]);
		}
		pthread_mutex_unlock(&mutex);
	}
	}

	pthread_exit(NULL);
}

void betcher_sort(std::vector<int> &v, int max_threads)
{
	int n = v.size();
	for (int p = 1; p < n; p *= 2)
	{
	for (int k = p; k >= 1; k /= 2)
	{
		pthread_t threads[max_threads];
		ThreadData threadData[max_threads];

		for (int t = 0; t < max_threads; ++t)
		{
		threadData[t] = {&v, n, p, k, t, max_threads};
		pthread_create(&threads[t], NULL, thread_function, (void *)&threadData[t]);
		}
```

```
            for (int t = 0; t < max_threads; ++t)
            {
            pthread_join(threads[t], NULL);
            }
      }
      }
}

int main(int argc, char **argv)
{
      if (argc != 3)
      {
      std_out("invalid numer of args. enter max number of threads and array len\n");
      return 1;
      }
      int max_threads = atoi(argv[1]);
      int len = atoi(argv[2]);

      std::srand(std::time(0));

      std::string buffer;
      std::vector<int> unsorted_vector(len);

      for (int i = 0; i < len; i++)
      {
      unsorted_vector[i] = std::rand() % len;
      }
      std_out("Unsorted array:\n");
      for (int i = 0; i < len; i++)
      {
      std_out(std::to_string(unsorted_vector[i]) + " ");
      }
      std_out("\nSorted array:\n");

      struct timeval start, end;
      pthread_mutex_init(&mutex, NULL);
      gettimeofday(&start, NULL);
      betcher_sort(unsorted_vector, max_threads);
      gettimeofday(&end, NULL);
      double time_spent = (end.tv_sec - start.tv_sec) + (end.tv_usec - start.tv_usec) / 1e6;

      for (int i = 0; i < len; i++)
      {
      std_out(std::to_string(unsorted_vector[i]) + " ");
      }
      std_out("\n");
      std_out("Time elapsed:" + std::to_string(time_spent) + " s\n");
      pthread_mutex_destroy(&mutex);
      return 0;
}
```

**#std.cpp**

```
#include "std.h"

int std_in(std::string* message) {
        return file_scan(STDIN_FILENO, message);
}

void std_out(const std::string& message) {
        file_print(STDOUT_FILENO, message);
}

void log_stderr(const std::string& message) {
        file_print(STDERR_FILENO, message);
}

int file_scan(int input_file, std::string* message) {
        char buffer[DEFAULT_BUFF_SIZE];
        std::string output;
        ssize_t bytes = read(input_file, buffer, DEFAULT_BUFF_SIZE);
        for (int i = 0; i < bytes; i++) {
        output += buffer[i];
        }
        *message = output;
        return (int) bytes;
}

int file_print(int output_file, const std::string& message) {
        ssize_t bytes = write(output_file, message.c_str(), message.size());
        return (int) bytes;
}
```

**#std.h**

```
#ifndef SRC_STD_H
#define SRC_STD_H

#include <string>
#include <csignal>
#include <unistd.h>

#define DEFAULT_BUFF_SIZE 1024

int std_in(std::string *);

void std_out(const std::string &);

void log_stderr(const std::string &);

int file_scan(int, std::string *);

int file_print(int, const std::string &);

#endif
```

# Протокол работы программы

```
setrakovfs@osx ~/oslabs/OSLABS/lab2/src $ docker run -it --rm my-cpp-strace-image strace ./a.out 4 20
execve("./a.out", ["./a.out", "4", "20"], 0x7ffd9aadb670 /* 4 vars */) = 0
brk(NULL)                               = 0x613d66ede000
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x799dedbb5000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=9303, ...}) = 0
mmap(NULL, 9303, PROT_READ, MAP_PRIVATE, 3, 0) = 0x799dedbb2000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libstdc++.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0644, st_size=2592224, ...}) = 0
mmap(NULL, 2609472, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x799ded934000
mmap(0x799ded9d1000, 1343488, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x9d000) =
0x799ded9d1000
mmap(0x799dedb19000, 552960, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1e5000) =
0x799dedb19000
mmap(0x799dedba0000, 57344, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x26b000) =
0x799dedba0000
mmap(0x799dedbae000, 12608, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) =
0x799dedbae000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libgcc_s.so.1", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0644, st_size=183024, ...}) = 0
mmap(NULL, 185256, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x799ded906000
mmap(0x799ded90a000, 147456, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x4000) =
0x799ded90a000
mmap(0x799ded92e000, 16384, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) =
0x799ded92e000
mmap(0x799ded932000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x2b000) =
0x799ded932000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220\243\2\0\0\0\0\0"..., 832) = 832
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
fstat(3, {st_mode=S_IFREG|0755, st_size=2125328, ...}) = 0
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
mmap(NULL, 2170256, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x799ded6f4000
mmap(0x799ded71c000, 1605632, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) =
0x799ded71c000
mmap(0x799ded8a4000, 323584, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1b0000) =
0x799ded8a4000
mmap(0x799ded8f3000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1fe000) =
0x799ded8f3000
mmap(0x799ded8f9000, 52624, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) =
0x799ded8f9000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libm.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\0\0\0\0\0\0\0\0"..., 832) = 832
fstat(3, {st_mode=S_IFREG|0644, st_size=952616, ...}) = 0
mmap(NULL, 950296, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x799ded60b000
mmap(0x799ded61b000, 520192, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x10000) =
0x799ded61b000
mmap(0x799ded69a000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x8f000) =
0x799ded69a000
mmap(0x799ded6f2000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0xe7000) =
0x799ded6f2000
close(3)                                = 0
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x799ded609000
mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x799ded606000
arch_prctl(ARCH_SET_FS, 0x799ded606740) = 0
```

```
set_tid_address(0x799ded606a10)         = 8
set_robust_list(0x799ded606a20, 24)     = 0
rseq(0x799ded607060, 0x20, 0, 0x53053053) = 0
mprotect(0x799ded8f3000, 16384, PROT_READ) = 0
mprotect(0x799ded6f2000, 4096, PROT_READ) = 0
mprotect(0x799ded932000, 4096, PROT_READ) = 0
mprotect(0x799dedba0000, 45056, PROT_READ) = 0
mprotect(0x613d66317000, 4096, PROT_READ) = 0
mprotect(0x799dedbed000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
munmap(0x799dedbb2000, 9303)            = 0
futex(0x799dedbae7bc, FUTEX_WAKE_PRIVATE, 2147483647) = 0
getrandom("\x57\x9a\x4f\x95\x21\x41\xd9\x58", 8, GRND_NONBLOCK) = 8
brk(NULL)                               = 0x613d66ede000
brk(0x613d66eff000)                     = 0x613d66eff000
write(1, "Unsorted array:\n", 16Unsorted array:
)      = 16
write(1, "1 ", 21 )                     = 2
write(1, "12 ", 312 )                   = 3
write(1, "14 ", 314 )                   = 3
write(1, "10 ", 310 )                   = 3
write(1, "3 ", 23 )                     = 2
write(1, "19 ", 319 )                   = 3
write(1, "5 ", 25 )                     = 2
write(1, "16 ", 316 )                   = 3
write(1, "0 ", 20 )                     = 2
write(1, "5 ", 25 )                     = 2
write(1, "1 ", 21 )                     = 2
write(1, "3 ", 23 )                     = 2
write(1, "10 ", 310 )                   = 3
write(1, "10 ", 310 )                   = 3
write(1, "10 ", 310 )                   = 3
write(1, "3 ", 23 )                     = 2
write(1, "4 ", 24 )                     = 2
write(1, "8 ", 28 )                     = 2
write(1, "9 ", 29 )                     = 2
write(1, "19 ", 319 )                   = 3
write(1, "\nSorted array:\n", 15
Sorted array:
)      = 15
gettimeofday({tv_sec=1733996165, tv_usec=654671}, NULL) = 0
rt_sigaction(SIGRT_1, {sa_handler=0x799ded78d520, sa_mask=[],
sa_flags=SA_RESTORER|SA_ONSTACK|SA_RESTART|SA_SIGINFO, sa_restorer=0x799ded739320}, NULL, 8) = 0
rt_sigprocmask(SIG_UNBLOCK, [RTMIN RT_1], NULL, 8) = 0
mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) = 0x799dece00000
mprotect(0x799dece01000, 8388608, PROT_READ|PROT_WRITE) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone3({flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLON
E_PARENT_SETTID|CLONE_CHILD_CLEARTID, child_tid=0x799ded600990, parent_tid=0x799ded600990,
exit_signal=0, stack=0x799dece00000, stack_size=0x7fff80, tls=0x799ded6006c0}, 88) = -1 ENOSYS
(Function not implemented)
clone(child_stack=0x799ded5fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[9], tls=0x799ded6006c0, child_tidptr=0x799ded600990) = 9
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) = 0x799dec400000
mprotect(0x799dec401000, 8388608, PROT_READ|PROT_WRITE) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799decbfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[10], tls=0x799decc006c0, child_tidptr=0x799decc00990) = 10
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
```

```
mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) = 0x799de7600000
mprotect(0x799de7601000, 8388608, PROT_READ|PROT_WRITE) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de7dfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[11], tls=0x799de7e006c0, child_tidptr=0x799de7e00990) = 11
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
mmap(NULL, 8392704, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS|MAP_STACK, -1, 0) = 0x799de6c00000
mprotect(0x799de6c01000, 8388608, PROT_READ|PROT_WRITE) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de73fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[12], tls=0x799de74006c0, child_tidptr=0x799de7400990) = 12
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de73fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[13], tls=0x799de74006c0, child_tidptr=0x799de7400990) = 13
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de7dfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[14], tls=0x799de7e006c0, child_tidptr=0x799de7e00990) = 14
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799decbfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[15], tls=0x799decc006c0, child_tidptr=0x799decc00990) = 15
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799ded5fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[16], tls=0x799ded6006c0, child_tidptr=0x799ded600990) = 16
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
futex(0x799ded600990, FUTEX_WAIT_BITSET|FUTEX_CLOCK_REALTIME, 16, NULL, FUTEX_BITSET_MATCH_ANY) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799ded5fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[17], tls=0x799ded6006c0, child_tidptr=0x799ded600990) = 17
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799decbfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[18], tls=0x799decc006c0, child_tidptr=0x799decc00990) = 18
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de7dfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[19], tls=0x799de7e006c0, child_tidptr=0x799de7e00990) = 19
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de73fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[20], tls=0x799de74006c0, child_tidptr=0x799de7400990) = 20
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de73fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[21], tls=0x799de74006c0, child_tidptr=0x799de7400990) = 21
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de7dfff70,
```

```
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[22], tls=0x799de7e006c0, child_tidptr=0x799de7e00990) = 22
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799decbfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[0], tls=0x799decc006c0, child_tidptr=0x799decc00990) = 23
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799ded5fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[24], tls=0x799ded6006c0, child_tidptr=0x799ded600990) = 24
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799ded5fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[25], tls=0x799ded6006c0, child_tidptr=0x799ded600990) = 25
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799decbfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[0], tls=0x799decc006c0, child_tidptr=0x799decc00990) = 26
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de7dfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[27], tls=0x799de7e006c0, child_tidptr=0x799de7e00990) = 27
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de73fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[0], tls=0x799de74006c0, child_tidptr=0x799de7400990) = 28
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de73fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[29], tls=0x799de74006c0, child_tidptr=0x799de7400990) = 29
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de7dfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[30], tls=0x799de7e006c0, child_tidptr=0x799de7e00990) = 30
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799decbfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[31], tls=0x799decc006c0, child_tidptr=0x799decc00990) = 31
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799ded5fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[0], tls=0x799ded6006c0, child_tidptr=0x799ded600990) = 32
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799ded5fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[33], tls=0x799ded6006c0, child_tidptr=0x799ded600990) = 33
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799decbfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[34], tls=0x799decc006c0, child_tidptr=0x799decc00990) = 34
```

```
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799de7dfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[0], tls=0x799de7e006c0, child_tidptr=0x799de7e00990) = 35
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799de73fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[0], tls=0x799de74006c0, child_tidptr=0x799de7400990) = 36
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799de73fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[37], tls=0x799de74006c0, child_tidptr=0x799de7400990) = 37
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799de7dfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[38], tls=0x799de7e006c0, child_tidptr=0x799de7e00990) = 38
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799decbfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[39], tls=0x799decc006c0, child_tidptr=0x799decc00990) = 39
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799ded5fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[40], tls=0x799ded6006c0, child_tidptr=0x799ded600990) = 40
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799ded5fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[0], tls=0x799ded6006c0, child_tidptr=0x799ded600990) = 41
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799decbfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[42], tls=0x799decc006c0, child_tidptr=0x799decc00990) = 42
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799de7dfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[0], tls=0x799de7e006c0, child_tidptr=0x799de7e00990) = 43
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799de73fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[44], tls=0x799de74006c0, child_tidptr=0x799de7400990) = 44
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799de73fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[45], tls=0x799de74006c0, child_tidptr=0x799de7400990) = 45
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799de7dfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[46], tls=0x799de7e006c0, child_tidptr=0x799de7e00990) = 46
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
```

```
clone(child_stack=0x799decbfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[47], tls=0x799decc006c0, child_tidptr=0x799decc00990) = 47
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799ded5fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[48], tls=0x799ded6006c0, child_tidptr=0x799ded600990) = 48
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799ded5fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[0], tls=0x799ded6006c0, child_tidptr=0x799ded600990) = 49
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799decbfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[50], tls=0x799decc006c0, child_tidptr=0x799decc00990) = 50
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de7dfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[51], tls=0x799de7e006c0, child_tidptr=0x799de7e00990) = 51
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de73fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[52], tls=0x799de74006c0, child_tidptr=0x799de7400990) = 52
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de73fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[53], tls=0x799de74006c0, child_tidptr=0x799de7400990) = 53
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de7dfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[54], tls=0x799de7e006c0, child_tidptr=0x799de7e00990) = 54
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799decbfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[0], tls=0x799decc006c0, child_tidptr=0x799decc00990) = 55
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799ded5fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[0], tls=0x799ded6006c0, child_tidptr=0x799ded600990) = 56
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799ded5fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[57], tls=0x799ded6006c0, child_tidptr=0x799ded600990) = 57
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799decbfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[0], tls=0x799decc006c0, child_tidptr=0x799decc00990) = 58
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)   = 0
clone(child_stack=0x799de7dfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
```

```
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[59], tls=0x799de7e006c0, child_tidptr=0x799de7e00990) = 59
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799de73fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[0], tls=0x799de74006c0, child_tidptr=0x799de7400990) = 60
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799de73fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[0], tls=0x799de74006c0, child_tidptr=0x799de7400990) = 61
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799de7dfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[0], tls=0x799de7e006c0, child_tidptr=0x799de7e00990) = 62
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799decbfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[63], tls=0x799decc006c0, child_tidptr=0x799decc00990) = 63
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799ded5fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[0], tls=0x799ded6006c0, child_tidptr=0x799ded600990) = 64
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799ded5fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[65], tls=0x799ded6006c0, child_tidptr=0x799ded600990) = 65
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799decbfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[66], tls=0x799decc006c0, child_tidptr=0x799decc00990) = 66
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799de7dfff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[67], tls=0x799de7e006c0, child_tidptr=0x799de7e00990) = 67
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
rt_sigprocmask(SIG_BLOCK, ~[], [], 8)    = 0
clone(child_stack=0x799de73fff70,
flags=CLONE_VM|CLONE_FS|CLONE_FILES|CLONE_SIGHAND|CLONE_THREAD|CLONE_SYSVSEM|CLONE_SETTLS|CLONE_PARENT
_SETTID|CLONE_CHILD_CLEARTID, parent_tid=[68], tls=0x799de74006c0, child_tidptr=0x799de7400990) = 68
rt_sigprocmask(SIG_SETMASK, [], NULL, 8) = 0
gettimeofday({tv_sec=1733996165, tv_usec=814816}, NULL) = 0
write(1, "0 ", 20 )                      = 2
write(1, "1 ", 21 )                      = 2
write(1, "1 ", 21 )                      = 2
write(1, "3 ", 23 )                      = 2
write(1, "3 ", 23 )                      = 2
write(1, "3 ", 23 )                      = 2
write(1, "4 ", 24 )                      = 2
write(1, "5 ", 25 )                      = 2
write(1, "5 ", 25 )                      = 2
write(1, "8 ", 28 )                      = 2
write(1, "9 ", 29 )                      = 2
write(1, "10 ", 310 )                    = 3
write(1, "10 ", 310 )                    = 3
write(1, "10 ", 310 )                    = 3
```

```
write(1, "10 ", 310 )                   = 3
write(1, "12 ", 312 )                   = 3
write(1, "14 ", 314 )                   = 3
write(1, "16 ", 316 )                   = 3
write(1, "19 ", 319 )                   = 3
write(1, "19 ", 319 )                   = 3
write(1, "\n", 1
)                           = 1
write(1, "Time elapsed:0.160145 s\n", 24Time elapsed:0.160145 s
) = 24
exit_group(0)                           = ?
+++ exited with 0 +++
```

# Вывод

**Задание мне показалось интересным, так как многопоточное программирование - одна из самых частых и важных тем. Я столкнулся с многими проблемами, самой сложной задачей, мне показалось распараллеливание исходного алгоритма**

| Число потоков | Время исполнения (с) | Ускорение | Эффективность |
|---|---|---|---|
| 1 | 1.006487 | 1 | 1 |
| 2 | 0.582498 | 1.72 | 0.86 |
| 4 | 0.332793 | 3.02 | 0.98 |
| 8 | 0.267251 | 3.76 | 0.47 |
| 16 | 0.225936 | 4.45 | 0.27 |