**Код программы**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <time.h>

#include <unistd.h>

#include <pthread.h>

#include <iostream>

using namespace std;

char\*\* filenames;

pthread\_t\* write\_threads;

void\* file\_create(void\*);

void\* file\_read(void\*);

void thread\_info(int, const char\*, unsigned int);

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

if(argc != 2){

cout << "Using: \n";

cout << "main [n]\n";

cout << " n: [1-50] - amount of pairs\n";

cout << " [ -1 ] - clear files\n";

return EXIT\_FAILURE;

}

if(atoi(argv[1]) == -1){

system("rm -r created\_files");

system("mkdir created\_files");

cout << "Files deleted\n";

return EXIT\_SUCCESS;

}

if(atoi(argv[1]) <= 0 || atoi(argv[1]) >= 50){

cout << "Incorrect amount of pairs!\n";

return EXIT\_FAILURE;

}

//---setup

void\* ret\_value;

unsigned int pairs\_amount(atoi(argv[1]));

srand(time(NULL));

write\_threads = new pthread\_t[pairs\_amount];

pthread\_t\* read\_threads = new pthread\_t[pairs\_amount];

filenames = new char\*[pairs\_amount];

//--------

for(int pair = 0; pair < pairs\_amount; ++pair){

filenames[pair] = new char[256];

if(pthread\_create(&write\_threads[pair], NULL, file\_create, new int(pair)) != 0){

cout << "PTHREAD\_CREATE::creating file thread creation failed\n";

return EXIT\_FAILURE;

}

}

for(int pair = 0; pair < pairs\_amount; pair++){

if(pthread\_create(&read\_threads[pair], NULL, file\_read, new int(pair)) != 0){

cout << "PTHREAD\_CREATE::reading file thread creation failed\n";

return EXIT\_FAILURE;

}

pthread\_join(read\_threads[pair], &ret\_value);

if(\*(int\*)ret\_value != EXIT\_SUCCESS){

cout << "PAIR #" << pair+1 << "::failed\n";

}

}

return EXIT\_SUCCESS;

}

void\* file\_create(void\* file\_index){

int ind(\*(int\*)file\_index);

sprintf(filenames[ind], "created\_files/%lu.txt", pthread\_self());

int\* creation\_status = new int(EXIT\_SUCCESS);

unsigned int length(rand() % 50000);

FILE\* file(fopen(filenames[ind], "w"));

if(file){

for(int i = 0 ; i < length; ++i){

char symbol(rand() % 40 + 40);

fputc(symbol, file);

}

if(fclose(file) != 0){

cout << "FILE\_CREATE::fclose::Can\'t close file (" << filenames[ind] << ")\n";

\*creation\_status = EXIT\_FAILURE;

pthread\_exit((void\*)creation\_status);

}

}else{

cout << "FILE\_CREATE::fopen::Can\'t open file (" << filenames[ind] << ")\n";

\*creation\_status = EXIT\_FAILURE;

pthread\_exit((void\*)creation\_status);

}

thread\_info(ind, "Writting thread", length);

pthread\_exit((void\*)creation\_status);

}

void\* file\_read(void\* file\_index){

int ind(\*(int\*) file\_index);

int\* read\_status = new int(EXIT\_SUCCESS);

void\* file\_create\_ret\_value(NULL);

pthread\_join(write\_threads[ind], &file\_create\_ret\_value);

if(\*(int\*)file\_create\_ret\_value != EXIT\_SUCCESS){

\*read\_status = EXIT\_FAILURE;

pthread\_exit((void\*)read\_status);

}

unsigned int length(0);

FILE\* file(fopen(filenames[ind], "r"));

if(file){

while(fgetc(file) != EOF){

++length;

}

if(fclose(file) != 0){

cout << "FILE\_READ::fclose::Can\'t close file (" << filenames[ind] << ")\n";

\*read\_status = EXIT\_FAILURE;

pthread\_exit((void\*)read\_status);

}

}else{

cout << "FILE\_READ::fopen::Can\'t open file (" << filenames[ind] << ")\n";

\*read\_status = EXIT\_FAILURE;

pthread\_exit((void\*)read\_status);

}

thread\_info(ind, "Reading thread", length);

pthread\_exit((void\*)read\_status);

}

void thread\_info(int num\_of\_pair, const char\* type, unsigned int symbols\_amount){

time\_t current\_time(time(NULL));

cout << " Thread info:\n";

cout << " Number of pair : " << num\_of\_pair + 1 << "\n";

cout << " Type : " << type << "\n";

cout << " ThreadID : " << pthread\_self() << "\n";

cout << " pid : " << getpid() << "\n";

cout << " ppid : " << getppid() << "\n";

cout << " Time : " << ctime(&current\_time);

cout << " File name : " << filenames[num\_of\_pair] << "\n";

cout << " Length : " << symbols\_amount << "\n";

}

**Пример работы программы**

