# Лабораторная работа №5

Исследование возможностей управления памятью и обмена данными между процессами в ОС Windows

## Цель работы

Изучить возможности программного интерфейса приложений (API) операционных систем Windows по управления памятью и обмена данными между процессами. Приобрести практические навыки использования Win API для управления памятью и обмена данными между процессами.

## Постановка задачи

Приобрести навыки работы с механизмами управления памяти с помощью средств WinAPI.

## Ход выполнения работы

### Тексты программ

Программа написана на языке программирования C, компилятор – gcc 6.3.0. Тексты программ представлены в Листингах 1 и 2.

Листинг 1 – программа Master5

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <windows.h>

#define MEMPAGE\_COUNT 3

#define MEMPAGE\_SIZE 4096

#define TOTAL\_MEMSIZE ((MEMPAGE\_COUNT) \* (MEMPAGE\_SIZE))

#define MAX 256

#define MUTEX\_NAME L"mutex\_lab5"

#define MAP\_NAME L"memshare\_lab5"

typedef char arr\_elem\_t;

/\* Заполнить массив случайными данными \*/

void fill\_array (arr\_elem\_t \* arrv, size\_t arrc)

{

for (size\_t i = 0; i < arrc; i++)

arrv[i] = rand() % MAX;

}

int main ()

{

/\* Создание процесса и Мьютекса \*/

HANDLE mutex = CreateMutex(NULL, TRUE, MUTEX\_NAME);

STARTUPINFO si; PROCESS\_INFORMATION pi;

memset(&si, 0, sizeof(STARTUPINFO)); memset(&pi, 0, sizeof(PROCESS\_INFORMATION));

si.cb = sizeof(STARTUPINFO); si.lpTitle = "MemSorter 3000";

CreateProcess(NULL, "MemSort.exe", 0, NULL, FALSE, CREATE\_NEW\_CONSOLE, NULL, NULL, &si, &pi);

/\* Выделение участка виртуальной памяти \*/

DWORD old\_sec\_attrs;

LPVOID base\_address = VirtualAlloc(NULL, TOTAL\_MEMSIZE, MEM\_COMMIT, PAGE\_READWRITE);

arr\_elem\_t \*arr = (arr\_elem\_t \*) base\_address;

/\* Заполнение памяти \*/

fill\_array (arr, TOTAL\_MEMSIZE / sizeof(arr\_elem\_t));

/\* Создание общей памяти \*/

HANDLE file\_map = CreateFileMapping(INVALID\_HANDLE\_VALUE, NULL, PAGE\_READWRITE, 0, TOTAL\_MEMSIZE, "memshare\_lab5");

/\* Отображение файла в адресное пространство программы \*/

LPVOID view = MapViewOfFile(file\_map, FILE\_MAP\_ALL\_ACCESS, 0, 0, TOTAL\_MEMSIZE);

/\* Копирование в файл из Виртуальной памяти \*/

CopyMemory(view, base\_address, TOTAL\_MEMSIZE);

/\* Изменение прав доступа к памяти \*/

VirtualProtect(base\_address, TOTAL\_MEMSIZE, PAGE\_READONLY, &old\_sec\_attrs);

/\* Освобождение мьютекса и ожидание завершения процесса \*/

ReleaseMutex(mutex);

WaitForSingleObject(pi.hProcess, INFINITE);

/\* Изменение прав доступа к памяти \*/

VirtualProtect(base\_address, TOTAL\_MEMSIZE, PAGE\_READWRITE, &old\_sec\_attrs);

/\* Копирование из общей памяти в виртуальную память \*/

CopyMemory(base\_address, view, TOTAL\_MEMSIZE);

/\* Печать данных на экран \*/

for (size\_t i = 0; i < TOTAL\_MEMSIZE / sizeof(arr\_elem\_t); i++)

printf("%d ", (int) arr[i]);

printf("\r\n");

/\* Освобождение памяти и Выход \*/

UnmapViewOfFile(view);

CloseHandle(file\_map);

VirtualFree(base\_address, TOTAL\_MEMSIZE, MEM\_RELEASE);

return 0;

}

Листинг 2 – программа MemSort

#include <stdio.h>

#include <stdlib.h>

#include <windows.h>

#define MEMPAGE\_COUNT 3

#define MEMPAGE\_SIZE 4096

#define TOTAL\_MEMSIZE ((MEMPAGE\_COUNT) \* (MEMPAGE\_SIZE))

#define MAX 256

#define MUTEX\_NAME L"mutex\_lab5"

#define MAP\_NAME "memshare\_lab5"

typedef char arr\_elem\_t;

void sort(arr\_elem\_t \*arrv, size\_t arrc)

{

arr\_elem\_t tmp = 0;

for (int i = arrc - 1; i >= 0; i--)

for (int j = 0; j < i; j++)

{

if (arrv[j] > arrv[j + 1])

{

tmp = arrv[j];

arrv[j] = arrv[j + 1];

arrv[j + 1] = tmp;

}

}

}

int main()

{

/\* Открытие мьютекса \*/

HANDLE mutex = OpenMutex(MUTEX\_ALL\_ACCESS, FALSE, MUTEX\_NAME);

WaitForSingleObject(mutex, INFINITE);

/\* Открытие общей памяти и отображение её в программе \*/

HANDLE shared\_mem = OpenFileMapping(FILE\_MAP\_ALL\_ACCESS, FALSE, MAP\_NAME);

arr\_elem\_t \*arr = (arr\_elem\_t \*) MapViewOfFile(shared\_mem, FILE\_MAP\_ALL\_ACCESS, 0, 0, TOTAL\_MEMSIZE);

sort(arr, TOTAL\_MEMSIZE / sizeof(arr\_elem\_t));

ReleaseMutex(mutex);

UnmapViewOfFile((LPVOID) arr);

CloseHandle(shared\_mem);

return 0;

}

### Тестовые примеры

Рисунок 5.1 демонстрирует вывод основной программы – отсортированные по возрастанию данные.

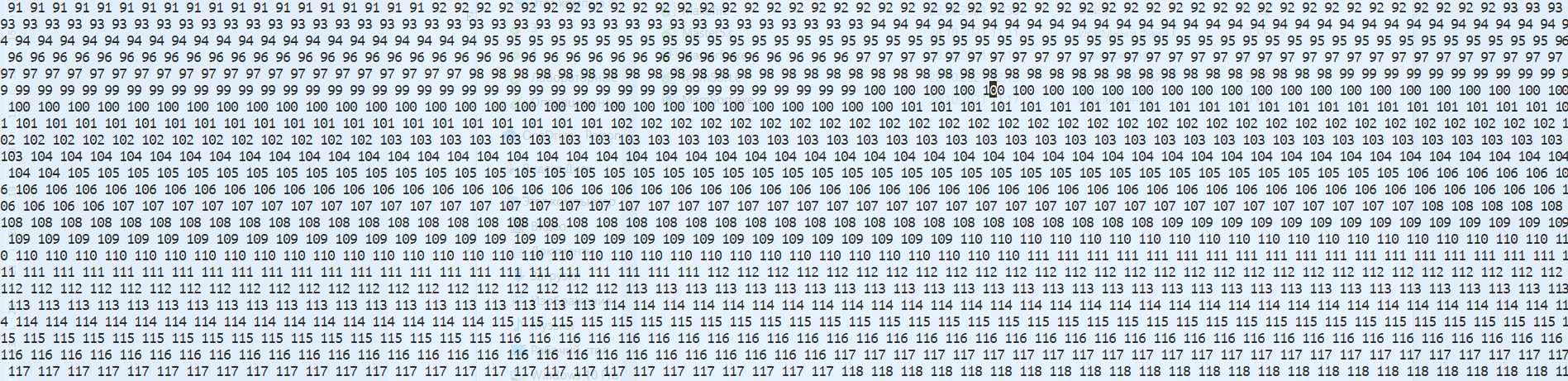


Рисунок 5.1 – Тестовый пример №1

## Вывод

При выполнении данной лабораторной работы были получены навыки создания и работы с общей памятью, выделения памяти с помощью средств WinAPI. Также были повторно закреплены навыки работы с мьютексами.