Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное  
учреждение высшего образования

«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»  
(ФГАОУ ВО «ЮФУ»)

Институт компьютерных технологий и информационной безопасности

Кафедра математического обеспечения и применения ЭВМ

ЛАБОРАТОРНАЯ РАБОТА №5

**«Обмен данными между процессами в Windows»**

по курсу «Операционные системы и системное программное обеспечение»

Вариант №7

Выполнили

студенты группы КТбо2-6 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_М. И. Домбрина

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Ю. А. Беликова

Принял

ассистент кафедры МОП ЭВМ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Н. А. Гуляев

СОДЕРЖАНИЕ

[Введение 3](#_Toc83677370)

[Ход работы 4](#_Toc83677371)

[Заключение 9](#_Toc83677372)

ВВЕДЕНИЕ

**Цель работы**

Цель настоящей работы состоит в изучении механизмов межпроцессного обмена данными в Windows и приобретении навыков использования различных средств обмена данными при разработке программ.

**Вариант № 7**

Родительский процесс A порождает два процесса-потомка (B и C) и создает два безымянных канала. Пользователь в цикле вводит произвольные целые числа от 0 до 100. Процесс A посылает введенное число по каналу процессу B, который возводит это число в квадрат и по другому каналу пересылает результат процессу C. Процесс C суммирует полученные числа. Когда сумма превышает 100000, процесс C сообщает об этом родителю. После этого процессы завершаются по нажатию <Enter>.

ХОД РАБОТЫ

Пример выполнялся при помощи языка программирования С++.

В ходе выполнения работы мы разработали 3 программы, отвечающие за процессы А, В, и С соответственно. Вначале родительский процесс А, создает 2 безымянных и порождает 2 процесса-потомка В и С. Далее в цикле процесса А пользователь вводит число, которое по первому каналу передается в процесс В. Процесс В возводит это число в квадрат и передает по второму каналу процессу С. Процесс С суммирует полученные квадраты и выводит на экран, текущую сумму. Когда сумма достигает 100000, процесс С при помощи ивента сообщает процессу А. По нажатию «ENTER» все процессы завершают свою работу.

Листинг программы:

Файл Laba5\_A.cpp

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <windows.h>

#include <stdio.h>

#include <io.h>

#include <conio.h>

#include <fcntl.h>

#include <iostream>

using std::cerr;

int main() {

SetWindowText(GetForegroundWindow(), (LPCWSTR)TEXT("Process A"));

int InputNumber = 0;

char End = 0;

FILE\* FirstPipeFile;

HANDLE hReadFirstPipe, hReadSecondPipe;

HANDLE hWriteFirstPipe, hWriteSecondPipe;

SECURITY\_ATTRIBUTES FirstPipeAttributes = { sizeof(SECURITY\_ATTRIBUTES), NULL, TRUE };

SECURITY\_ATTRIBUTES SecondPipeAttributes = { sizeof(SECURITY\_ATTRIBUTES), NULL, TRUE };

BOOL ProcBRun, ProcCRun;

HANDLE hBProcWaitEvent, hCProcWaitEvent;

HANDLE hProcWait[2];

HANDLE hNumberInputEvent, hNumberOutputEvent, hNextNumberEvent;

HANDLE hEndEnter, hEndBProcess, hEndCProcess;

PROCESS\_INFORMATION\* procBInfo = new PROCESS\_INFORMATION;

PROCESS\_INFORMATION\* procCInfo = new PROCESS\_INFORMATION;

STARTUPINFO startBInfo = { sizeof(startBInfo) };

STARTUPINFO startCInfo = { sizeof(startCInfo) };

hBProcWaitEvent = CreateEvent(NULL, FALSE, FALSE, TEXT("WaitBEvent"));

hCProcWaitEvent = CreateEvent(NULL, FALSE, FALSE, TEXT("WaitCEvent"));

hNumberInputEvent = CreateEvent(NULL, FALSE, FALSE, TEXT("NumberInputEvent"));

hNumberOutputEvent = CreateEvent(NULL, FALSE, FALSE, TEXT("NumberOutputEvent"));

hNextNumberEvent = CreateEvent(NULL, FALSE, FALSE, TEXT("NextNumberEvent"));

hEndEnter = CreateEvent(NULL, FALSE, FALSE, TEXT("EndEnter"));

hEndBProcess = CreateEvent(NULL, FALSE, FALSE, TEXT("EndBProcess"));

hEndCProcess = CreateEvent(NULL, FALSE, FALSE, TEXT("EndCProcess"));

hProcWait[0] = hBProcWaitEvent;

hProcWait[1] = hCProcWaitEvent;

CreatePipe(&hReadFirstPipe, &hWriteFirstPipe, &FirstPipeAttributes, 0);

CreatePipe(&hReadSecondPipe, &hWriteSecondPipe, &SecondPipeAttributes, 0);

FirstPipeFile = \_fdopen((HFILE)\_open\_osfhandle((intptr\_t)hWriteFirstPipe, \_O\_TEXT), "r+t");

startBInfo.lpTitle = (LPWSTR)TEXT("Process B");

startBInfo.dwFlags = STARTF\_USESTDHANDLES;

startBInfo.hStdInput = hReadFirstPipe;

startBInfo.hStdOutput = hWriteSecondPipe;

startBInfo.hStdError = GetStdHandle(STD\_ERROR\_HANDLE);

ProcBRun = CreateProcess(TEXT("Laba5\_Pr\_B.exe"), NULL, NULL, NULL, TRUE,

CREATE\_NEW\_CONSOLE, NULL, NULL, &startBInfo, procBInfo);

if (!ProcBRun) {

printf("%s", "Process B creation failed");

system("pause");

return 1;

}

startCInfo.lpTitle = (LPWSTR)TEXT("Process C");

startCInfo.dwFlags = STARTF\_USESTDHANDLES;

startCInfo.hStdInput = hReadSecondPipe;

startCInfo.hStdOutput = GetStdHandle(STD\_OUTPUT\_HANDLE);

startCInfo.hStdError = GetStdHandle(STD\_ERROR\_HANDLE);

ProcCRun = CreateProcess(TEXT("Laba5\_Pr\_C.exe"), NULL, NULL, NULL, TRUE,

CREATE\_NEW\_CONSOLE, NULL, NULL, &startCInfo, procCInfo);

if (!ProcCRun) {

printf("%s", "Process C creation failed");

system("pause");

return 1;

}

WaitForMultipleObjects(2, hProcWait, TRUE, INFINITE);

while (WaitForSingleObject(hEndEnter, 0)) {

printf("%s", "Enter a number: ");

scanf("%d", &InputNumber);

if (InputNumber > 100 || InputNumber < 0) {

printf("%s", "Number more 100 or less 0\n");

}

else {

fprintf(FirstPipeFile, "%d ", InputNumber);

fflush(FirstPipeFile);

SetEvent(hNumberInputEvent);

WaitForSingleObject(hNextNumberEvent, INFINITE);

}

}

fprintf(FirstPipeFile, "%d ", 0);

fflush(FirstPipeFile);

SetEvent(hNumberInputEvent);

SetEvent(hEndEnter);

while (End != 13) {

printf("%s", "Press ENTER\n");

End = \_getch();

}

SetEvent(hEndBProcess);

SetEvent(hEndCProcess);

CloseHandle(hWriteFirstPipe);

CloseHandle(hWriteSecondPipe);

CloseHandle(hBProcWaitEvent);

CloseHandle(hCProcWaitEvent);

CloseHandle(hNumberInputEvent);

CloseHandle(hNumberOutputEvent);

CloseHandle(hEndBProcess);

CloseHandle(hNextNumberEvent);

CloseHandle(hEndEnter);

CloseHandle(hEndCProcess);

return 0;

}

Файл Laba5\_B.cpp

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <windows.h>

#include <stdio.h>

int main() {

int \*InputNumber;

int \*OutputNumber;

HANDLE hBProcWaitEvent, hNumberInputEvent, hNumberOutputEvent;

HANDLE hAtoBMapping;

HANDLE hBtoCMapping;

HANDLE hEndEnter, hEndBProcess;

hBProcWaitEvent = OpenEvent(EVENT\_ALL\_ACCESS, FALSE, TEXT("WaitBEvent"));

hNumberInputEvent = OpenEvent(EVENT\_ALL\_ACCESS, FALSE, TEXT("NumberInputEvent"));

hNumberOutputEvent = OpenEvent(EVENT\_ALL\_ACCESS, FALSE, TEXT("NumberOutputEvent"));

hEndEnter = OpenEvent(EVENT\_ALL\_ACCESS, FALSE, TEXT("EndEnter"));

hEndBProcess = OpenEvent(EVENT\_ALL\_ACCESS, FALSE, TEXT("EndBProcess"));

hAtoBMapping = OpenFileMapping(FILE\_MAP\_WRITE, FALSE, TEXT("AtoBMapping"));

hBtoCMapping = OpenFileMapping(FILE\_MAP\_WRITE, FALSE, TEXT("BtoCMapping"));

InputNumber = (int\*)MapViewOfFile(hAtoBMapping, FILE\_MAP\_WRITE, 0, 0, 4);

OutputNumber = (int\*)MapViewOfFile(hBtoCMapping, FILE\_MAP\_WRITE, 0, 0, 4);

if (!hBProcWaitEvent) {

printf("%s", "WaitBEvent don't open");

system("pause");

return 2;

}

if (!hNumberInputEvent) {

printf("%s", "NumberInputEvent don't open");

system("pause");

return 2;

}

if (!hNumberOutputEvent) {

printf("%s", "NumberOutputEvent don't open");

system("pause");

return 2;

}

if (!hEndEnter) {

printf("%s", "EndEnter don't open");

system("pause");

return 2;

}

if (!hEndBProcess) {

printf("%s", "EndBProcess don't open");

system("pause");

return 2;

}

if (!hAtoBMapping) {

printf("%s", "AtoBMapping don't open");

system("pause");

return 3;

}

if (!hBtoCMapping) {

printf("%s", "BtoCMapping don't open");

system("pause");

return 3;

}

fflush(stdout);

SetEvent(hBProcWaitEvent);

while (WaitForSingleObject(hEndEnter, 0)) {

WaitForSingleObject(hNumberInputEvent, INFINITE);

\*OutputNumber = (\*InputNumber) \* (\*InputNumber);

SetEvent(hNumberOutputEvent);

fflush(stdout);

}

WaitForSingleObject(hEndBProcess, INFINITE);

UnmapViewOfFile(InputNumber);

UnmapViewOfFile(OutputNumber);

CloseHandle(hBProcWaitEvent);

CloseHandle(hNumberInputEvent);

CloseHandle(hNumberOutputEvent);

CloseHandle(hEndEnter);

CloseHandle(hEndBProcess);

return 0;

}

Файл Laba5\_C.cpp

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <windows.h>

#include <stdio.h>

#include <iostream>

using std::cout;

using std::endl;

int main() {

int Count = 0;

int OutputNumber = 0;

HANDLE hCProcWaitEvent, hNumberOutputEvent, hNextNumberEvent;

HANDLE hEndEnter, hEndCProcess;

hCProcWaitEvent = OpenEvent(EVENT\_ALL\_ACCESS, FALSE, TEXT("WaitCEvent"));

hNumberOutputEvent = OpenEvent(EVENT\_ALL\_ACCESS, FALSE, TEXT("NumberOutputEvent"));

hNextNumberEvent = OpenEvent(EVENT\_ALL\_ACCESS, FALSE, TEXT("NextNumberEvent"));

hEndEnter = OpenEvent(EVENT\_ALL\_ACCESS, FALSE, TEXT("EndEnter"));

hEndCProcess = OpenEvent(EVENT\_ALL\_ACCESS, FALSE, TEXT("EndCProcess"));

if (!hCProcWaitEvent) {

printf("%s", "CProcWaitEvent don't open");

system("pause");

return 2;

}

if (!hNextNumberEvent) {

printf("%s", "NextNumberEvent don't open");

system("pause");

return 2;

}

if (!hNumberOutputEvent) {

printf("%s", "NumberOutputEvent don't open");

system("pause");

return 2;

}

if (!hEndEnter) {

printf("%s", "EndEnter don't open");

system("pause");

return 2;

}

if (!hEndCProcess) {

printf("%s", "EndCProcess don't open");

system("pause");

return 2;

}

SetEvent(hCProcWaitEvent);

while (Count < 100000) {

WaitForSingleObject(hNumberOutputEvent, INFINITE);

scanf("%d", &OutputNumber);

Count += OutputNumber;

cout << "Current amount = " << Count << endl;

SetEvent(hNextNumberEvent);

}

SetEvent(hEndEnter);

WaitForSingleObject(hEndCProcess, INFINITE);

CloseHandle(hCProcWaitEvent);

CloseHandle(hNextNumberEvent);

CloseHandle(hNumberOutputEvent);

CloseHandle(hEndEnter);

CloseHandle(hEndCProcess);

return 0;

}

ЗАКЛЮЧЕНИЕ

В ходе лабораторной работы мы ознакомились с механизмом межпроцессного обмена данными в Windows, приобрели навыки использования различных средств обмена данными при разработке программ, разработали программу, использующую средства обмена данными.