ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ПЕРМСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Математическое моделирование систем и процессов»

**ЛАБОРАТОРНАЯ РАБОТА №1**

по дисциплине: «Системное и прикладное программное обеспечение»

Выполнил:

студент 3 курса группы ММ-19-2б:

Демид Мельников

Проверил асс. каф. ММСП ПНИПУ:

Р.М. Герасимов

Пермь 2021

**Задание**

Задание:

1. Требуется написать программу, создающую процесс textfile.txt. Также необходимо предусмотреть:

a. Запуск процесса при помощи программ notepad.exe, winword.exe, программы, заданной в командной строке – по выбору пользователя.

b. Вывод описателя процесса и его идентификатора.

c. Вывод идентификатора первичного потока данного процесса.

d. Безаварийное завершение процесса (различными известными способами).

2. Программа также должна выводить на экран информацию об операционной системе (при помощи вызова функций WinAPI VOID GetSystemInfo(LPSYSTEM\_INFO psinf) и BOOL GetVersionEx(POSVERSIONINFO pVersionInformation), см. справку):

a. тип архитектуры процессора;

b. версия процессора;

c. количество процессоров (ядер);

d. минимальный и максимальный адрес доступного адресного пространства;

e. основной и дополнительный номера версии ОС;

f. версия сборки ОС.

3. Написать программы двух консольных процессов Parent и Child, которые выполняют следующие действия.

Процесс Parent:

1. Создает файл, записи которого имеют следующую структуру:

struct emp

{

int num; // номер зачетки

char name[10]; // имя студента

double grade; // средний бал

};

Имя файла и данные о студентах вводятся с консоли.

2. Выводит созданный файл на консоль.

3. Запрашивает с консоли номер зачетки, имя студента и новый средний бал этого студента.

4. Формирует командную строку, которая содержит имя созданного файла и информацию, полученную в пункте 3.

5. Запускает дочерний процесс Child, которому как параметр передается командная строка, сформированная в пункте 4.

6. Ждет завершения работы процесса Child.

7. Выводит откорректированный файл на консоль.

8. Завершает свою работу.

Процесс Child:

a. Выводит на консоль информацию, полученную через командную строку.

b. Корректирует в файле, созданном процессом Parent, нужную запись, т.е. устанавливает новый средний бал студента.

c. Завершает свою работу.

**Реализация**

1. **Main**
2. const char\* WindowsVersion()
3. {
5. if (!IsWindows10OrGreater())
6. {
7. return "10";
8. }
9. else if (!IsWindows8OrGreater())
10. {
11. return "8";
12. }
13. else if (!IsWindows7OrGreater())
14. {
15. return "7";
16. }
17. else if (!IsWindowsVistaOrGreater())
18. {
19. return "Vista";
20. }
21. else if (!IsWindowsXPOrGreater())
22. {
23. return "XP";
24. }
25. else return "Unknown";
26. }
27. int main()
28. {
29. int n;
30. cout << "Enter the starting number: 0 - notepad , 1 - Word , any other number - your own .exe"<<endl;
31. cin >> n;
32. LPWSTR WStrExeLoc;
33. wstring StrExeLoc;
34. if (n == 0)
35. {
36. StrExeLoc = L"c:\\Windows\\notepad.exe textfile";
38. }
39. else if (n == 1)
40. {
41. StrExeLoc = L"c:\\Program Files\\Microsoft Office\\root\\Office16\\WINWORD.exe textfile";
43. }
44. else
45. {
46. wcin >> StrExeLoc;
47. }
48. WStrExeLoc = (LPWSTR)StrExeLoc.c\_str();

51. STARTUPINFO si;//структура,определяет размеры окна процесса и т.п.
52. PROCESS\_INFORMATION pi;//структура информация о процессе
53. ZeroMemory(&si, sizeof(si));// тоже самое, что и ={0}
54. ZeroMemory(&pi, sizeof(pi));//заполняем структуры нулями, т.к. их параметры нам не нужны
55. CreateProcess
56. (
57. NULL,// имя исполняемого модуля
58. WStrExeLoc,// Командная строка
59. NULL, // Указатель на структуру SECURITY\_ATTRIBUTES
60. NULL, // Указатель на структуру SECURITY\_ATTRIBUTES
61. FALSE, // Флаг наследования текущего процесса
62. NULL,// Флаги способов создания процесса
63. NULL,// Указатель на блок среды
64. NULL, // Текущий диск или каталог
65. &si, // Указатель нас структуру STARTUPINFO
66. &pi // Указатель нас структуру PROCESS\_INFORMATION
67. );
69. cout << "Handle: " << pi.hProcess << endl;
70. cout << "Process ID: " << pi.dwProcessId << endl;
71. cout << "Thread ID: " << pi.dwThreadId << endl;
72. TerminateProcess(pi.hProcess,0);
73. system("pause");
74. SYSTEM\_INFO SystInf;
75. GetSystemInfo(&SystInf);
76. cout << "\t Hardware Info: \n"
77. << "Processor Type: " << SystInf.wProcessorArchitecture << endl
78. << "Number Of Processor: " << SystInf.wProcessorRevision << endl
79. << "Number Of Cores: " << SystInf.dwNumberOfProcessors << endl
80. << "Minimum application address: " << SystInf.lpMinimumApplicationAddress << endl
81. << "Maximum Application address: " << SystInf.lpMaximumApplicationAddress << endl;
82. cout << "Windows Version: " << WindowsVersion() << endl;
83. system("pause");
84. STARTUPINFO si1;
85. PROCESS\_INFORMATION pi1;
86. ZeroMemory(&si1, sizeof(si1));
87. ZeroMemory(&pi1, sizeof(pi1));
88. si1.cb = sizeof(si1);
90. wstring S = L"c:\\Users\\Коптутер\\source\\repos\\Parent\\Debug\\Parent.exe";
91. LPWSTR STR = (LPWSTR)S.c\_str();
92. if (CreateProcess
93. (
94. NULL,
95. STR,
96. NULL,
97. NULL,
98. TRUE,
99. NULL,
100. NULL,
101. NULL,
102. &si1,
103. &pi1
104. ))
105. {
106. WaitForSingleObject(pi1.hProcess, INFINITE);
107. CloseHandle(pi1.hThread);
108. CloseHandle(pi1.hProcess);
109. }
111. return 0;
112. }

**2.Parent**

struct emp

{

int num;

char name[10];

double grade;

};

auto GetWstring = [](const string str)->wstring

{

const char\* c = str.c\_str();

const size\_t cSize = str.size() + 1;

wchar\_t\* wc = new wchar\_t[cSize];

mbstowcs(wc, c, cSize);

wstring w\_string(wc);

delete[] wc;

return w\_string;

};

int main()

{

string NotxtName;

cout << "Enter The File Name (without .txt):\n";

cin >> NotxtName;

string Name = NotxtName +".txt";

int m;

cout << "Enter The Number Of The Students: ";

cin >> m;

cout << "\t Enter The Information About The Students:" << endl<<endl;

ofstream out;

out.open(Name);

string str = Name;

str.append(" ");

out << "Total Number Of The Students: " << m << endl << endl << '\t'

<< "The Information About The Students : " << endl;

for (int i = 1; i <= m; i++)

{

emp NewStudent;

cout << "Student Number " << i << ": " << endl

<< "Enter The Student's Book Number: ";

cin >> NewStudent.num;

out << "Student Number " << i << ": " << endl

<< "Student's Book Number: " << NewStudent.num << endl;

str.append(to\_string(i));

str.append(" ");

str.append(to\_string(NewStudent.num));

str.append(" ");

cout << "Enter The Student's Name: ";

cin >> NewStudent.name;

out << "Student's Name: " << NewStudent.name << endl;

str.append(NewStudent.name);

str.append(" ");

cout << "Enter the Student's Grade On Average: ";

cin >> NewStudent.grade;

out << "Student's Grade On Average: " << NewStudent.grade << endl<<endl;

cout << endl;

str.append(to\_string(NewStudent.grade).erase(1));

if (i < m)

{

str.append(" ");

}

}

cout << endl;

out.close();

ifstream in;

in.open(Name);

string line;

while (getline(in,line))

{

cout << line << endl;

}

cout << endl;

in.close();

wstring Path = L"c:\\Users\\Коптутер\\source\\repos\\Child\\Debug\\Child.exe " + GetWstring(str); ;

LPWSTR UltPath = (LPWSTR)Path.c\_str();

STARTUPINFO si2;

PROCESS\_INFORMATION pi2;

ZeroMemory(&si2, sizeof(si2));

ZeroMemory(&pi2, sizeof(pi2));

if (CreateProcess

(

NULL,

UltPath,

NULL,

NULL,

NULL,

NULL,

NULL,

NULL,

&si2,

&pi2

))

{

WaitForSingleObject(pi2.hProcess, INFINITE);

CloseHandle(pi2.hThread);

CloseHandle(pi2.hProcess);

}

system("pause");

ifstream last;

last.open(Name);

string line2;

while (getline(last, line2))

{

cout << line2 << endl;

}

cout << endl;

return 0;

}

**3. Child**

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

{

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

{

cout << "[" << i << "] ";

cout << argv[i] << endl;

}

system("pause");

string Name = argv[1];

string Open = "C:\\Users\\Коптутер\\source\\repos\\sppo\_1\\sppo\_1\\" + Name;

string line;

ifstream in;

ofstream of;

in.open(Open);

of.open("C:\\Users\\Коптутер\\source\\repos\\sppo\_1\\sppo\_1\\tempor.txt");

int j = 3;

while (getline(in,line))

{

if (line.empty() == FALSE)

{

string linesup = line;

if( linesup.erase(linesup.size() - 1) == "Student's Grade On Average: ")

{

int i = stoi(argv[j]);

double NewGrade;

cout << "New Grade for Student Number " << i << ": ";

cin >> NewGrade;

of << "Student's Grade On Average: " << NewGrade<<endl;

j = j + 4;

}

else

{

of << line << endl;

}

}

else

{

of << line << endl;

}

}

const char\* D = Name.c\_str();

in.close();

rename(D,"DELETE.txt");

remove("DELETE.txt");

of.close();

rename("tempor.txt", D);

return 0;

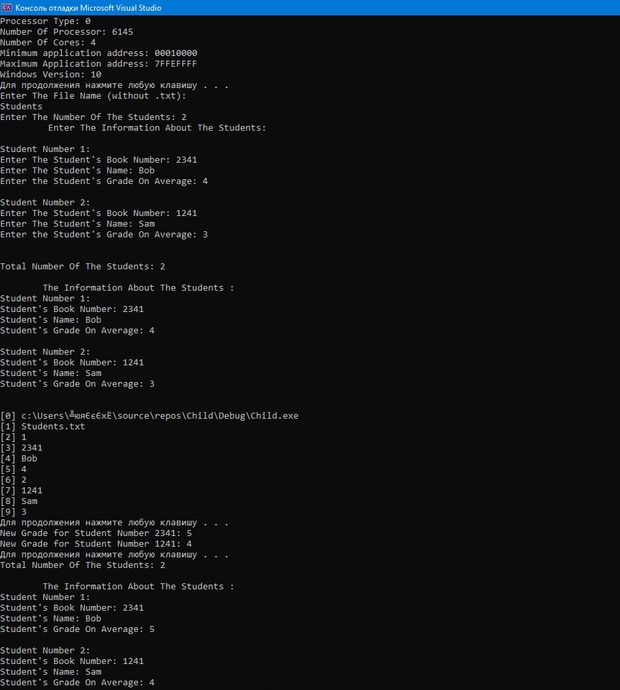
}

**Результаты**

**1.**

****

**2.**

****