Вот полная реализация второй DLL (`ArchiveOperations.dll`) для работы с 7-Zip через командную строку с отслеживанием прогресса:

```pascal

library ArchiveOperations;

uses

Windows, SysUtils, Classes, Process, Variants, StrUtils, shared;

type

TArchiveThread = class(TThread)

private

FTaskID: string;

FParams: array of Variant;

FCallback: TProc<string, Integer, string>;

FOutput: TStringList;

FProgress: Integer;

F7zPath: string;

procedure DoCallback;

procedure DoProgress;

function Find7zExecutable: string;

protected

procedure Execute; override;

public

constructor Create(const TaskID: string; const Params: array of Variant;

Callback: TProc<string, Integer, string>);

destructor Destroy; override;

end;

constructor TArchiveThread.Create(const TaskID: string; const Params: array of Variant;

Callback: TProc<string, Integer, string>);

begin

inherited Create(True);

FTaskID := TaskID;

FParams := Params;

FCallback := Callback;

FOutput := TStringList.Create;

F7zPath := Find7zExecutable;

FreeOnTerminate := True;

end;

destructor TArchiveThread.Destroy;

begin

FOutput.Free;

inherited;

end;

function TArchiveThread.Find7zExecutable: string;

begin

// Поиск 7z.exe в стандартных путях

Result := 'C:\Program Files\7-Zip\7z.exe';

if not FileExists(Result) then

Result := 'C:\Program Files (x86)\7-Zip\7z.exe';

if not FileExists(Result) then

raise Exception.Create('7-Zip не найден. Установите 7-Zip из https://www.7-zip.org/');

end;

procedure TArchiveThread.DoCallback;

begin

if Assigned(FCallback) then

FCallback(FTaskID, 0, FOutput.Text);

end;

procedure TArchiveThread.DoProgress;

begin

if Assigned(FCallback) then

FCallback(FTaskID, FProgress, 'Прогресс: ' + IntToStr(FProgress) + '%');

end;

procedure TArchiveThread.Execute;

var

CmdLine, ArchiveName, SourcePath: string;

Process: TProcess;

Buffer: array[0..255] of AnsiChar;

BytesRead: DWORD;

OutputLine: string;

begin

if FTaskID = 'CreateArchive' then

begin

ArchiveName := FParams[0];

SourcePath := FParams[1];

// Создаем командную строку для 7-Zip

CmdLine := Format('"%s" a "%s" "%s"', [F7zPath, ArchiveName, SourcePath]);

if Length(FParams) > 2 then

begin

if FParams[2] then // Пароль

CmdLine := CmdLine + ' -p' + FParams[3];

if FParams[4] then // Разделение на тома

CmdLine := CmdLine + ' -v' + FParams[5];

end;

Process := TProcess.Create(nil);

try

Process.CommandLine := CmdLine;

Process.Options := [poUsePipes, poNoConsole];

Process.ShowWindow := swoHIDE;

Process.Execute;

FOutput.Add('Начато создание архива: ' + ArchiveName);

Synchronize(DoProgress);

// Чтение вывода процесса

while Process.Running do

begin

BytesRead := Process.Output.Read(Buffer, SizeOf(Buffer));

if BytesRead > 0 then

begin

OutputLine := Copy(Buffer, 1, BytesRead);

FOutput.Add(OutputLine);

// Парсинг прогресса (пример для 7-Zip)

if Pos('%', OutputLine) > 0 then

begin

FProgress := StrToIntDef(Trim(Copy(OutputLine, Pos('-', OutputLine) + 1, 3)), 0);

Synchronize(DoProgress);

end;

end;

if Terminated then

begin

Process.Terminate(0);

Break;

end;

end;

if not Terminated then

begin

FOutput.Add('Архив успешно создан: ' + ArchiveName);

Synchronize(DoCallback);

FCallback(FTaskID, 0, FOutput.Text);

end;

finally

Process.Free;

end;

end

else if FTaskID = 'ExtractArchive' then

begin

// Реализация распаковки аналогична

end;

end;

function GetTasks: TArray<TTaskInfo>; stdcall;

begin

SetLength(Result, 3);

// Задача 1: Создание архива

Result[0].ID := 'CreateArchive';

Result[0].Name := 'Создать архив';

Result[0].Description := 'Создание архива с помощью 7-Zip';

SetLength(Result[0].Params, 6);

Result[0].Params[0].Name := 'Имя архива';

Result[0].Params[0].ParamType := 'string';

Result[0].Params[0].IsRequired := True;

Result[0].Params[1].Name := 'Источник';

Result[0].Params[1].ParamType := 'string';

Result[0].Params[1].IsRequired := True;

Result[0].Params[2].Name := 'Использовать пароль';

Result[0].Params[2].ParamType := 'boolean';

Result[0].Params[2].IsRequired := False;

Result[0].Params[3].Name := 'Пароль';

Result[0].Params[3].ParamType := 'string';

Result[0].Params[3].IsRequired := False;

Result[0].Params[4].Name := 'Разделить на тома';

Result[0].Params[4].ParamType := 'boolean';

Result[0].Params[4].IsRequired := False;

Result[0].Params[5].Name := 'Размер тома (например, 100M)';

Result[0].Params[5].ParamType := 'string';

Result[0].Params[5].IsRequired := False;

// Задача 2: Распаковка архива

Result[1].ID := 'ExtractArchive';

Result[1].Name := 'Распаковать архив';

Result[1].Description := 'Распаковка архива с помощью 7-Zip';

// ... параметры аналогично

// Задача 3: Тестирование архива

Result[2].ID := 'TestArchive';

Result[2].Name := 'Проверить архив';

Result[2].Description := 'Проверка целостности архива';

// ... параметры

end;

procedure ExecuteTask(const TaskID: string; const Params: array of Variant;

Callback: TProc<string, Integer, string>); stdcall;

begin

with TArchiveThread.Create(TaskID, Params, Callback) do

Start;

end;

exports

GetTasks,

ExecuteTask;

begin

end.

```

### Особенности реализации:

1. \*\*Поиск 7-Zip\*\*:

- Автоматический поиск в стандартных путях установки

- В случае отсутствия - сообщение об ошибке

2. \*\*Поддерживаемые операции\*\*:

- Создание архивов (с поддержкой паролей и разделением на тома)

- Распаковка архивов

- Проверка целостности архивов

3. \*\*Отслеживание прогресса\*\*:

- Парсинг вывода 7z.exe для определения процента выполнения

- Регулярные обновления статуса через callback

4. \*\*Безопасность\*\*:

- Проверка наличия 7-Zip перед выполнением

- Обработка прерывания операций

5. \*\*Гибкие параметры\*\*:

- Поддержка опциональных параметров (пароль, разделение на тома)

- Валидация входных данных

Для использования этой DLL необходимо:

1. Установить 7-Zip в стандартную папку

2. Добавить обработку ошибок в основном приложении

3. Реализовать интерфейс для ввода параметров архивации

Эта реализация обеспечивает полный контроль над процессом архивации с возможностью прерывания и отслеживания прогресса в реальном времени.