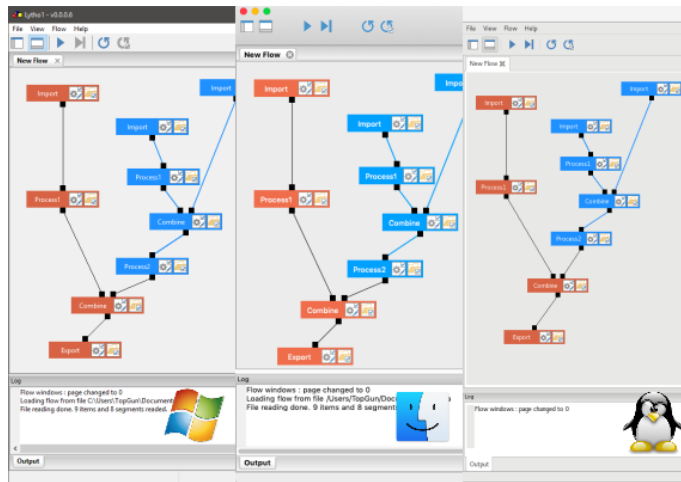




Кроссплатформенная разработка программного обеспечения

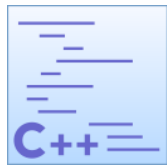
Лабораторная работа №3

Статические и динамические библиотеки



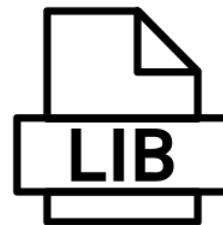
Возможности использования кода в различных проектах

Исходный код



Скомпилированный код

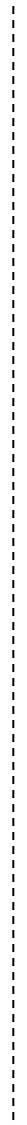
Статическая библиотека



Динамическая библиотека



Обычно нужен заголовочный файл



Создание статической библиотеки: создание проекта

The image displays four sequential screenshots of the Visual Studio IDE interface, illustrating the process of creating a new static library project.

- Screenshot 1: Create a new project**
The 'Create a new project' dialog is shown. Under 'Recent project templates', 'Windows Desktop Wizard' is selected. The filters are set to 'C++', 'Windows', and 'Console'.
- Screenshot 2: Configure your new project**
The 'Configure your new project' dialog for 'Windows Desktop Wizard' is shown. The 'Project name' is 'test_StaticLibrary' and the 'Location' is 'C:\Users\Дмитрий Булах\source\repos'. The 'Solution name' is also 'test_StaticLibrary'. The checkbox 'Place solution and project in the same directory' is unchecked.
- Screenshot 3: Configure your new project**
The 'Configure your new project' dialog is shown with the 'Application type' dropdown set to 'Static Library (.lib)'. The 'Additional options' section is expanded, and the 'Empty project' checkbox is checked.
- Screenshot 4: Configure your new project**
The 'Configure your new project' dialog is shown with the 'Application type' dropdown set to 'Static Library (.lib)'. The 'Additional options' section is expanded, and the 'Empty project' checkbox is checked. The 'Back' and 'Create' buttons are visible at the bottom.

Создание статической библиотеки: код библиотеки

Файл main.cpp:

```
#include <iostream>
#include <locale>

void print_message() {
    setlocale(LC_CTYPE, "rus");
    std::cout << "Привет из статической библиотеки!" << std::endl;
}
```

Файл header.hpp:

```
#pragma once

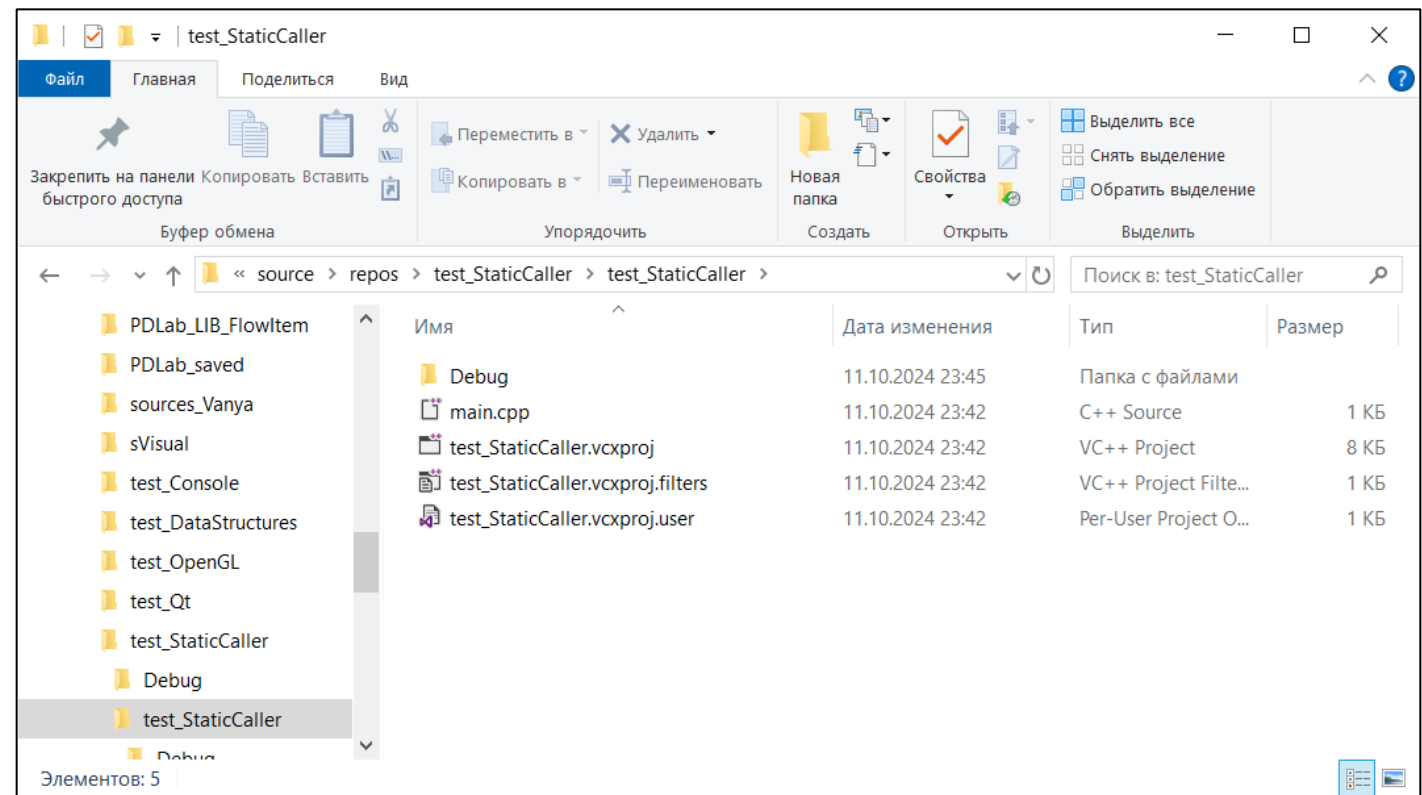
void print_message();
```

Использование статической библиотеки: написание кода

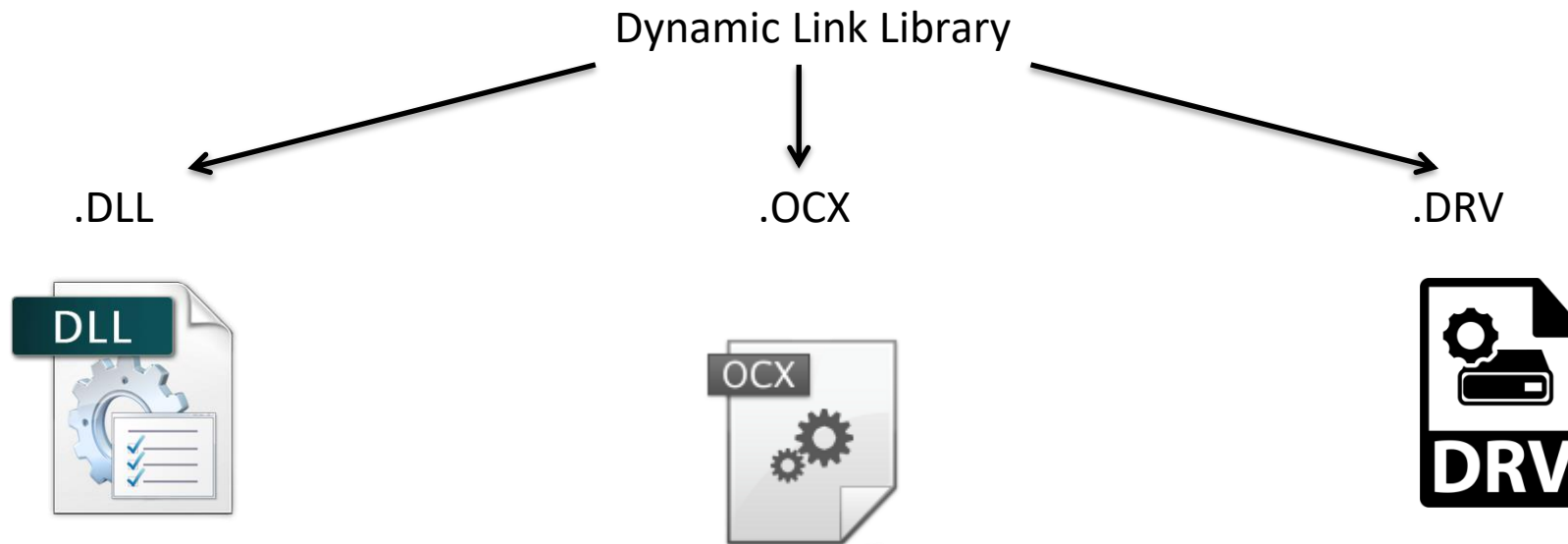
Код:

```
#include <iostream>
#include <Header.hpp>

int main(int argc, char *argv[]) {
    print_message();
    return EXIT_SUCCESS;
}
```



Динамически подключаемые библиотеки





Создание динамической библиотеки: создание проекта

The image shows a sequence of three overlapping dialog boxes in Visual Studio:

- Windows Desktop Wizard:** The main wizard window. The 'Project name' field contains 'test_DLL'. The 'Location' field contains 'C:\Users\Дмитрий Булах\source\repos'. The 'Solution' section has 'Create new solution' selected. The 'Solution name' field contains 'test_DLL'. There is an unchecked checkbox for 'Place solution and project in the same location'.
- Windows Desktop Project (first instance):** A dialog box titled 'Windows Desktop Project'. The 'Application type' dropdown menu is open, showing options: 'Console Application (.exe)', 'Desktop Application (.exe)', 'Dynamic Link Library (.dll)' (which is highlighted), and 'Static Library (.lib)'. Below the dropdown are two unchecked checkboxes: 'Export symbols' and 'MFC headers'. 'OK' and 'Cancel' buttons are at the bottom.
- Windows Desktop Project (second instance):** A dialog box titled 'Windows Desktop Project'. The 'Application type' dropdown menu is set to 'Dynamic Link Library (.dll)'. Under 'Additional options', the 'Empty project' checkbox is checked, while 'Precompiled header', 'Export symbols', and 'MFC headers' are unchecked. 'OK' and 'Cancel' buttons are at the bottom.



Создание динамической библиотеки: код библиотеки

Файл main.cpp:

```
#include <iostream>
#include <locale>

__declspec(dllexport) void print_message() {
    setlocale(LC_STYPE, "rus");
    std::cout << "Привет из динамической библиотеки!" << std::endl;
}
```

Лог компилятора:

```
1>----- Build started: Project: test_DLL, Configuration: Debug Win32 -----
1>main.cpp
1>test_DLL.vcxproj -> C:\Users\...\test_DLL\Debug\test_DLL.dll
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====
```




Использование динамической библиотеки: код

```
#include <Windows.h>
#include <iostream>
#include <locale>

typedef void(*func_ptr)();

int main(int argc, char *argv[]) {
    setlocale(LC_CTYPE, "rus");
    HMODULE hdll = LoadLibrary(L"test_DLL.dll");
    if (!hdll) {
        std::cout << "Не могу загрузить файл DLL :(" << std::endl;
        return EXIT_FAILURE;
    }

    func_ptr func = (func_ptr)GetProcAddress(hdll, "print_message");
    if (func != NULL)
        func();

    FreeLibrary(hdll);

    return EXIT_SUCCESS;
}
```



Динамическая библиотека: функция DllMain (1)

```
BOOL __stdcall DllMain(HANDLE hInst,  
                      DWORD dwReason,  
                      LPVOID lpReserved) {  
    BOOL bAllWentWell = TRUE;  
    switch (dwReason) {  
        case DLL_PROCESS_ATTACH: // Инициализация процесса.  
            break;  
        case DLL_THREAD_ATTACH: // Инициализация потока.  
            break;  
        case DLL_THREAD_DETACH: // Очистка структур потока.  
            break;  
        case DLL_PROCESS_DETACH: // Очистка структур процесса.  
            break;  
    }  
    return bAllWentWell;  
}
```

Динамическая библиотека: функция DllMain (2)

```
#include <iostream>
#include <locale>
#include <Windows.h>

__declspec(dllexport) void print_message() {
    setlocale(LC_CTYPE, "rus");
    std::cout << "Привет из динамической библиотеки!" << std::endl;
}

BOOL __stdcall DllMain(HANDLE hInst, DWORD dwReason, LPVOID IpReserved) {
    BOOL bAllWentWell = TRUE;
    switch (dwReason) {
        case DLL_PROCESS_ATTACH: // Инициализация процесса.
            std::cout << "DLL_PROCESS_ATTACH" << std::endl;
            break;
        case DLL_THREAD_ATTACH: // Инициализация потока.
            std::cout << "DLL_THREAD_ATTACH" << std::endl;
            break;
        case DLL_THREAD_DETACH: // Очистка структур потока.
            std::cout << "DLL_THREAD_DETACH" << std::endl;
            break;
        case DLL_PROCESS_DETACH: // Очистка структур процесса.
            std::cout << "DLL_PROCESS_DETACH" << std::endl;
            break;
    }
    return bAllWentWell;
}
```

Работа со статическими библиотеками в ОС Linux: код библиотеки

Код библиотеки (library.cpp):



static

```
#include <iostream>

extern "C" void print_message() {
    std::cout << "Hello from the .a file!" << std::endl;
}
```

Компиляция библиотеки:

```
g++ -c -o library.o library.cpp
```

```
ar rcs liblibrary.a library.o
```

Работа со статическими библиотеками в ОС Linux: вызывающий код

Код библиотеки (main.cpp):

```
#include <iostream>
#include <library.hpp>

int main(int argc, char *argv[]) {
    print_message();
    return EXIT_SUCCESS;
}
```

Компиляция программы:

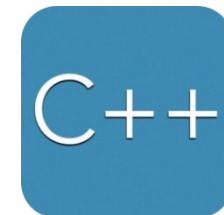
```
g++ main.cpp
-I/home/student/KRPO_Lab0x03/static/library
-L/home/student/KRPO_Lab0x03/static/library -llibrary
-o main
```

Работа с динамическими библиотеками в ОС Linux: код библиотеки

Код библиотеки (library.cpp):

```
#include <iostream>

extern "C" void print_message() {
    std::cout << "Hello from the .a file!" << std::endl;
}
```



dynamic

Работа с динамическими библиотеками в ОС Linux: вызывающий код

Код вызывающей программы (main.cpp):

```
#include <dlfcn.h>
#include <iostream>

typedef void(*func_ptr)();

int main(int argc, char *argv[]) {
    void *handle = dlopen("./library.so", RTLD_LAZY);
    if (!handle) {
        std::cout << "Can't load shared object" << std::endl;
        return 1;
    }
    func_ptr hello = (func_ptr)dlsym(handle, "print_message");
    if (!hello)
        ...

    hello();

    dlclose(handle);

    return 0;
}
```

Компиляция динамической библиотеки в Linux

Компиляция динамической библиотеки:

```
g++ -shared -fPIC ./library.cpp -o library.so
```

Компиляция вызывающей программы:

```
g++ ./main.cpp -o main -ldl
```


Отладка проектов в ОС Linux (1)

Запуск отладчика:

```
gdb ./main
```

```
student@localhost:~/KRPO_Lab0x03/gdb
File Edit View Search Terminal Help
student@localhost:~/KRPO_Lab0x03/gdb> gdb ./main
GNU gdb (GDB; SUSE Linux Enterprise 15) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-suse-linux".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://bugs.opensuse.org/>.
Find the GDB manual and other documentation resources online at:
  <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./main...
(gdb) █
```

Отладка проектов в ОС Linux (2)

Запуск программы:

run или r

```
student@localhost:~/KRPO_Lab0x03/gdb
File Edit View Search Terminal Help
68 0 7 0 4196158 0 13 0 0 32767 31 0 0 -2032262634 1852138613 1831821924 9762366
05 808467822 859517232 993211195 825507186 1835216426 976368443 825245044 181850
4746 976302907 808467834 1701064234 976302907 808467816 2049845818 825441072 825
441072 808467762 2054434346 825441072 825245033 1818504746 708457779 859517232 1
734701162 708457779 993079357 1735421230 708457779 993079357 1836085294 70845777
9 993079357 1702309418 892549937 809330275 1630415418 859516976 1029923180 77451
9346 859516976 1030186863 774519346 859516976 1498698835 1279345487 1275098227 1
932420467 1966030146 1699436591 1684483950 1145503800 1230438456 1936028461 1291
866669 1163154265 1397310464 1668246589 795176303 1230982735 1684157028 12293443
35 1769156453 1768697198 1667329381 1634038636 1953509238 1380998980 1448042315
1329791024 1095565426 1814914419 1431127377 1430347592 808529778 1128877641 1970
170220 1476408125 1028802884 1952539693 1701082484 1330924371 1027953231 1414353
487 795176303 858814512 1396793160 1932486771 1836017711 1414750028 1869364564 1
937059645 1598506072 1195661312 1918986355 1752379244 1598379073 1950172233 1146
617956 1634549072 826102351 1397051972 1702065455 859136868 892809265 1145128274
1633893715 1380319333 943543118 1634545516 792549701 1279608915 1397900630 7621
47429 1313818963 1413563731 1953198949 1699950450 1835628605 1633906540 14155333
97 1330205776 1412243577 1414222661 1028214852 1952788285 1028413004 1180649216
1869365309 758723896 1966030152 1752379250 943272765 1347772243 1028801878 18352
96559 1230197573 1685419123 1398096479 792553588 1969695859 1714906672 159850607
2 1702065455 1747926361 1919903860 1768697714 1852141679 1348423247 1886352491 1
936942419 1397901636 1701670760 1869361010 17680
Program received signal SIGSEGV, Segmentation fault.
0x0000000000400860 in print_matrix (matrix=0x7fffffffcd80) at ./main.cpp:8
8         std::cout << matrix[i][j] << " ";
Missing separate debuginfos, use: zypper install libgcc_s1-debuginfo-13.2.1+git7
813-150000.1.6.1.x86_64 libstdc++6-debuginfo-13.2.1+git7813-150000.1.6.1.x86_64
(gdb)
```

Отладка проектов в ОС Linux (3)

Посмотреть исходник:
`list` или `l`

```
student@localhost:~/KRPO_Lab0x03/gdb
File Edit View Search Terminal Help
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./main...
(gdb) list
1      #include <iostream>
2
3      #define MATRIX_SIZE 5
4
5      void print_matrix(int matrix[MATRIX_SIZE][MATRIX_SIZE]) {
6          for(int i = 0; i < MATRIX_SIZE; ++i) {
7              for(int j = 0; j < MATRIX_SIZE; ++j)
8                  std::cout << matrix[i][j] << " ";
9                  std::cout << std::endl;
10         }
(gdb) list 10
5      void print_matrix(int matrix[MATRIX_SIZE][MATRIX_SIZE]) {
6          for(int i = 0; i < MATRIX_SIZE; ++i) {
7              for(int j = 0; j < MATRIX_SIZE; ++j)
8                  std::cout << matrix[i][j] << " ";
9                  std::cout << std::endl;
10         }
11     }
12
13     int main() {
14         int matrix[MATRIX_SIZE][MATRIX_SIZE] = {
(gdb)
```

Отладка проектов в ОС Linux (4)

Установка breakpoint:

`break <N>`

или

`break <name>`

```
student@localhost:~/KRPO_Lab0x03/gdb
File Edit View Search Terminal Help
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./main...
(gdb) list
1      #include <iostream>
2
3      #define MATRIX_SIZE 5
4
5      void print_matrix(int matrix[MATRIX_SIZE][MATRIX_SIZE]) {
6          for(int i = 0; i < MATRIX_SIZE; ++i) {
7              for(int j = 0; j < MATRIX_SIZE; ++i)
8                  std::cout << matrix[i][j] << " ";
9                  std::cout << std::endl;
10         }
(gdb) list 10
5      void print_matrix(int matrix[MATRIX_SIZE][MATRIX_SIZE]) {
6          for(int i = 0; i < MATRIX_SIZE; ++i) {
7              for(int j = 0; j < MATRIX_SIZE; ++i)
8                  std::cout << matrix[i][j] << " ";
9                  std::cout << std::endl;
10         }
11     }
12
13     int main() {
14         int matrix[MATRIX_SIZE][MATRIX_SIZE] = {
(gdb) break 8
Breakpoint 1 at 0x40083d: file ./main.cpp, line 8.
(gdb) █
```



Отладка проектов в ОС Linux (5)

Запуск отладчика:

```
gdb ./main
```

Шагнуть (step over): next

Запуск программы:

```
run или r
```

Провалиться (step into): step

Посмотреть исходник:

```
list или l
```

Получение информации: info

```
info breakpoints
```

```
info locals
```

```
info functions
```

Установка breakpoint:

```
break <N>
```

или

```
break <name>
```

Получение значения переменной: print

```
print i
```