Технологии программирования

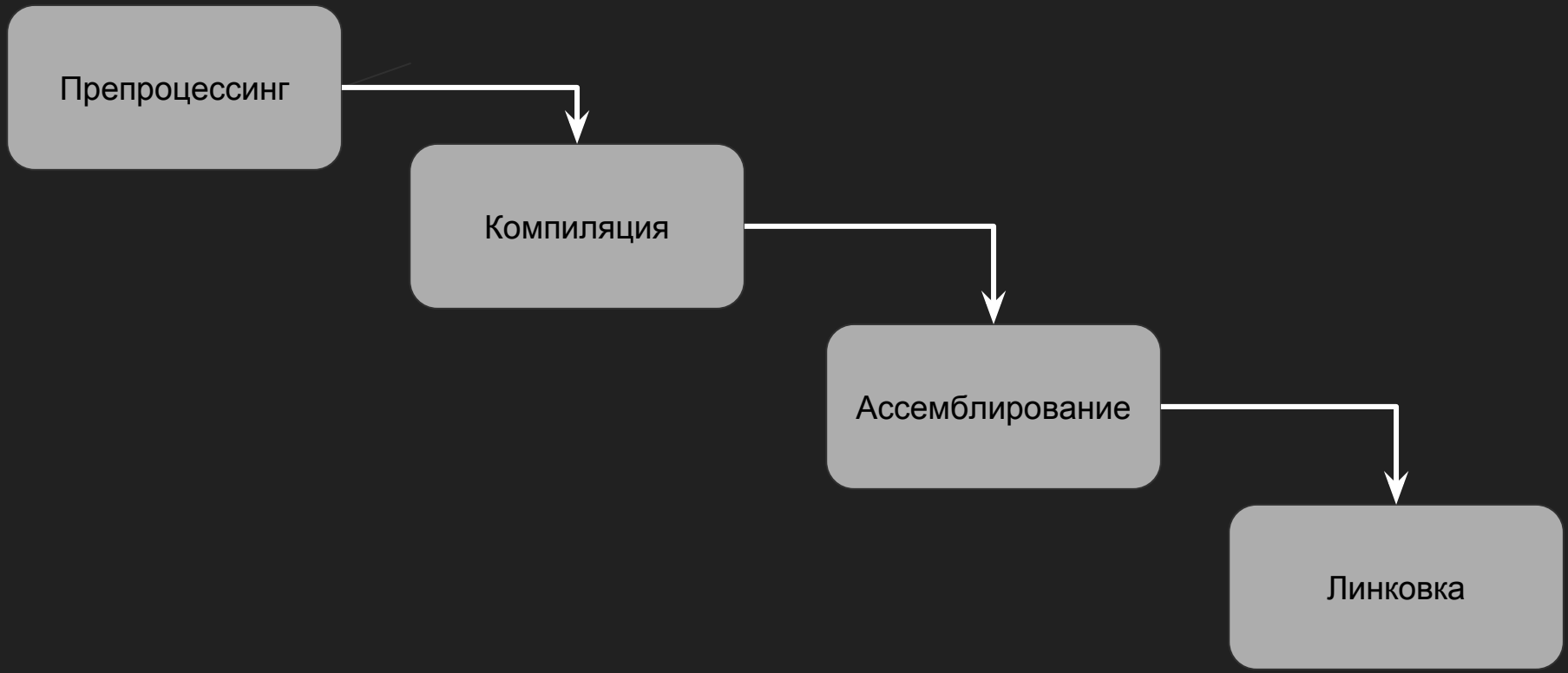
Сборка. Линковка. Отладка. Методы отладки

Александр Михалевич

Замечания к дз

1. Задание по CMake не исправляется
2. Делать отдельный main для запуска тестов - плохо
3. Не заливать бинари и gtest в репо
4. Codestyle, чистота кода
5. Builder для иммутабельных объектов

Сборка



Рассмотрим пример...

```
#include <iostream>

int main(int argc, char** argv) {
	std::cout << "Hello world" << std::endl;
	return 0;
}
```

Препроцессинг

```
$ g++ -E -o example.i example.cpp
$ cat example.i
...
namespace std __attribute__ ((__visibility__ ("default")))
{
# 60 "/usr/lib/gcc/x86_64-pc-linux-gnu/6.4.0/include/g++-v6/iostream" 3
  extern istream cin;
  extern ostream cout;
  extern ostream cerr;
  extern ostream clog;
  extern wistream wcin;
  extern wostream wcout;
  extern wostream wcerr;
  extern wostream wclog;
  static ios_base::Init __ioinit;
}
# 2 "example.cpp" 2
# 3 "example.cpp"
int main(int argc, char** argv) {
 std::cout << "Hello world" << std::endl;
 return 0;
}
```

Препроцессинг

```
$ ls -lh
-rw-r--r-- 1 alexander alexander 110      Mar  4 16:07 example.cpp
-rw-r--r-- 1 alexander alexander 657K     Mar  4 16:12 example.i
```

~ 6116 раз

Компиляция

```
$ g++ -S -o example.s example.cpp
$ cat example.s
...
main:
.LFB1461:
        .cfi_startproc
        pushq  %rbp
        .cfi_def_cfa_offset 16
        .cfi_offset 6, -16
        movq   %rsp, %rbp
        .cfi_def_cfa_register 6
        subq   $16, %rsp
        movl   %edi, -4(%rbp)
        movq   %rsi, -16(%rbp)
        movl   $.LC0, %esi
        movl   $_ZSt4cout, %edi
        call
_ZStlsISt11char_traitsIcEERSt13basic_ostreamIcT_ES5_PKc
        movl
$_ZSt4endlIcSt11char_traitsIcEERSt13basic_ostreamIT_T0_ES6_, %esi
        movq   %rax, %rdi
        call      _ZNSolsEPFRSoS_E
...
```

Компиляция

```
$ ls -lh
-rw-r--r-- 1 alexander alexander 110   Mar  4 16:07 example.cpp
-rw-r--r-- 1 alexander alexander 657K  Mar  4 16:12 example.i
-rw-r--r-- 1 alexander alexander 2.0K  Mar  4 16:19 example.s
```

Ассемблирование

```
$ g++ -c -o example.o example.cpp
$ objdump -d example.o
0000000000000000 <main>:
   0:   55                      push   %rbp
   1:   48 89 e5                mov    %rsp,%rbp
   4:   48 83 ec 10             sub    $0x10,%rsp
   8:   89 7d fc                mov    %edi,-0x4(%rbp)
   b:   48 89 75 f0             mov    %rsi,-0x10(%rbp)
   f:   be 00 00 00 00          mov    $0x0,%esi
  14:   bf 00 00 00 00          mov    $0x0,%edi
  19:   e8 00 00 00 00          callq  1e <main+0x1e>
  1e:   be 00 00 00 00          mov    $0x0,%esi
  23:   48 89 c7                mov    %rax,%rdi
  26:   e8 00 00 00 00          callq  2b <main+0x2b>
  2b:   b8 00 00 00 00          mov    $0x0,%eax
  30:   c9                      leaveq
  31:   c3                      retq
...
```

Ассемблирование

```
$ ls -lh
-rw-r--r-- 1 alexander alexander 110  Mar  4 16:07 example.cpp
-rw-r--r-- 1 alexander alexander 657K Mar  4 16:12 example.i
-rw-r--r-- 1 alexander alexander 2.0K Mar  4 16:19 example.s
-rw-r--r-- 1 alexander alexander 2.7K Mar  4 16:25 example.o
```

Линковка. Определение и объявление

Определение
(declaration)

Связывает имя с реализацией

Пример:

```
int my_func(int a) {
        return a+2;
}
```

Объявление
(definition)

Говорит компилятору, что определение функции или переменной существует в другом месте программы

Пример:

```
int my_func(int a);
```

Линковка. Определение и объявление

линковщик выполняет проверку “обещаний”

Линковка. Типичные ошибки. Undefined reference

usr/lib/**libgtest**.a(**gtest-all.cc.o**): In function
`testing::internal::ThreadLocal<testing::TestPartResultReporterInterface*>::~ThreadLocal()':

gtest-all.cc:(.text._ZN7testing8internal11ThreadLocalIPNS_31TestPartResultReporterInterfaceEED2Ev[_ZN7testing8internal11ThreadLocalIPNS_31TestPartResultReporterInterfaceEED5Ev]+0x25): **undefined reference to `pthread_getspecific'**

gtest-all.cc:(.text._ZN7testing8internal11ThreadLocalIPNS_31TestPartResultReporterInterfaceEED2Ev[_ZN7testing8internal11ThreadLocalIPNS_31TestPartResultReporterInterfaceEED5Ev]+0x3a): **undefined reference to `pthread_key_delete'**
...

Линковка. Типичные ошибки. Undefined reference

usr/lib/libgtest.a(gtest-all.cc.o): In function
`testing::internal::ThreadLocal<testing::TestPartResultReporterInterface*>::~ThreadLocal()':

gtest-all.cc:(.text._ZN7testing8internal11ThreadLocalIPNS_31TestPartResultReporterInterfaceEED2Ev[_ZN7testing8internal11ThreadLocalIPNS_31TestPartResultReporterInterfaceEED5Ev]+0x25): undefined reference to `pthread_getspecific'

gtest-all.cc:(.text._ZN7testing8internal11ThreadLocalIPNS_31TestPartResultReporterInterfaceEED2Ev[_ZN7testing8internal11ThreadLocalIPNS_31TestPartResultReporterInterfaceEED5Ev]+0x3a): undefined reference to `pthread_key_delete'
…

$ g++ main.cpp test.cpp -lgtest -lgmock -lpthread

Линковка. Типичные ошибки. Multiple definition

One Definition Rule (ODR)

раздел 6.2 стандарта
http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2017/n4659.pdf

Линковка. Типичные ошибки. Multiple definition

header.hpp

```
#pragma once
int my_func(int a) {
        return a + 2;
}
```

source1.cpp

```
#include “header.hpp”

…
std::cout << my_func(2);
...
```

source2.cpp

```
#include “header.hpp”

…
another_func(my_func(2));
...
```

Линковка. Типичные ошибки. Multiple definition

```
/tmp/ccTzsnq4.o: In function `my_func(int)':
source2.cpp:(.text+0x0): multiple definition of `my_func(int)'
/tmp/ccaq9Ze9.o:source1.cpp:(.text+0x0): first defined here
collect2: error: ld returned 1 exit status
```

Линковка. Типичные ошибки. Multiple definition. Стандартный способ исправления.

header.h
```
#pragma once
int my_func(int a);
```

impl.cpp
```
#include “header.h”
int my_func(int a) {
        return a + 2;
}
```

source1.cpp
```
#include “header.h”

…
std::cout << my_func(2);
...
```

source2.cpp
```
#include “header.h”

…
another_func(my_func(2));
...
```

Линковка. Типичные ошибки. Multiple definition. Второй способ исправления: inline functions

header.hpp

```
#pragma once
inline int my_func(int a) {
        return a + 2;
}
```

source1.cpp

```
#include “header.hpp”

…
std::cout << my_func(2);
...
```

source2.cpp

```
#include “header.hpp”

…
another_func(my_func(2));
...
```

Линковка. Типичные ошибки. Multiple definition. Третий способ исправления: грязный хак

header.hpp

```
#pragma once
template<class T>
int my_func(int a) {
        return a + 2;
}
```

source1.cpp

```
#include “header.hpp”

…
std::cout << my_func<void>(2);
...
```

source2.cpp

```
#include “header.hpp”

…
another_func(my_func<void>(2));
...
```

Линковка в CMake

```
# проверка версии
cmake_minimum_required(VERSION 2.8)
# название проекта
project(bot)
# установим флаги компилятора
set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++11
-Wall”)
# перечислим все исходники
set(SOURCES_LIST “hello.cpp bye.cpp tests.cpp”)
# создать исполняемый файл
add_executable(bot ${SOURCES_LIST})
# подключаем библиотеку gtest
target_link_libraries(bot gtest gmock)
```

Отладка. GDB

```
#include <iostream>
void free(int* ptr) {
	delete ptr;
}
int main() {
	int* myArr = new int[20];
	for (int i = 0; i < 20; ++i)
		myArr[i] = i;
	for(int i = 0; i < 23; ++i)
		std::cout << myArr[i] << " ";
	std::cout << std::endl;
	delete[] myArr;
	free(myArr);
	return 0;
}
```

Отладка. GDB

```
$ g++ -g main.cpp
$ ./a.out
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 0 0 1041
*** Error in `./a.out': double free or corruption (fasttop): 0x0000000001d49c20 ***
======= Backtrace: =========
/lib64/libc.so.6(+0x77adb)[0x7f1b4e4a4adb]
/lib64/libc.so.6(+0x7f73e)[0x7f1b4e4ac73e]
/lib64/libc.so.6(+0x80026)[0x7f1b4e4ad026]
./a.out[0x400893]
./a.out[0x400946]
/lib64/libc.so.6(__libc_start_main+0xf1)[0x7f1b4e44d521]
./a.out[0x4007aa]
======= Memory map: ========
00400000-00401000 r-xp 00000000 103:01 3809786                   /home/alexander/TP/test_bug/a.out
00600000-00601000 r--p 00000000 103:01 3809786                   /home/alexander/TP/test_bug/a.out
```

Отладка. GDB

```
$ gdb -q ./a.out
(gdb) run
…
(gdb) bt
#0  0x00007ffff713af70 in raise () from /lib64/libc.so.6
#1  0x00007ffff713cc3a in abort () from /lib64/libc.so.6
#2  0x00007ffff717dae0 in __libc_message () from /lib64/libc.so.6
#3  0x00007ffff718573e in malloc_printerr () from /lib64/libc.so.6
#4  0x00007ffff7186026 in _int_free () from /lib64/libc.so.6
#5  0x0000000000400893 in free (ptr=0x613c20) at main.cpp:4
#6  0x0000000000400946 in main () at main.cpp:18
```

Отладка. GDB

```
#include <iostream>
void free(int* ptr) {
        delete ptr;
}
int main() {
        int* myArr = new int[20];
        for (int i = 0; i < 20; ++i)
                myArr[i] = i;
        for(int i = 0; i < 23; ++i)
                std::cout << myArr[i] << " ";
        std::cout << std::endl;
        delete[] myArr;
        free(myArr);
        return 0;
}
```

Отладка. GDB

```
$ g++ -g main.cpp
$ ./a.out
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 0 0 1041
```

Отладка. GDB

```
$ gdb -q ./a.out
(gdb) run
…
(gdb) list
…
(gdb) b 12
(gdb) run
...
(gdb) p myArr
$1 = (int *) 0x613c20
(gdb) x/32dw 0x613c20
0x613c20:  0       1       2       3
0x613c30:  4       5       6       7
0x613c40:  8       9       10      11
0x613c50:  12      13      14      15
0x613c60:  16      17      18      19
0x613c70:  0       0       131985      0
0x613c80:  0       0       0       0
0x613c90:  0       0       0       0
```