

Понятие программы

Лекция 1

План лекции

- Информация об этом курсе
- Понятие программы
- Этапы создания программ

Информация об этом курсе

- http://koi.nsu.ru/new/courses/programming_yvu/index.html
- Петров Евгений Сергеевич
- 1й семестр
 - 16 учебных недель
 - 1-2 потоковых контрольных работы
 - Дифференцированный зачёт
- 2й семестр
 - 16 учебных недель
 - 1-2 потоковых контрольных работы
 - Экзамен
- Лекция + семинар + практика каждую учебную неделю

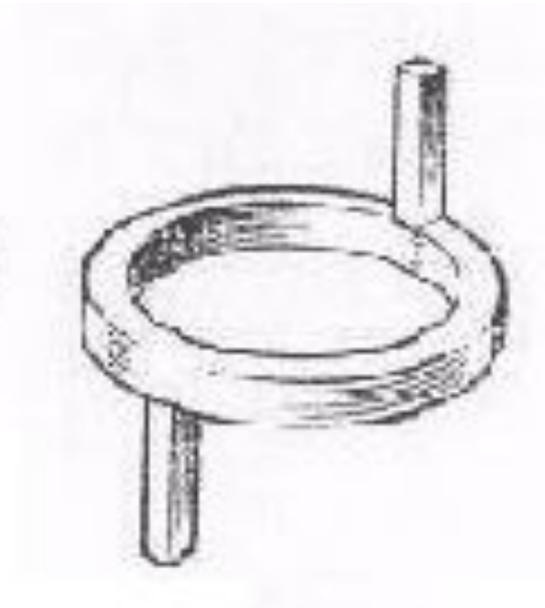
Понятие программы

- Программа – это данные, предназначенные для управления конкретными компонентами системы обработки информации в целях реализации определенного алгоритма. (ГОСТ 19781—90)
- Программа – это представленная в объективной форме совокупность данных и команд, предназначенных для функционирования ЭВМ и других компьютерных устройств с целью получения определённого результата, включая подготовительные материалы, полученные в ходе разработки программы для ЭВМ, и порождаемые ею аудиовизуальные отображения. (ГК РФ)

Понятие программы

- Программа – это размещённые в оперативной памяти компьютера данные и машинные инструкции, исполняемые процессором для достижения некоторой цели.
(Википедия)

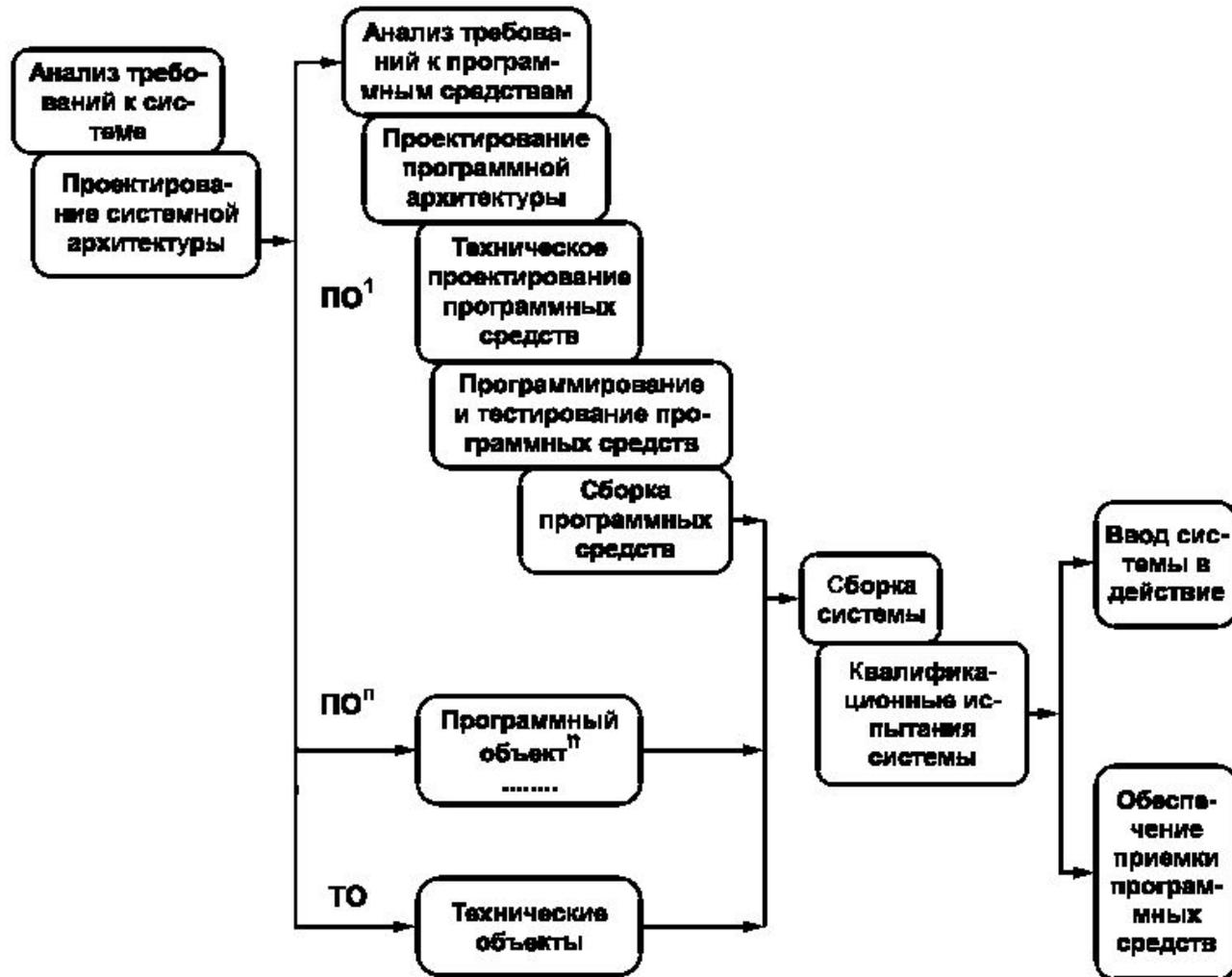
Понятие программы



Этапы создания программ

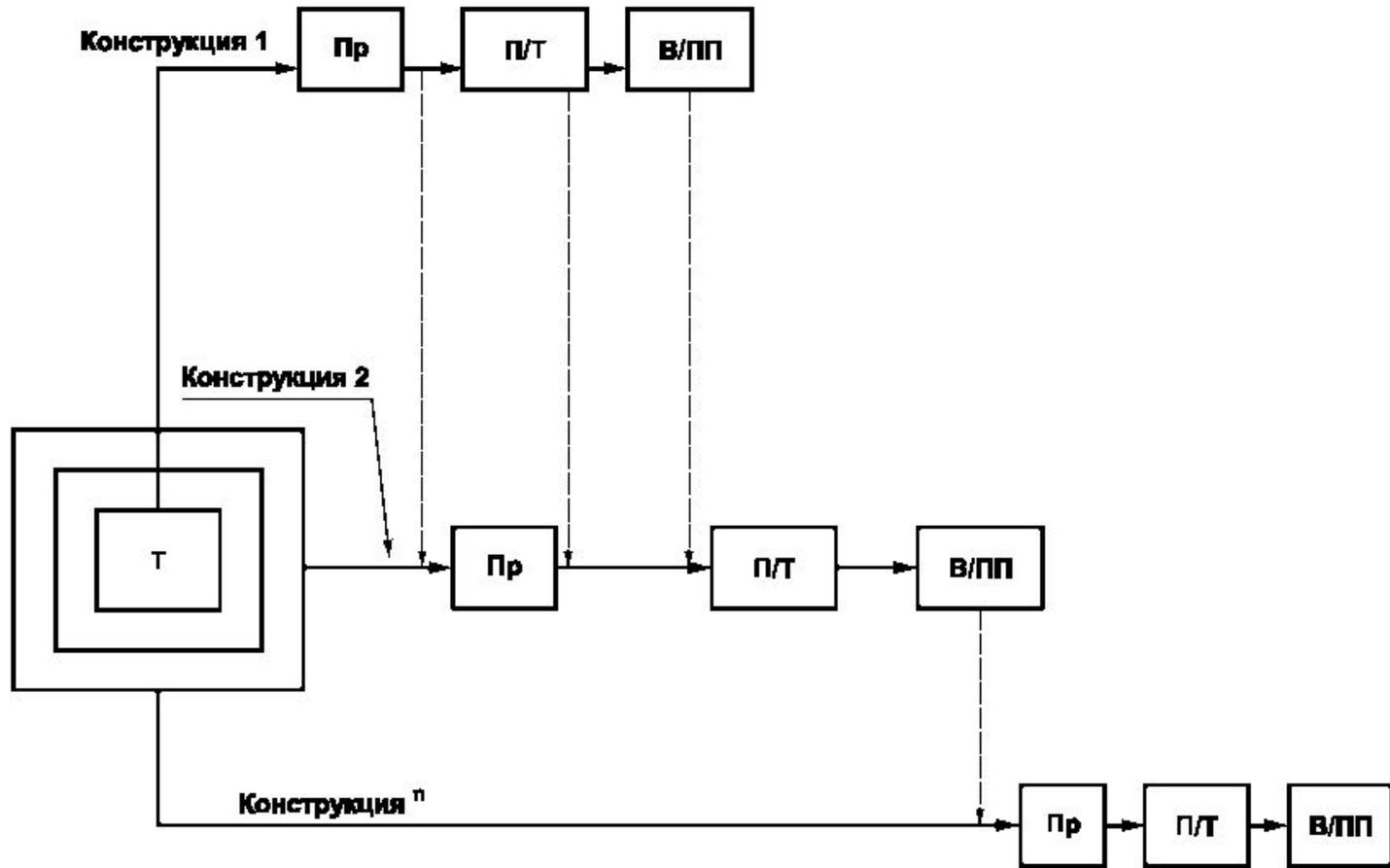
- Накопление требований, работа с заказчиком
- Проектирование – процедурная декомпозиция, ОО, др.
- Внутреннее и внешнее документирование
- Разработка
 - Написание исходного кода
 - Компиляция исходного кода
 - Сборка
 - Отладка
 - Оптимизация
 - Тестирование
- Сдача в эксплуатацию (релиз)
- Сопровождение

Каскадный подход



ПО - программный объект;
ТО - технический объект

Инкрементальный подход



-----> Возможный информационный поток

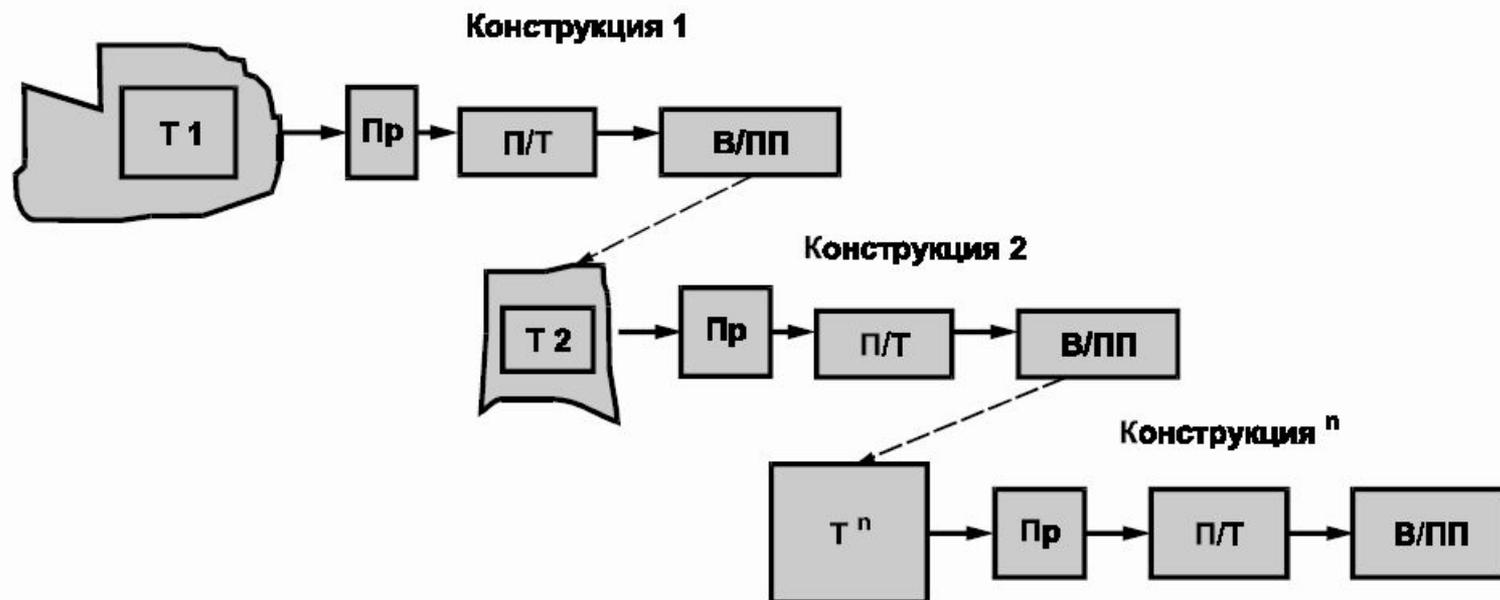
Т - требования;

Пр - проектирование;

П/Т - программирование и тестирование;

В/ПП - ввод в действие и обеспечения приемки

Эволюционный подход



-----> Информационный поток (уточненный)

Т - требования;

Пр - проект;

П/Т - программирование и тестирование;

В/ПП - ввод в действие и поддержка приемки

Этапы создания программ

	Каскад	Инкремент	Эволюция
Создание программы	Один проход	Итерации	Итерации
Размер программы	Ограниченный	Любой	Любой
Изменение функциональности	Нет	Частично	Без ограничений
Качество программы	Высокое	Ограниченное	Ограниченное
Требования к квалификации разработчиков	Низкие	Высокие	Высокие
Примеры технологий	Плановая	Extreme Programming (XP) Feature-Driven Development (FDD) Agile Programming	

Написание исходного кода

- Выбор языка программирования
- Следование стандарту языка
- Следование системе правил
 - Именованние типов, переменных, констант, функций, файлов
 - Деление кода на функции, файлы, компоненты
 - Форматирование и комментирование кода
- Минимальное дублирование кода
- Похожим действия -- похожая запись («устойчивые обороты»)
- Рефакторинг кода (refactoring)
- Ревью кода (review)

```
if (1==x) x=0; else x=1;
x=1-x;
int T[2] = {1, 0}; x = T[x];
x = x?0:1;
x = !x;
if (1==x) x=0;
else if (0==x) x=1;
else assert(x==0 | x==1);
```

Компиляция исходного кода

- Файлы с исходным кодом называются единицами компиляции
- Результатом компиляции является файл с объектным кодом
- Если изменения в файле Ф1 *могут* нарушить логику работы кода в файле Ф2, то Ф2 зависит от Ф1
- Системы компиляции умеют автоматически учитывать *некоторые* зависимости между файлами
 - GNU make, MS nmake, scons, ...
- За учёт *всех* зависимостей отвечает программист

Компиляция исходного кода

`worker.h`

```
void do_some_work();
```

`worker.c`

```
#include "worker.h"  
void do_some_work() { /* ... */ }
```

`main.c`

```
#include "worker.h"  
int main()  
{  
    do_some_work();  
    return 0;  
}
```

Для чего нужна
строка
`#include "worker.h"`
в файле `worker.c`?

Изменения в `worker.c` требуют
изменений в `worker.h` и
перекомпиляции `worker.c` и `main.c`

Сборка (линковка)

- Различают три вида сборки
 - Сборка статической библиотеки
 - Сборка динамической библиотеки
 - Сборка исполняемого файла

Сборка статической библиотеки

- Вход: объектные файлы
- Выход: архив, содержащий эти файлы
- Статическая библиотека – средство группирования логически связанных объектных файлов

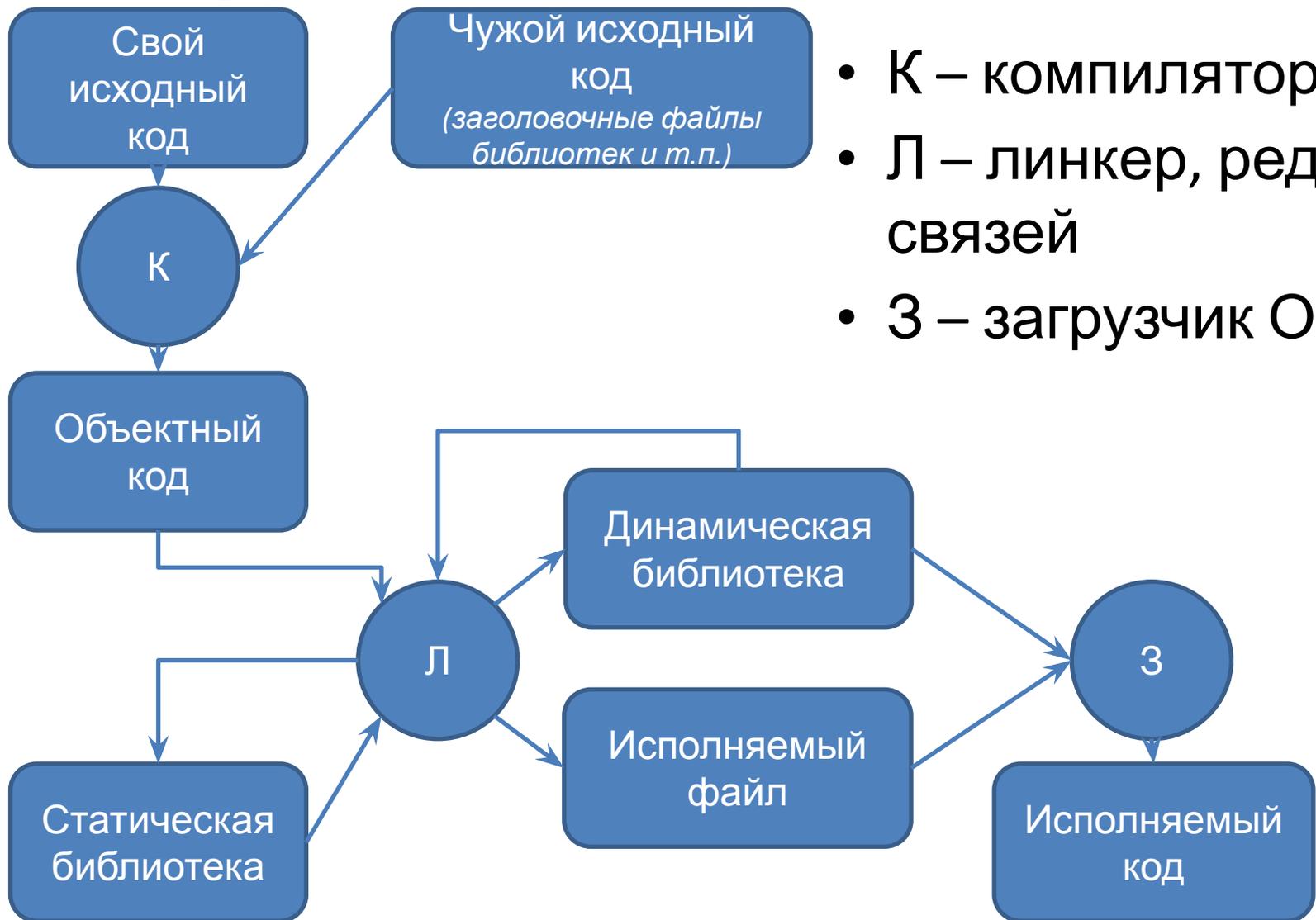
Сборка динамической библиотеки

- Вход: объектные файлы, статические библиотеки, ранее созданные динамические библиотеки
- Выход: файл со служебной информацией для ОС и машинными инструкциями, годными для исполнения процессором
- Динамическая библиотека – средство построения программ в процессе их работы

Сборка исполняемого файла

- Вход: объектные файлы, статические библиотеки, динамические библиотеки
- Выход: файл со служебной информацией для ОС, машинными инструкциями, годными для исполнения процессором, и «точкой входа»

Компиляция, сборка, загрузка в память для исполнения



- К – компилятор
- Л – линкер, редактор связей
- З – загрузчик ОС

Сборка (линковка)

```
c:\Users\espetrov>cl -c worker.c
```

```
c:\Users\espetrov>dumpbin /all /disasm worker.obj
```

<p>Microsoft (R) COFF/PE Dumper Version 9.00.21022.08 Copyright (C) Microsoft Corporation. All rights reserved.</p> <p>Dump of file worker.obj</p> <p>File Type: COFF OBJECT</p> <p>FILE HEADER VALUES 14C machine (x86) 3 number of sections 5048208D time date stamp Thu Sep 06 11:03:25 2012 130 file pointer to symbol table 9 number of symbols 0 size of optional header 0 characteristics</p> <p>SECTION HEADER #1 .drectve name 0 physical address 0 virtual address 2F size of raw data 8C file pointer to raw data (0000008C to 000000BA) 0 file pointer to relocation table 0 file pointer to line numbers 0 number of relocations 0 number of line numbers 100A00 flags Info Remove 1 byte align</p> <p>RAW DATA #1 00000000: 20 20 20 2F 44 45 46 41 55 4C 54 4C 49 42 3A 22 /DEFAULTLIB:" 00000010: 4C 49 42 43 4D 54 22 20 2F 44 45 46 41 55 4C 54 LIBCMT" /DEFAULT 00000020: 4C 49 42 3A 22 4F 4C 44 4E 41 4D 45 53 22 20 LIB:"OLDNAMES" Linker Directives ----- /DEFAULTLIB:"LIBCMT" /DEFAULTLIB:"OLDNAMES"</p>	<p>SECTION HEADER #2 .debug\$\$ name 0 physical address 0 virtual address 70 size of raw data BB file pointer to raw data (000000BB to 0000012A) 0 file pointer to relocation table 0 file pointer to line numbers 0 number of relocations 0 number of line numbers 42100040 flags Initialized Data Discardable 1 byte align Read Only</p> <p>RAW DATA #2 00000000: 04 00 00 00 F1 00 00 00 61 00 00 00 23 00 01 11 ...ñ...a...#... 00000010: 00 00 00 00 63 3A 5C 55 73 65 72 73 5C 65 73 70 ...c:\Users\esp 00000020: 65 74 72 6F 76 5C 77 6F 72 6B 65 72 2E 6F 62 6A etrov\worker.obj 00000030: 00 3A 00 3C 11 00 22 00 00 07 00 0F 00 00 00 1E :.<.: "..... 00000040: 52 08 00 0F 00 00 00 1E 52 08 00 4D 69 63 72 6F R.....R..Micro 00000050: 73 6F 66 74 20 28 52 29 20 4F 70 74 69 6D 69 7A soft (R) Optimiz 00000060: 69 6E 67 20 43 6F 6D 70 69 6C 65 72 00 00 00 00 ing Compiler....</p> <p>SECTION HEADER #3 .text name 0 physical address 0 virtual address 5 size of raw data 12B file pointer to raw data (0000012B to 0000012F) 0 file pointer to relocation table 0 file pointer to line numbers 0 number of relocations 0 number of line numbers 60500020 flags</p>	<p>Code 16 byte align Execute Read</p> <p><u>_do_some_work:</u> 00000000: 55 push ebp 00000001: 8B EC mov ebp,esp 00000003: 5D pop ebp 00000004: C3 ret</p> <p>RAW DATA #3 00000000: 55 8B EC 5D C3 U.i]Ã</p> <p>COFF SYMBOL TABLE 000 0083521E ABS notype Static @comp.id 001 00000001 ABS notype Static @feat.00 002 00000000 SECT1 notype Static .drectve Section length 2F, #relocs 0, #linenums 0, checksum 0 004 00000000 SECT2 notype Static .debug\$\$ Section length 70, #relocs 0, #linenums 0, checksum 0 006 00000000 SECT3 notype Static .text Section length 5, #relocs 0, #linenums 0, checksum 672BE856 008 00000000 SECT3 notype () External _do_some_work</p> <p>String Table Size = 0x12 bytes</p> <p>Summary 70 .debug\$\$ 2F .drectve 5 .text</p>
---	---	--

Сборка (линковка)

```
c:\Users\espetrov>cl -c main.c
```

```
c:\Users\espetrov>dumpbin /all /disasm main.obj
```

<pre>Microsoft (R) COFF/PE Dumper Version 9.00.21022.08 Copyright (C) Microsoft Corporation. All rights reserved. Dump of file main.obj File Type: COFF OBJECT FILE HEADER VALUES 14C machine (x86) 3 number of sections 50482092 time date stamp Thu Sep 06 11:03:30 2012 13D file pointer to symbol table A number of symbols 0 size of optional header 0 characteristics SECTION HEADER #1 .drectve name 0 physical address 0 virtual address 2F size of raw data 8C file pointer to raw data (0000008C to 000000BA) 0 file pointer to relocation table 0 file pointer to line numbers 0 number of relocations 0 number of line numbers 100A00 flags Info Remove 1 byte align RAW DATA #1 00000000: 20 20 20 2F 44 45 46 41 55 4C 54 4C 49 42 3A 22 /DEFAULTLIB:" 00000010: 4C 49 42 43 4D 54 22 20 2F 44 45 46 41 55 4C 54 LIBCMT" /DEFAULT 00000020: 4C 49 42 3A 22 4F 4C 44 4E 41 4D 45 53 22 20 LIB:"OLDNAMES" Linker Directives ----- /DEFAULTLIB:"LIBCMT" /DEFAULTLIB:"OLDNAMES"</pre>	<pre>SECTION HEADER #2 .debug\$\$ name 0 physical address 0 virtual address 6C size of raw data BB file pointer to raw data (000000BB to 00000126) 0 file pointer to relocation table 0 file pointer to line numbers 0 number of relocations 0 number of line numbers 42100040 flags Initialized Data Discardable 1 byte align Read Only RAW DATA #2 00000000: 04 00 00 00 F1 00 00 00 5F 00 00 00 21 00 01 11ñ.....!... 00000010: 00 00 00 00 63 3A 5C 55 73 65 72 73 5C 65 73 70c:\Users\esp 00000020: 65 74 72 6F 76 5C 6D 61 69 6E 2E 6F 62 6A 00 3A etrov\main.obj.: 00000030: 00 3C 11 00 22 00 00 07 00 0F 00 00 00 1E 52 08 <.<.....R. 00000040: 00 0F 00 00 00 1E 52 08 00 4D 69 63 72 6F 73 6FR..Microso 00000050: 66 74 20 28 52 29 20 4F 70 74 69 6D 69 7A 69 6E ft (R) Optimizin 00000060: 67 20 43 6F 6D 70 69 6C 65 72 00 00 g Compiler.. SECTION HEADER #3 .text name 0 physical address 0 virtual address C size of raw data 127 file pointer to raw data (00000127 to 00000132) 133 file pointer to relocation table 0 file pointer to line numbers 1 number of relocations 0 number of line numbers 60500020 flags Code 16 byte align Execute Read</pre>	<pre>_main: 00000000: 55 push ebp 00000001: 8B EC mov ebp,esp 00000003: E8 00 00 00 00 call _do_some_work 00000008: 33 C0 xor eax,eax 0000000A: 5D pop ebp 0000000B: C3 ret RAW DATA #3 00000000: 55 8B EC E8 00 00 00 00 33 C0 5D C3 U.ìè...3À]Ā RELOCATIONS #3 Offset Type Symbol Applied To Index Name ----- 00000004 REL32 00000000 9 _do_some_work COFF SYMBOL TABLE 000 0083521E ABS notype Static @comp.id 001 00000001 ABS notype Static @feat.00 002 00000000 SECT1 notype Static .drectve Section length 2F, #relocs 0, #linenums 0, checksum 0 004 00000000 SECT2 notype Static .debug\$\$ Section length 6C, #relocs 0, #linenums 0, checksum 0 006 00000000 SECT3 notype Static .text Section length C, #relocs 1, #linenums 0, checksum 226120D7 008 00000000 SECT3 notype () External _main 009 00000000 UNDEF notype () External _do_some_work String Table Size = 0x12 bytes Summary 6C .debug\$\$ 2F .drectve C .text</pre>
---	---	--

Сборка (линковка)

```
c:\Users\espetrov>link main.obj worker.obj /nodefaultlib /entry:main -out:main.exe
```

```
c:\Users\espetrov>dumpbin/all /disasm main.exe
```

<p>Microsoft (R) COFF/PE Dumper Version 9.00.21022.08 Copyright (C) Microsoft Corporation. All rights reserved.</p> <p>Dump of file main.exe</p> <p>PE signature found</p> <p>File Type: EXECUTABLE IMAGE</p> <p>FILE HEADER VALUES</p> <p>14C machine (x86) 1 number of sections 504828F7 time date stamp Thu Sep 06 11:39:19 2012 0 file pointer to symbol table 0 number of symbols E0 size of optional header 103 characteristics Relocations stripped Executable 32 bit word machine</p> <p>OPTIONAL HEADER VALUES</p> <p>10B magic # (PE32) 9.00 linker version 200 size of code 0 size of initialized data 0 size of uninitialized data 1000 entry point (00401000) 1000 base of code 2000 base of data 400000 image base (00400000 to 00401FFF) 1000 section alignment 200 file alignment 5.00 operating system version 0.00 image version 5.00 subsystem version 0 Win32 version 2000 size of image 200 size of headers 0 checksum 3 subsystem (Windows CUI) 8400 DLL characteristics No structured exception handler Terminal Server Aware</p>	<p>100000 size of stack reserve 1000 size of stack commit 100000 size of heap reserve 1000 size of heap commit 0 loader flags 10 number of directories 0 [0] RVA [size] of Export Directory 0 [0] RVA [size] of Import Directory 0 [0] RVA [size] of Resource Directory 0 [0] RVA [size] of Exception Directory 0 [0] RVA [size] of Certificates Directory 0 [0] RVA [size] of Base Relocation Directory 0 [0] RVA [size] of Debug Directory 0 [0] RVA [size] of Architecture Directory 0 [0] RVA [size] of Global Pointer Directory 0 [0] RVA [size] of Thread Storage Directory 0 [0] RVA [size] of Load Configuration Directory 0 [0] RVA [size] of Bound Import Directory 0 [0] RVA [size] of Import Address Table</p> <p>Directory</p> <p>0 [0] RVA [size] of Delay Import Directory 0 [0] RVA [size] of COM Descriptor Directory 0 [0] RVA [size] of Reserved Directory</p>	<p>SECTION HEADER #1</p> <p>.text name 15 virtual size 1000 virtual address (00401000 to 00401014) 200 size of raw data 200 file pointer to raw data (00000200 to 000003FF) 0 file pointer to relocation table 0 file pointer to line numbers 0 number of relocations 0 number of line numbers 60000020 flags Code Execute Read</p> <p>00401000: 55 push ebp 00401001: 8B EC mov ebp,esp 00401003: E8 08 00 00 00 call 00401010</p> <p>00401008: 33 C0 xor eax,eax 0040100A: 5D pop ebp 0040100B: C3 ret 0040100C: CC int 3 0040100D: CC int 3 0040100E: CC int 3 0040100F: CC int 3 00401010: 55 push ebp 00401011: 8B EC mov ebp,esp 00401013: 5D pop ebp 00401014: C3 ret</p> <p>RAW DATA #1 00401000: 55 8B EC E8 08 00 00 00 33 C0 5D C3 CC CC CC CC U.iè....3À]Ãìììì 00401010: 55 8B EC 5D C3 U.ijÃ</p> <p>Summary 1000 .text</p>
--	---	---

Отладка

- Достижение работоспособности программы, устранение грубых ошибок
- Методы отладки
 - Имитация пошагового исполнения с помощью «карандаша и бумаги» для простых случаев
 - Трассировка работы программы с помощью отладочной печати
 - Проверка необходимых условий корректности в ходе работы программы
 - Пошаговое исполнение программы с помощью отладчика

Оптимизация

- Улучшение количественных характеристик программы
 - Время компиляции
 - Время загрузки
 - Время работы
 - Размер используемой памяти (данных на диске)
 - Размер исходного кода
 - Размер исполняемого кода
- Компилятор и линкер умеют автоматически делать *некоторые* преобразования программ, не зависящие от смысла (семантики) программы
 - Сохраняют корректность программы
 - Могут менять некорректную программу неожиданным образом
 - Могут *ухудшать* количественные характеристики программы
- За результат оптимизации отвечает программист
 - Понимая семантику программы, программист имеет возможность добиться большего эффекта, чем компилятор и линкер

Заключение

- Информация об этом курсе
- Понятие программы
- Этапы создания программ
 - Накопление требований, проектирование, документирование, сдача в эксплуатацию, сопровождение
 - Обзор
 - Разработка
 - Написание исходного кода
 - Компиляция исходного кода
 - Сборка
 - Отладка
 - Оптимизация
 - Тестирование