МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«Гжельский государственный университет»** (ГГУ)

Колледж ГГУ

Специальность 09.02.07 Информационные системы и программирования

**РЕФЕРАТ**

**По дисциплине «Моделирование и анализ программного обеспечения»**

**на тему «Обратное проектирование алгоритма»**

ВЫПОЛНИЛ:

Студент группы ИСП-О-17

Шашков И.С.

ПРОВЕРИЛА:

Прокуронова А.Ю.

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

п. Электроизолятор

1. г.

**Обра́тная разрабо́тка** (**обратное проектирование, обратный инжиниринг**, **реверс-инжиниринг**; [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *reverse engineering*) — исследование некоторого готового устройства или программы, а также [документации](https://ru.wikipedia.org/wiki/%D0%94%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0%D1%86%D0%B8%D1%8F_%D0%BD%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) на него с целью понять принцип его работы; например, чтобы обнаружить [недокументированные возможности](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%B4%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B5_%D0%B2%D0%BE%D0%B7%D0%BC%D0%BE%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D0%B8) (в том числе [программные закладки](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%B0%D1%8F_%D0%B7%D0%B0%D0%BA%D0%BB%D0%B0%D0%B4%D0%BA%D0%B0)), сделать изменение или воспроизвести устройство, программу или иной объект с аналогичными функциями, но без прямого копирования.

Применяется обычно в том случае, если создатель оригинального объекта не предоставил информации о структуре и способе создания (производства) объекта. Правообладатели таких объектов могут заявить, что проведение **обратной разработки** или использование её результатов нарушает их [исключительное право](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D0%BA%D0%BB%D1%8E%D1%87%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%B0%D0%B2%D0%BE) по закону об [авторском праве](https://ru.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%BE%D1%80%D1%81%D0%BA%D0%BE%D0%B5_%D0%BF%D1%80%D0%B0%D0%B2%D0%BE) и [патентному](https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D1%82%D0%B5%D0%BD%D1%82) законодательству

Исследование и обратная разработка программ обычно осуществляются с целью дальнейшей модификации, копирования, или, например, написания [генераторов ключей](https://ru.wikipedia.org/wiki/%D0%9A%D0%B5%D0%B9%D0%B3%D0%B5%D0%BD), алгоритм работы которых получен на основе анализа алгоритма их проверки. Также исследование программ применяется с целью получения некоторых закрытых сведений о внутреннем устройстве программы — о [протоколе](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_%D0%BF%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) [сетевого](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C) обмена с сервером, аппаратным средством, [ключом защиты](https://ru.wikipedia.org/w/index.php?title=%D0%9A%D0%BB%D1%8E%D1%87_%D0%B7%D0%B0%D1%89%D0%B8%D1%82%D1%8B&action=edit&redlink=1) или о взаимодействии с другой программой. Ещё одна область применения — получение информации о способах экспортирования данных из многочисленных [проприетарных](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%BF%D1%80%D0%B8%D0%B5%D1%82%D0%B0%D1%80%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) форматов файлов.

С развитием [Интернета](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82) популярные операционные системы и программы всё интенсивнее исследуются на предмет обнаружения в них [уязвимостей](https://ru.wikipedia.org/wiki/%D0%A3%D1%8F%D0%B7%D0%B2%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D1%8C_(%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%B1%D0%B5%D0%B7%D0%BE%D0%BF%D0%B0%D1%81%D0%BD%D0%BE%D1%81%D1%82%D1%8C)) или т. н. «дыр». В дальнейшем найденные дыры могут использоваться для получения несанкционированного доступа к удалённому компьютеру или компьютерной сети. C другой стороны, обратная разработка применяется при исследовании [антивирусными](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D1%82%D0%B8%D0%B2%D0%B8%D1%80%D1%83%D1%81%D0%BD%D0%B0%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0) компаниями [вредоносного ПО](https://ru.wikipedia.org/wiki/%D0%92%D1%80%D0%B5%D0%B4%D0%BE%D0%BD%D0%BE%D1%81%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) c целью добавления его [сигнатур](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%B3%D0%BD%D0%B0%D1%82%D1%83%D1%80%D0%B0_%D0%B0%D1%82%D0%B0%D0%BA%D0%B8) в базы своих продуктов.

Одним из широко известных примеров обратной разработки является исследование [BIOS](https://ru.wikipedia.org/wiki/BIOS) [персонального компьютера](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D1%81%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80) [IBM](https://ru.wikipedia.org/wiki/IBM), ставшее серьёзным шагом на пути развития производства IBM-совместимых компьютеров сторонними производителями. Создание [сервера](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5)) [Samba](https://ru.wikipedia.org/wiki/Samba) (входящего в состав ОС [GNU/Linux](https://ru.wikipedia.org/wiki/GNU/Linux) и работающего с серверами на базе ОС [Windows](https://ru.wikipedia.org/wiki/Windows)) также потребовало обратной разработки используемого [Microsoft](https://ru.wikipedia.org/wiki/Microsoft) [протокола](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_%D0%BF%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) [SMB](https://ru.wikipedia.org/wiki/SMB). Создание многих [ICQ](https://ru.wikipedia.org/wiki/ICQ)-клиентов также потребовало обратной разработки [протокола](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_%D0%BF%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) ICQ.

Обратная разработка программного обеспечения производится с помощью следующих методик.

1. Анализ обмена данными, наиболее распространённый в обратной разработке протоколов обмена данными, который производится с помощью [анализатора шины](https://en.wikipedia.org/wiki/Bus_analyzer) и [пакетного сниффера](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%82%D0%BE%D1%80_%D1%82%D1%80%D0%B0%D1%84%D0%B8%D0%BA%D0%B0) для прослушивания [шины компьютера](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D0%BD%D0%B0_(%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80)) и [компьютерной сети](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C) соответственно.
2. [Дизассемблирование](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%B7%D0%B0%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80) [машинного кода](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4) программы для получения её [листинга](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4) на [языке ассемблера](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%B0%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80%D0%B0). Этот способ работает на любой [компьютерной программе](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0), но требует достаточно много времени, особенно для неспециалиста.
3. [Декомпиляция](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D0%BA%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80) машинного или [байт-кода](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B9%D1%82-%D0%BA%D0%BE%D0%B4) программы для создания исходного кода на некотором [языке программирования высокого уровня](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D0%B2%D1%8B%D1%81%D0%BE%D0%BA%D0%BE%D0%B3%D0%BE_%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D1%8F).

В настоящее время под словами «reverse engineering» чаще всего понимается т. н. [clean room reverse engineering](https://ru.wikipedia.org/wiki/Clean_room_design), то есть процесс, при котором одна группа разработчиков анализирует [машинный код](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%88%D0%B8%D0%BD%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4) [программы](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0), составляет алгоритм данной программы на [псевдокоде](https://ru.wikipedia.org/wiki/%D0%9F%D1%81%D0%B5%D0%B2%D0%B4%D0%BE%D0%BA%D0%BE%D0%B4_(%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BE%D0%BF%D0%B8%D1%81%D0%B0%D0%BD%D0%B8%D1%8F_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D0%BE%D0%B2)) либо, если программа является [драйвером](https://ru.wikipedia.org/wiki/%D0%94%D1%80%D0%B0%D0%B9%D0%B2%D0%B5%D1%80) какого-либо устройства, составляет исчерпывающие [спецификации](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B5%D1%86%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F) интересующего устройства. После получения спецификаций другая группа разработчиков пишет собственный драйвер на основе полученных спецификаций или [алгоритмов](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC). Такой подход позволяет избежать обвинений в нарушении авторских прав на исходную программу, так как по законам, к примеру в [США](https://ru.wikipedia.org/wiki/%D0%A1%D0%A8%D0%90), попадает под понятие «[fair use](https://ru.wikipedia.org/wiki/%D0%94%D0%BE%D0%B1%D1%80%D0%BE%D1%81%D0%BE%D0%B2%D0%B5%D1%81%D1%82%D0%BD%D0%BE%D0%B5_%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)», то есть добросовестного использования оригинальной программы. Результат обратной разработки редко идентичен оригиналу, что и позволяет избежать ответственности перед законом, особенно при условии контроля отсутствия этой идентичности первой группой разработчиков и отсутствия нарушений торговых марок и патентов.

Основы реверс инжиниринга

Инициализация переменных

Переменные — одна из основных составляющих программирования. Они делятся на несколько видов, вот некоторые из них:

* строка;
* целое число;
* логическая переменная;
* символ;
* вещественное число с двойной точностью;
* вещественное число;
* массив символов.

Стандартные переменные:

string stringvar = "Hello World";

int intvar = 100;

bool boolvar = false;

char charvar = 'B';

double doublevar = 3.1415;

float floatvar = 3.14159265;

char carray[] = { 'a', 'b', 'c', 'd', 'e' };

**Примечание** в С++ строка — не примитивная переменная, но важно понять, как она будет выглядеть в машинном коде.

ассемблерный код:

[](https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse1.jpeg)

Инициализация переменных

Здесь можно увидеть как IDA показывает распределение пространства для переменных. Сначала под каждую переменную выделяется пространство, а потом уже она инициализируется.



Инициализация переменных

Как только пространство выделено, в него помещается значение, которое мы хотим присвоить переменной. Инициализация большинства переменных представлена на картинке выше, но как инициализируется строка, показано ниже.



Инициализация строковой переменной в C++

Для инициализации строки требуется вызов встроенной функции.

Стандартная функция вывода

**Примечание** Здесь речь пойдёт о том, что переменные помещаются в стек и затем используются в качестве параметров для функции вывода. Концепт функции с параметрами будет рассмотрен позднее.

Для вывода данных было решено использовать printf(), а не cout.

Стандартный вывод:

printf("Hello String Literal");

printf("%s", stringvar);

printf("%i", intvar);

printf("%c", charvar);

printf("%f", doublevar);

printf("%f", floatvar);

printf("%c", carray[3]);

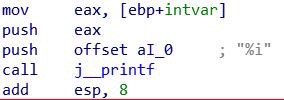
Теперь посмотрим на машинный код. Сначала строковый литерал:

https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse4.jpeg

Вывод строкового литерала

Как видите, строковый литерал сначала помещается в стек для вызова в качестве параметра функции printf().

Теперь посмотрим на вывод одной из переменных:



Вывод переменной

Как можно заметить, сначала переменная intvar помещается в регистр EAX, который в свою очередь записывается в стек вместе со строковым литералом %i, используемым для обозначения целочисленного вывода. Эти переменные затем берутся из стека и используются в качестве параметров при вызове функции printf().

Математические операции

Сейчас поговорим о следующих математических операциях:

1. Сложение.
2. Вычитание.
3. Умножение.
4. Деление.
5. Поразрядная конъюнкция (И).
6. Поразрядная дизъюнкция (ИЛИ).
7. Поразрядное исключающее ИЛИ.
8. Поразрядное отрицание.
9. Битовый сдвиг вправо.
10. Битовый сдвиг влево.

void mathfunctions() { // математические операции

int A = 10;

int B = 15;

int add = A + B;

int sub = A - B;

int mult = A \* B;

int div = A / B;

int and = A & B;

int or = A | B;

int xor = A ^ B;

int not = ~A;

int rshift = A >> B;

int lshift = A << B;

}

Переведём каждую операцию в ассемблерный код:

сначала присвоим переменной A значение 0A в шестнадцатеричной системе счисления или 10 в десятичной. Переменной B — 0F, что равно 15 в десятичной.

https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse6.jpeg

Инициализация переменных

Для сложения мы используем инструкцию add:

https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse7.jpeg

Сложение

При вычитании используется инструкция sub:

https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse8.jpeg

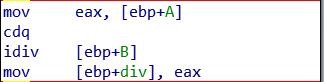
Вычитание

При умножении — imul:

https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse9.jpeg

Умножение

Для деления используется инструкция idiv. Также мы используем оператор cdq, чтобы удвоить размер EAX и результат деления уместился в регистре.



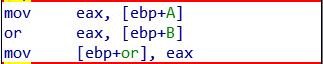
Деление

При поразрядной конъюнкции используется инструкция and:

https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse11.jpeg

Поразрядная конъюнкция

При поразрядной дизъюнкции — or:



Поразрядная дизъюнкция

При поразрядном исключающем ИЛИ — xor:

https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse13.jpeg

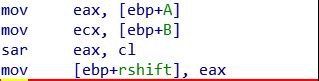
Поразрядное исключающее ИЛИ

При поразрядном отрицании — not:

https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse14.jpeg

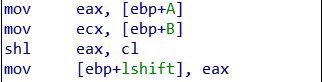
Поразрядное отрицание

При битовом сдвиге вправо — sar:



Битовый сдвиг вправо

При битовом сдвиге влево — shl:



Битовый сдвиг влево

Вызов функций

рассмотрим три вида функций:

1. Функция, не возвращающая значение (void).
2. Функция, возвращающая целое число.
3. Функция с параметрами.

Вызов функций:

newfunc();

newfuncret();

funcparams(intvar, stringvar, charvar);

Сначала посмотрим, как происходит вызов функций newfunc() и newfuncret(), которые вызываются без параметров.

https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse17.jpeg

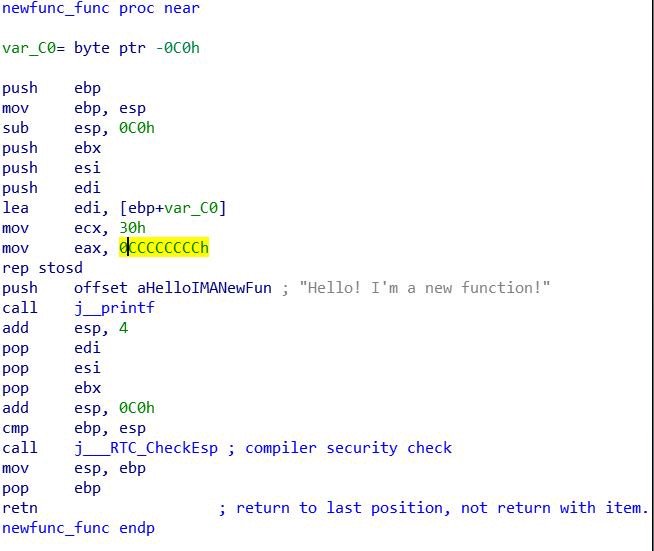
Вызов функций без параметров

Функция newfunc() просто выводит сообщение «Hello! I’m a new function!»:

void newfunc() { // новая функция без параметров

printf("Hello! I'm a new function"!);

}

[](https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse18.jpeg)

Функция newfunc()

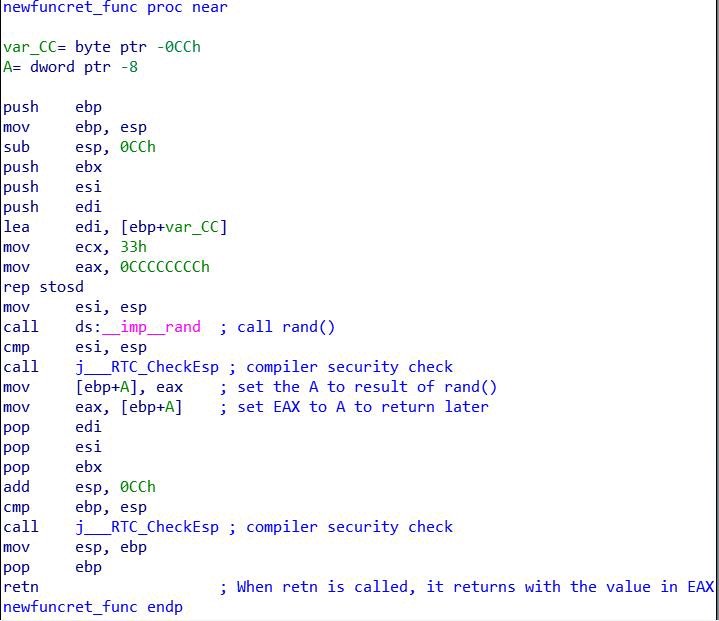
Эта функция использует инструкцию retn, но только для возврата к предыдущему местоположению (чтобы программа могла продолжить свою работу после завершения функции). Посмотрим на функцию newfuncret(), которая генерирует случайное целое число с помощью функции С++ rand() и затем его возвращает.

int newfuncret() { // новая функция, которая что-то возвращает

int A = rand();

return A;

}

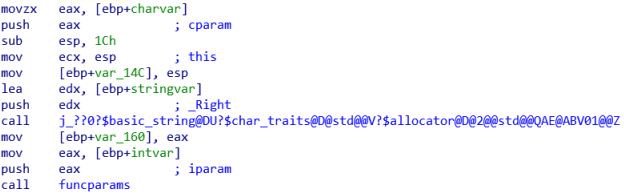


Функция newfuncret()

Сначала выделяется место под переменную A. Затем вызывается функция rand(), результат которой помещается в регистр EAX. Затем значение EAX помещается в место, выделенное под переменную A, фактически присваивая переменной A результат функции rand(). Наконец, переменная A помещается в регистр EAX, чтобы функция могла его использовать в качестве возвращаемого параметра. Теперь, когда мы разобрались, как происходит вызов функций без параметров и что происходит при возврате значения из функции, поговорим о вызове функции с параметрами.

Вызов такой функции выглядит следующим образом:

funcparams(intvar, stringvar, charvar);

[](https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse20.jpeg)

Вызов функции с параметрами

Строки в С++ требуют вызова функции basic\_string, но концепция вызова функции с параметрами не зависит от типа данных. Сначала переменная помещается в регистр, затем оттуда в стек, а потом происходит вызов функции.

Посмотрим на код функции:

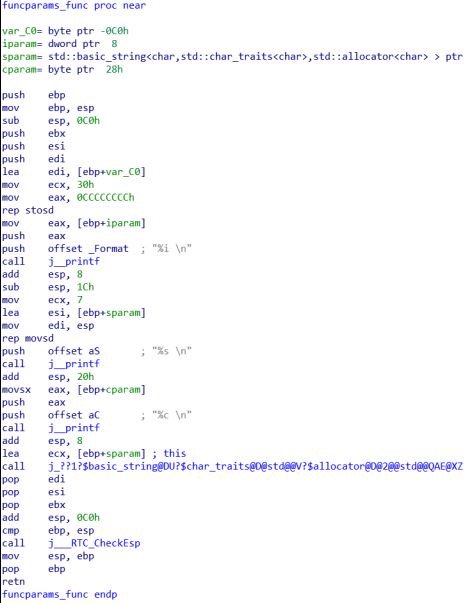
void funcparams (int iparam, string sparam, char cparam) { // функция с параметрами

printf("%i \n", iparam);

printf("%s \n", sparam);

printf("%c \n", cparam);

}

[](https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse21.jpeg)

Функция funcparams()

Эта функция берёт строку, целое число и символ и печатает их с помощью функции printf(). Как видите, сначала переменные размещаются в начале функции, затем они помещаются в стек для вызова в качестве параметров функции printf(). Очень просто.

Циклы

Теперь, когда мы изучили вызов функции, вывод, переменные и математику, перейдём к контролю порядка выполнения кода (flow control). Сначала мы изучим цикл for:

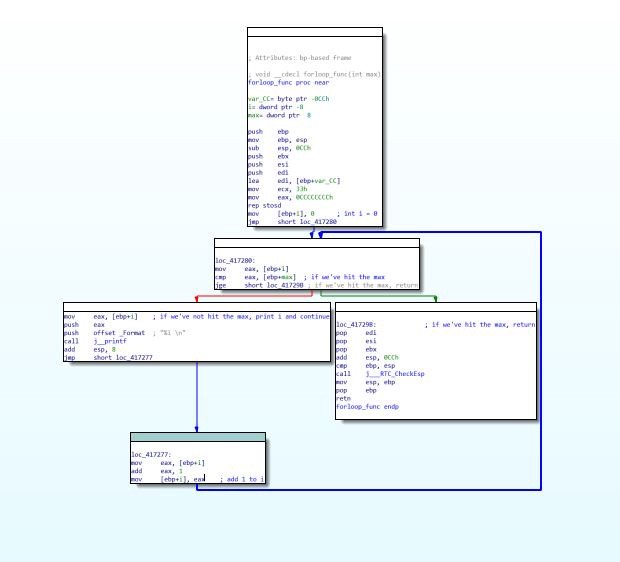
void forloop (int max) { // обычный цикл for

for (int i = 0; i < max; ++i){

printf("%i \n", i);

}

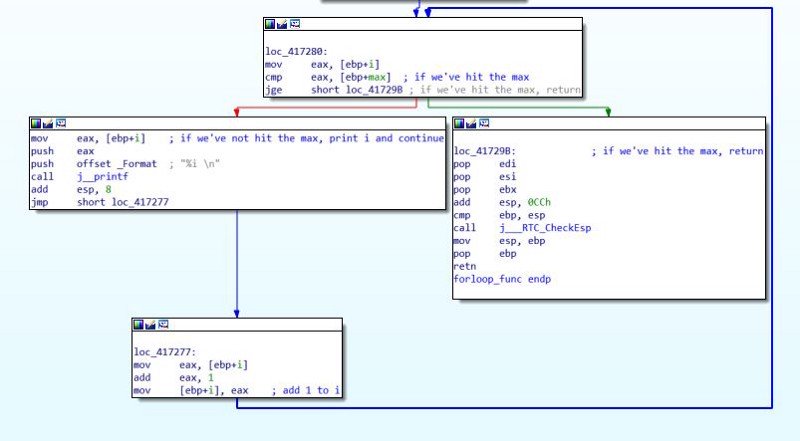
}

[](https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse22.jpeg)

Графический обзор цикла for

Прежде чем разбить ассемблерный код на более мелкие части, посмотрим на общий вариант. Как вы можете видеть, когда цикл for запускается, у него есть 2 варианта:

* он может перейти к блоку справа (зелёная стрелка) и вернуться в основную программу;
* он может перейти к блоку слева (красная стрелка) и вернуться к началу цикла for.

[](https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse23.jpeg)

Цикл for подробно

Сначала сравниваются переменные i и max, чтобы проверить, достигла ли переменная максимального значения. Если переменная i не больше или не равна переменной max, то подпрограмма пойдёт по красной стрелке (вниз влево) и выведет переменную i, затем i увеличится на 1 и произойдёт возврат к началу цикла. Если переменная i больше или равна max, то подпрограмма пойдёт по зелёной стрелке, то есть выйдет из цикла for и вернётся в основную программу.

Теперь давайте взглянем на цикл while:

void whileloop() { // цикл while

int A = 0;

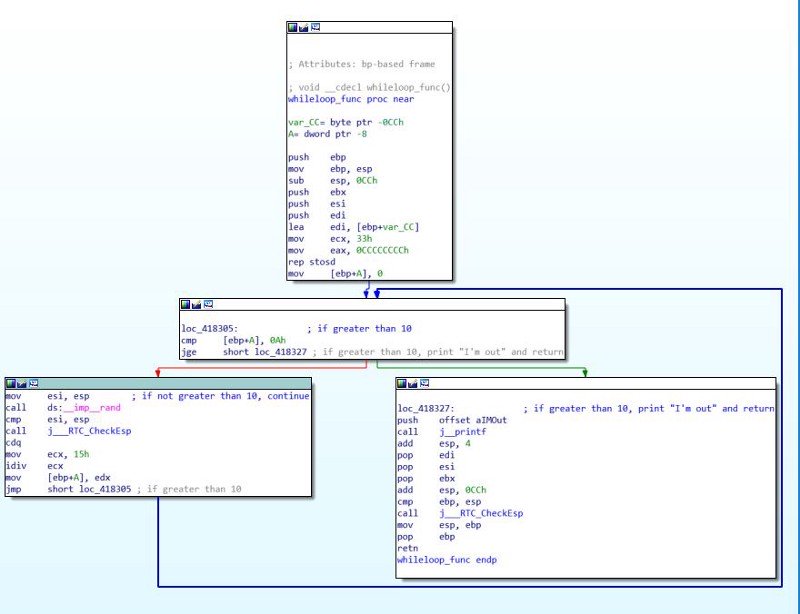
while (A<10) {

A = 0 + (rand()%(int)(20-0+1))

}

printf("I'm out!");

}

[](https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse24.jpeg)

Цикл while

В этом цикле генерируется случайное число от 0 до 20. Если число больше 10, то произойдёт выход из цикла со словами «I’m out!», в противном случае продолжится работа в цикле.

В машинном коде переменная А сначала инициализируется и приравнивается к нулю, а затем инициализируется цикл, A сравнивается с шестнадцатеричным числом 0A, которое равно 10 в десятичной системе счисления. Если А не больше и не равно 10, то генерируется новое случайное число, которое записывается в А, и снова происходит сравнение. Если А больше или равно 10, то происходит выход из цикла и возврат в основную программу.

Условный оператор

Теперь поговорим об условных операторах. Для начала посмотрим код:

void ifstatement() { // условные операторы

int A = 0 + (rand()%(int)(20-0+1));

if (A < 15) {

if (A < 10) {

if (A < 5) {

printf("less than 5");

}

else {

printf("less than 10, greater than 5");

}

}

else {

printf("less than 15, greater than 10");

}

}

else {

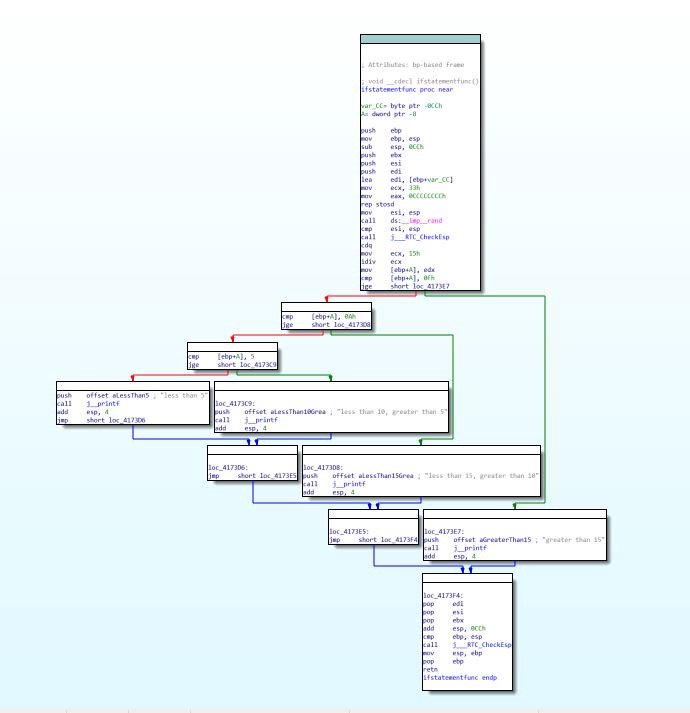
printf("greater than 15");

}

}

Эта функция генерирует случайное число от 0 до 20 и сохраняет получившееся значение в переменной А. Если А больше 15, то программа выведет «greater than 15». Если А меньше 15, но больше 10 — «less than 15, greater than 10». Если меньше 5 — «less than 5».

Посмотрим на ассемблерный граф:

[](https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse25.jpeg)

Ассемблерный граф для условного оператора

Граф структурирован аналогично фактическому коду, потому что условный оператор выглядит просто: «Если X, то Y, иначе Z». Если посмотреть на первую сверху пару стрелок, то оператору предшествует сравнение А с 0F, что равно 15 в десятичной системе счисления. Если А больше или равно 15, то подпрограмма выведет «greater than 15» и вернётся в основную программу. В другом случае произойдёт сравнение А с 0A (1010). Так будет продолжаться, пока программа не выведет что-нибудь на экран и не вернётся.

Оператор выбора

Оператор выбора очень похож на оператор условия, только в операторе выбора одна переменная или выражение сравнивается с несколькими «случаями» (возможными эквивалентностями). Посмотрим код:

void switchcase() { // оператор выбора

int A = 0 + (rand()%(int)(10-0+1));

switch (A) {

case 0:

printf("0");

break;

case 1:

printf("1");

break;

case 2:

printf("2");

break;

case 3:

printf("3");

break;

case 4:

printf("4");

break;

case 5:

printf("5");

break;

case 6:

printf("6");

break;

case 7:

printf("7");

break;

case 8:

printf("8");

break;

case 9:

printf("9");

break;

case 10:

printf("10");

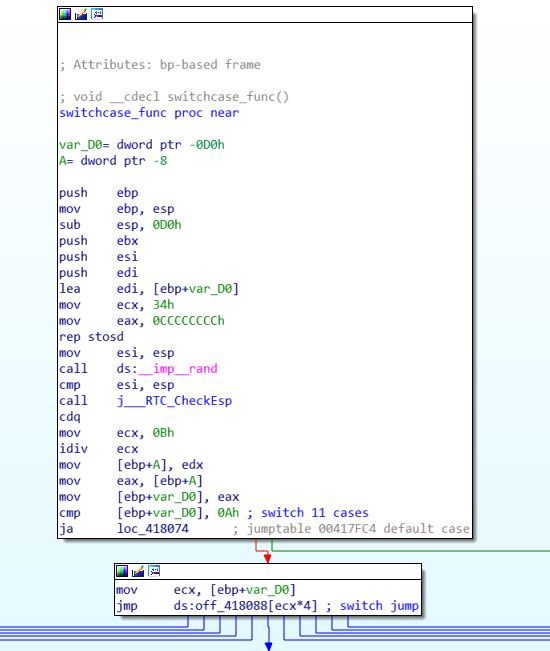
break;

}

}

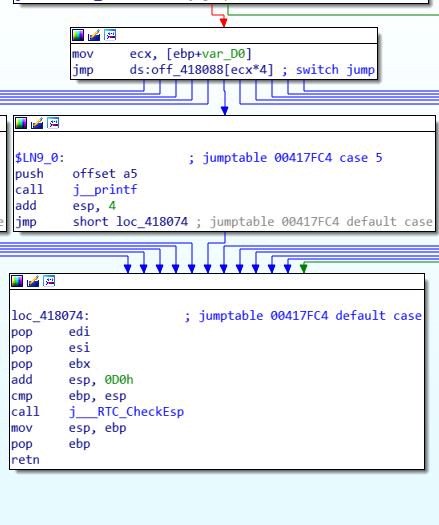
В этой функции переменная А получает случайное значение от 0 до 10. Затем А сравнивается с несколькими случаями, используя switch. Если значение А равно одному из случаев, то на экране появится соответствующее число, а затем произойдёт выход из оператора выбора и возврат в основную программу.

Оператор выбора не следует правилу «Если X, то Y, иначе Z» в отличии от условного оператора. Вместо этого программа сравнивает входное значение с существующими случаями и выполняет только тот случай, который соответствует входному значению. Рассмотрим два первых блока подробней.

[](https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse26.jpeg)

Два первых блока оператора выбора

Сначала генерируется случайное число и записывается в А. Теперь программа инициализирует оператор выбора, приравняв временную переменную var\_D0 к А, затем проверяет, что она равна хотя бы одному из случаев. Если var\_D0 требуется значение по умолчанию, то программа пойдёт по зелёной стрелке в секцию окончательного возврата из подпрограммы. Иначе программа совершит переход в нужный case.

[](https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse27.jpeg)

Если var\_D0 (A) равно 5, то код перейдёт в секцию, которая показана выше, выведет «5» и затем перейдёт в секцию возврата.

Пользовательский ввод

В этом разделе мы рассмотрим ввод пользователя с помощью потока сin из C++. Во-первых, посмотрим на код:

void userinput() { // ввод с клавиатуры

string sentence;

cin >> sentence;

printf("%s", sentence);

}

В этой функции мы просто записываем строку в переменную sentence с помощью функции C++ cin и затем выводим предложение с помощью оператора printf().

Разберём это в машинном коде. Во-первых, функция cin:

[](https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse28.jpeg)

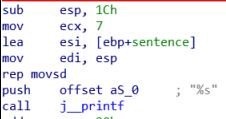
cin (C++)

Сначала происходит инициализация строковой переменной sentence, затем вызов cin и запись введённых данных в sentence.

[https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse29.jpeg](https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse29.jpeg)

Функция C++ cin детальнее

Сначала программа устанавливает содержимое переменной sentence в EAX, затем помещает EAX в стек, откуда значение переменной будет использоваться в качестве параметра для потока cin, затем вызывается оператор потока >>. Его вывод помещается в ECX, который затем помещается в стек для оператора printf():

[](https://cdn.tproger.ru/wp-content/uploads/2019/02/reverse30.jpeg)