Министерство образования и науки Челябинской области

государственное бюджетное профессиональное

образовательное учреждение

«Златоустовский индустриальный колледж им. П.П. Аносова»

**ЗАЩИТА**

Руководитель УП ПМ.03

Преподаватель ГБОУ ЗлатИК

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Юлия Ю.В

Оценка\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата\_\_\_\_\_\_\_\_\_\_\_\_

**ОТЧЕТ**

По учебной практике

Специальность: 09.02.07 «Информационные системы и программирование»

ПМ.03 «Ревьюирирование программных продуктов»

Выполнил:

Студент группы ИС-32

\_\_\_Спиридонов Н.С

\_\_\_\_\_\_\_\_\_\_\_дата

2021-2022 уч.г.

План прохождения учебной практики

по ПМ.03 «Ревьюирирование программных продуктов»

Студент Спиридонов Николай Сергеевич

Группа ИС-32

Специальность 09.02.07 «Информационные системы и программирование»

Количество часов \_\_\_\_\_36\_\_\_\_\_\_\_\_\_\_

Руководитель практики \_\_\_Юлия Владимировна Майер\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Сроки проведения практики\_\_\_\_29.11.-04.11.2021\_\_\_\_\_\_\_

Таблицу 1 – План прохождения практики

|  |  |  |
| --- | --- | --- |
| Этапы практики | Дата | Освоен  (да/нет) |
| Измерение производительности приложения посредством анализа использования ЦП | 29.11 |  |
| Установка ПО (Linux, компилятор GCC, radare2, iaito) для обратного проектирования | 30.11 |  |
| Обратное проектирование | 01.12 |  |
| Дизассемблирование | 02.12 |  |
| Разработать сопроводительную документацию | 03.12 |  |
| Защита практики | 04.12 |  |

**Отзыв руководителя**

Студент Спиридонов Николай Сергеевич

Группа ИС-32

Специальность 09.02.07 «Информационные системы и программирование»

Количество часов \_\_\_\_\_36\_\_\_\_\_\_\_\_\_\_

Руководитель практики \_\_\_Юлия Владимировна Майер\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Сроки проведения практики\_\_\_\_29.11– 04.12 2021г.\_\_\_\_\_\_\_

Место работы ГБПОУ «Златоустовский индустриальный колледж им. П.П. Аносова»

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Показатели и критерии оценивания** | **Баллы** | **Фактический балл** |
| **1** | **Структура отчета по практике** | | |
|  | Структура полностью соответствует заданию | 3 |  |
| Структура имеет несущественное несоответствие заданию | 2 |  |
| Структура существенно не соответствует заданию | 1 |  |
| **2** | **Соответствие содержания рабочей теме,цели,задачам** |  |  |
|  | Полное соответствие | 3 |  |
| Частичное соответствие | 2 |  |
| Низкая степень соответствия | 1 |  |
| **3** | **Полнота раскрытия темы** | | |
|  | Тема раскрыта полностью, приведены необходимые расчеты, пояснения, аргументы, сделаны выводы | 3 |  |
| Тема раскрыта полностью, однако приведены не все необходимые расчеты, пояснения и аргументы | 2 |  |
| Тема раскрыта частично, нет необходимых расчетов, пояснений, аргументов, не сделаны выводы | 1 |  |
| **4** | **Логика изложения материала** | | |
|  | Все структурные элементы организованы в систему, прослеживается логика в раскрытии темы | 3 |  |
| Все структурные элементы организованы в систему, логика в раскрытии темы частично нарушена | 2 |  |
| Все структурные элементы организованы в систему, но нет логики в раскрытии темы | 1 |  |
| **5** | **Соблюдение требований ГОСТ к оформлению ПЗ** | | |
|  | Требования ГОСТ соблюдены полностью | 3 |  |
| Имеются незначительные отклонения от ГОСТ | 2 |  |
| Есть существенные нарушения требований ГОСТ | 1 |  |
| **6** | **Практическая часть** | | |
|  | Выполнена в соответствии с требованиями, без отклонений от нормативов | 3 |  |
| Имеется несущественное отклонение от нормативов | 2 |  |
| Имеется существенное отклонение от нормативов | 1 |  |
| ИТОГО | | |  |

Критерии оценивания:

18-15 б. – «отлично»;

11-14 б. – «хорошо»;

8-10 б. – «удовлетворительно»;

Меньше 8 б. работа не оценивается.

Подпись руководителя:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Ю.В.Майер

\_\_\_\_\_\_\_\_\_\_\_\_\_дата

Содержание

[**ВВЕДЕНИЕ** 5](#_Toc89421740)

[**1.** **Список терминов и определений** 7](#_Toc89421741)

[**2.** **Функциональные требования** 9](#_Toc89421742)

[**3.** **Измерение производительности приложения посредством анализа использования центрального процессора** 10](#_Toc89421743)

[3.1. Сбор данных профилирования 10](#_Toc89421744)

[3.2. Анализ данных о загрузке центрального процессора 14](#_Toc89421746)

[**4.** **Установка ПО** 16](#_Toc89421747)

[4.1. Установка Linux Ubuntu 16](#_Toc89421748)

[4.2. Установка остального ПО через терминал 21](#_Toc89421749)

[**6. Дизассемблирование** 23](#_Toc89421750)

[**7. Обратное проектирование** 25](#_Toc89421751)

[**ЗАКЛЮЧЕНИЕ** 27](#_Toc89421752)

**ВВЕДЕНИЕ**

Процессы разработки, приобретения и внедрения сложных систем, к которым относятся в частности программные комплексы, должны находится под жестким управленческим контролем. В настоящее время практически во всех организациях обеспечивается контроль важнейших характеристик, связанных с производством и использованием программных продуктов, таких как время, финансовые средства, ресурсы и т.п. Однако в большинстве случаев вне пределов сферы контроля оказывается наиболее важная характеристика программных продуктов, ради которой, собственно и осуществляются затраты времени, финансовых средств и ресурсов – это качество продукта, поскольку «невозможно контролировать то, что нельзя измерить» (“You cannot control what you cannot measure”).

Дизассемблирование **–** преобразованиепрограммы на машинном языке к ее ассемблерному представлению. Декомпиляция – получение кода языка высокого уровня из программы на машинном языке или ассемблере.

Под *анализом потоков данных* понимают совокупность задач, нацеленных на выяснение некоторых глобальных свойств программы, то есть извлечение информации о поведении тех или иных конструкций в некотором контексте.

Основным результатом деятельности группы разработчиков являются не диаграммы, а программное обеспечение, поэтому модели и основанные на них реализации должны соответствовать друг другу с минимальными затратами по поддержанию синхронизации между ними. Чаще всего разработанные модели преобразуются в программный код. Хотя UML не определяет конкретного способа отображения на какой-либо объектно-ориентированный язык, он проектировался с учетом этого требования. В наибольшей степени это относится к диаграммам классов, содержание которых без труда отображается на такие известные объектно-ориентированные языки программирования, как Java, C++, ObjectPascal, Visual Basic и др.

*Прямым проектированием* (Forward engineering) называется процесс преобразования модели в код путем отображения на некоторый язык реализации.

*Обратным проектированием* (Reverse engineering) называется процесс преобразования в модель кода, записанного на каком-либо языке программирования.

Цель учебной практики: Научиться выполнять прямое и обратное проектирование ПО, овладеть навыками приостановки и возобновления работы с рабочими задачами иинспекцией кода в Visual Studio, совместно работать над проектом в системе контроля версий GIT.

В ходе учебной практики для достижения цели, ставятся задачи:

* измерить производительность приложения посредством анализа использования ЦП,
* установить ПО (Linux, компилятор GCC, radare2, iaito) для обратного проектирования,
* выполнить обратное проектирование,
* выполнить дизассемблирование.

# **Список терминов и определений**

**Пользователь**— человек, который имеет, имел, или, возможно, будет иметь доступ в систему для совершения операций.

**Операция** — совокупность действий, составляющих содержание одного акта бизнес-деятельности. Операция должна соответствовать требованиям ACID (Atomicity, Consistency, Isolation, Durability). Совокупность операций одного модуля представляет интерфейс взаимодействия клиент-сервер этого модуля.

**Центральный процессор** (ЦП) – главная часть аппаратного обеспечения компьютера или программируемого логического контроллера. Иногда называют микропроцессором или просто процессором.

**Точка останова** – это преднамеренное прерывание выполнения программы, при котором выполняется вызов отладчика.

**Прокси** – это удаленный компьютер, который, при подключении к нему вашей машины, становится посредником для выхода абонента в интернет. Прокси передает все запросы программ абонента в сеть, и, получив ответ, отправляет его обратно абоненту.

**Репозиторий**(от англ. **repository** – склад, хранилище) – это профессиональный участник рынка ценных бумаг, осуществляющий ведение реестров договоров РЕПО и договоров с деривативами, заключенных на организованных торгах.

**GitHub** — это платформа, хранящая различные Git-репозитории на своих многочисленных серверах. Также GitHub называют крупнейшим веб-сервисом для хостинга и совместной разработки IT-проектов.

**Дизассемблирование** – это получение из исполняемого кода программы код на языке ассемблера.

**Дизассемблер** - программа, осуществляющая дизассемблирова­ние.

**Компилятор** — программа, переводящая текст, написанный на языке программирования, в набор машинных кодов.

**Декомпиляция** – это процесс восстановления исходного кода программы из машинного кода.

**Декомпилятор** – это программа, которая может совершить процесс декомпиляции.

**Обратное проектирование** – исследование некоторого готового устройства или программы, а также документации на него с целью понять принцип его работы; например, чтобы обнаружить недокументированные возможности, сделать изменение или воспроизвести устройство, программу или иной объект с аналогичными функциями, но без прямого копирования.

# **Функциональные требования**

ПО, используемое в ходе выполнения практики:

* Visual Studio 2019
* Oracle VM Virtual Box – программа для виртуализации ОС, в которой нужно создать виртуальную ОС Linux с дистрибутивом Ubuntu 64x
* Компилятор GCC - компилятор для различных языков программирования, используемый для свободных Unix-подобных операционных систем.
* Radare2 - свободный кроссплатформенный фреймворк для реверс-инжиниринга, который используется при реверсе, отладке вредоносного ПО и прошивок.
* Iaito - это одна из оболочек Radare2, предназначенная для платформы Linux и устанавливаемая при помощи терминала с веб-сайта GitHub.
* Декомпилятор r2ghidra

1. **Измерение производительности приложения посредством анализа использования центрального процессора**
   1. Создание репозитория

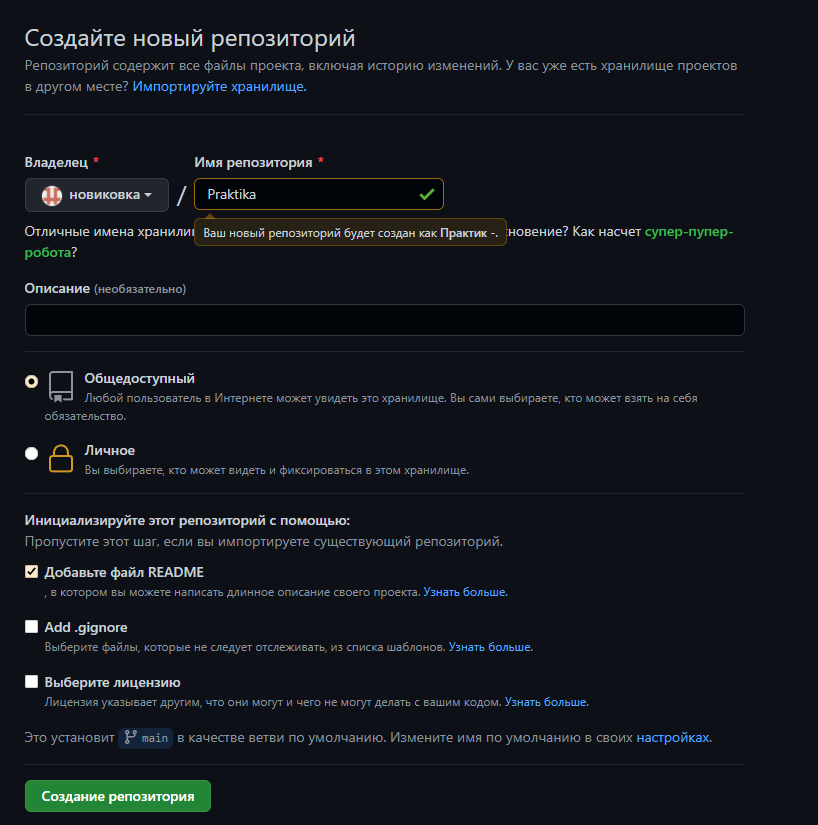


Рис.1-Создание репозитория

### Сбор данных профилирования

* + 1. Запускаем Visual Studio, открываем любой проект

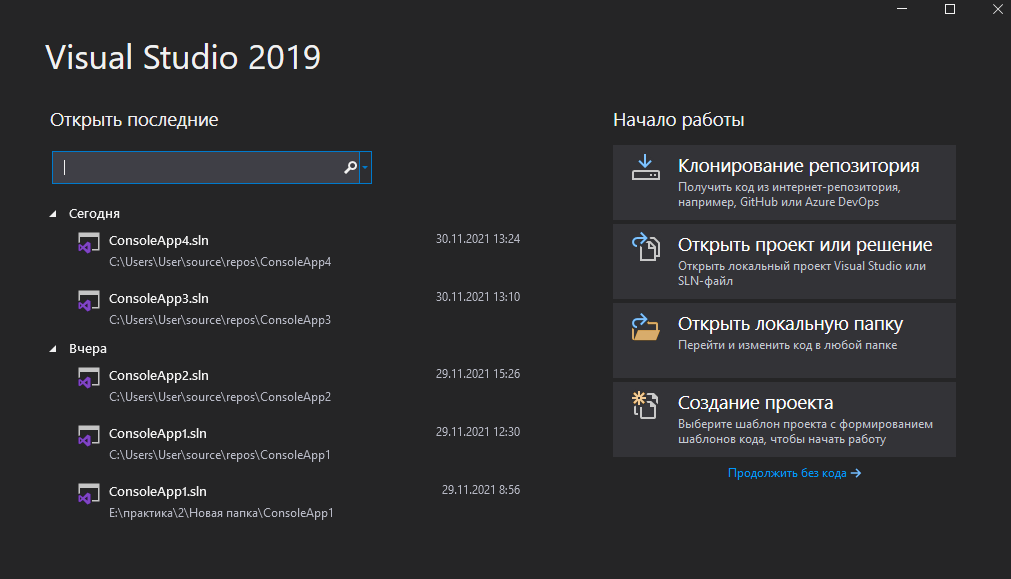


Рис. 1 -Visual Studio

3.1.2. Установите точку останова в приложении в точке, где вы хотите проверить загрузку центрального процессора.

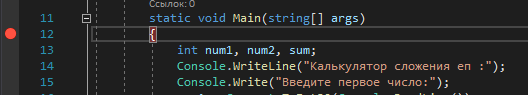


Рис. 2 -Первая точка остановы

* + 1. Устанавливаем вторую точку останова в конце функции или области кода, который требуется проанализировать.

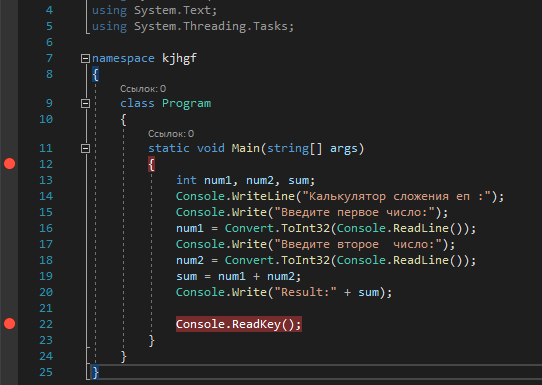


Рис. 3 -Вторая точка остановы

3.1.4. Окно Средства диагностики появится автоматически, если вы не отключали эту функцию. Чтобы снова открыть окно, щелкните **Отладка > Окна > Показать средства диагностики.**

3.1.5. Вы можете выбрать, что следует просмотреть, Использование памяти или Использование ЦП (либо оба средства), с помощью параметра Выбор средств на панели инструментов.

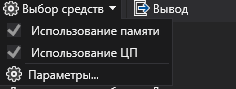


Рис. 4 -Выбор средств

3.1.6. По завершении загрузки приложения отображается представление "Сводка" средств диагностики.

3.1.7. Запустите сценарий, который вызвал срабатывание первой точки останова.

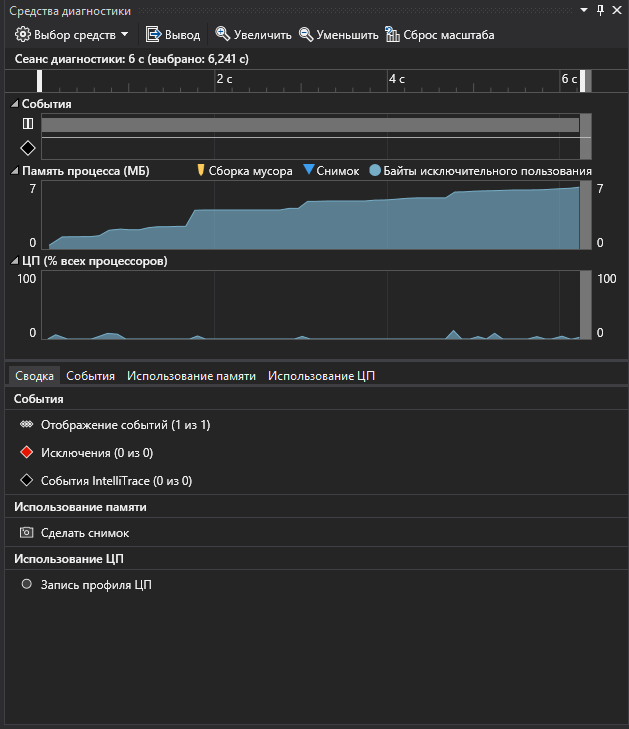


Рис. 5 -Окно диагностики во время отладки

3.1.8. Приостановив отладчик, включите сбор данных о загрузке центрального процессора, а затем откройте вкладку **Загрузка ЦП.** При выборе пункта Запись профиля ЦП Visual Studio начнет записывать функции и сведения о времени их выполнения. Эти собранные данные можно просматривать только в том случае, если приложение останавливается в точке останова.

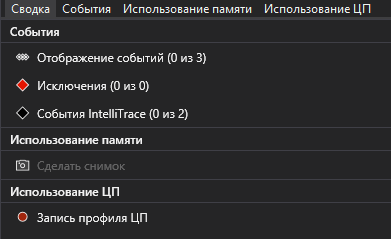


Рис. 6 -Запись профиля ЦП

3.1.9. Нажмите клавишу F5, чтобы запустить приложение до второй точки останова.

Теперь у вас есть данные о производительности приложения именно для той области кода, которая выполняется между двумя точками останова.

Профилировщик начинает подготавливать данные потока. Дождитесь завершения этой операции.

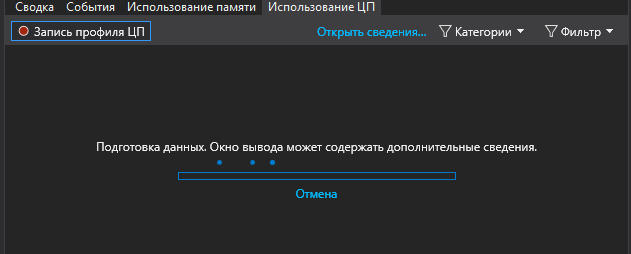


Рис. 7 -Подготовка данных

Средство "Загрузка ЦП" выведет отчет на вкладке **Загрузка ЦП.**

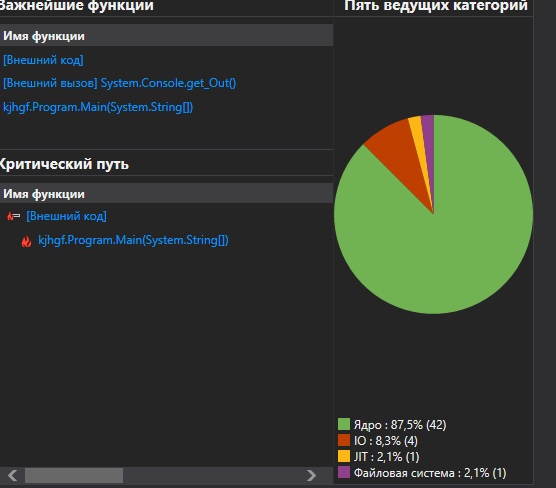


Рис. 8 -Загрузка центрального процессора

### 

### 3.2. Анализ данных о загрузке центрального процессора

3.2.1. В списке функций дважды щелкните одну из функций вашего приложения, которая выполняет много работы.

При двойном щелчке функции в левой панели откроется представление **Вызывающий/вызываемый.**

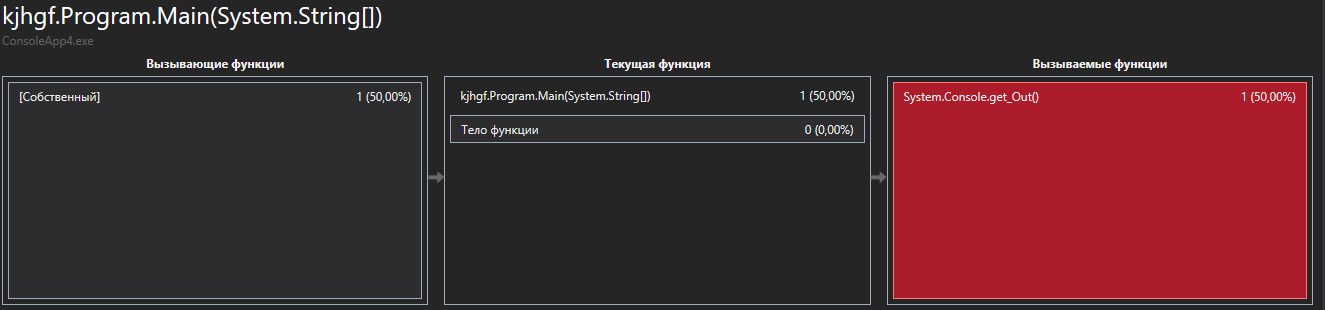


Рис. 9 -Представление Вызывающий/вызываемый

В этом представлении выбранная функции отображается в заголовке и в поле Текущая функция. Функция, вызывавшая текущую функцию, отображается в левой части окна в разделе Вызывающие функции, а все функции, вызываемые текущей функцией, отображаются в поле Вызываемые функции справа. (Можно выбрать любое поле, чтобы изменить текущую функцию.) В этом представлении показано общее время (мс) и доля общего времени выполнения приложения, затраченного на выполнение функции. В поле Тело функции также показан общий объем времени (и доля времени), затраченного в теле функции за исключением времени, затраченного в вызываемых и вызывающих функциях.

3.2.2. Чтобы увидеть более обобщенное представление, показывающее порядок, в котором вызываются функции, выберите в раскрывающемся списке в верхней части панели пункт **Дерево вызовов**.

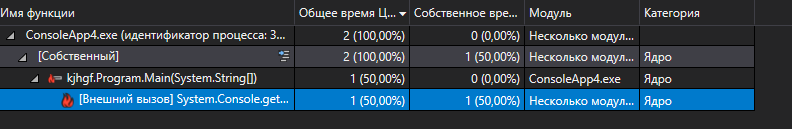


Рис. 10 -Дерево вызовов

Чтобы увидеть вызовы функций, которые используют самый высокий процент ЦП в представлении дерева вызовов, нажмите **«Развернуть критический путь».**

Просмотр внешнего кода

Чтобы посмотреть пути к вызовам внешнего кода, выберите **Показать внешний код** в списке **Представление фильтра** и выберите **Применить.**

1. **Установка ПО**
   1. Установка Linux Ubuntu

Для начала нужно установить Virtual Box что бы на нём поставить Linux.

После нажимаем создать, указываем имя, тип, версию, объем памяти,

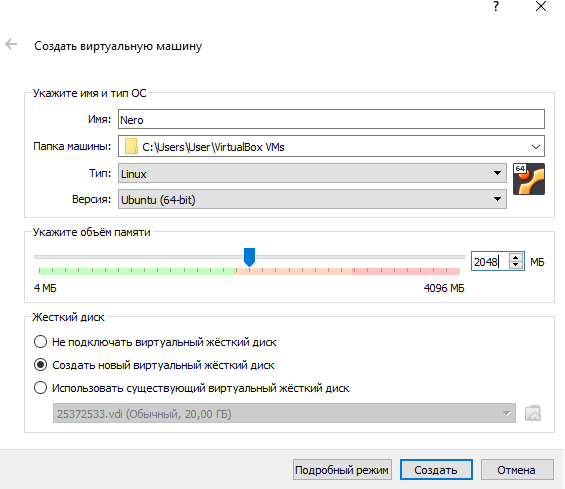


Рис. 11 -Настраиваем машину

Создаём новый жесткий диск

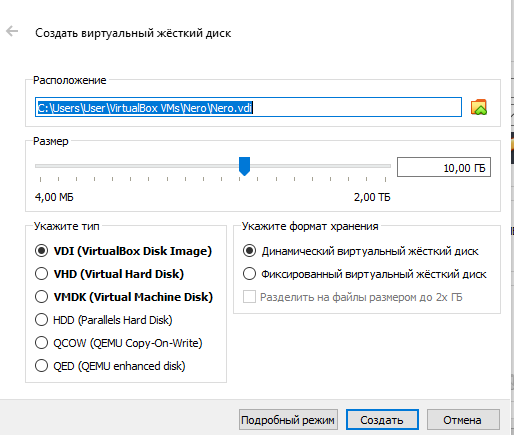


Рис. 12 -Новый жесткий диск

Затем в настройках нужно поставить контроллер

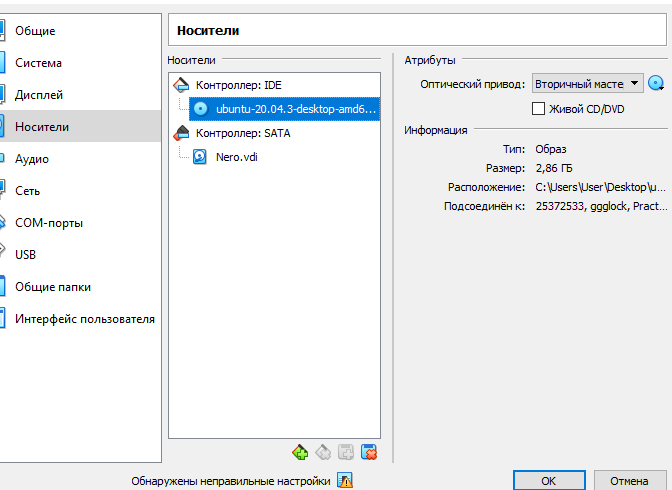


Рис. 13 -Выбираем привод диска

Запускаем нашу машину

После загрузки можно выбрать язык и нажимаем «Install Ubuntu»

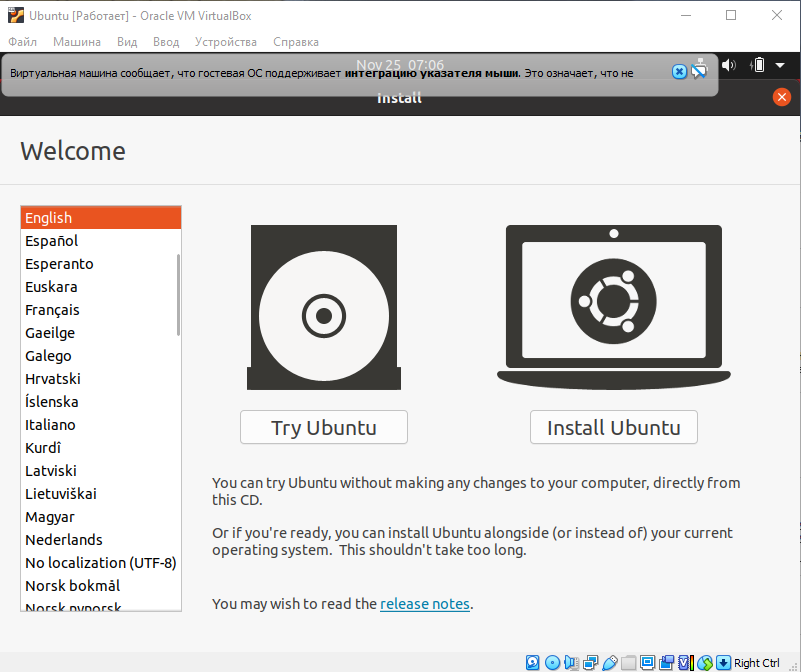


Рис. 14 -Выбираем Install Ubuntu

Выбираем «Minimal installation» и нажимаем продолжить

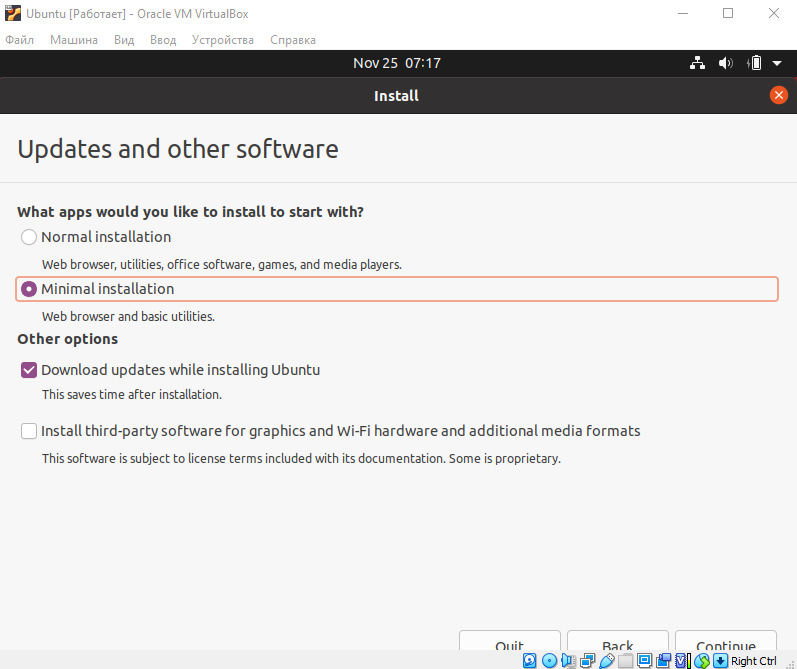


Рис.15-Выбираем **Minimal installation**

Далее ничего не меняем и продолжаем

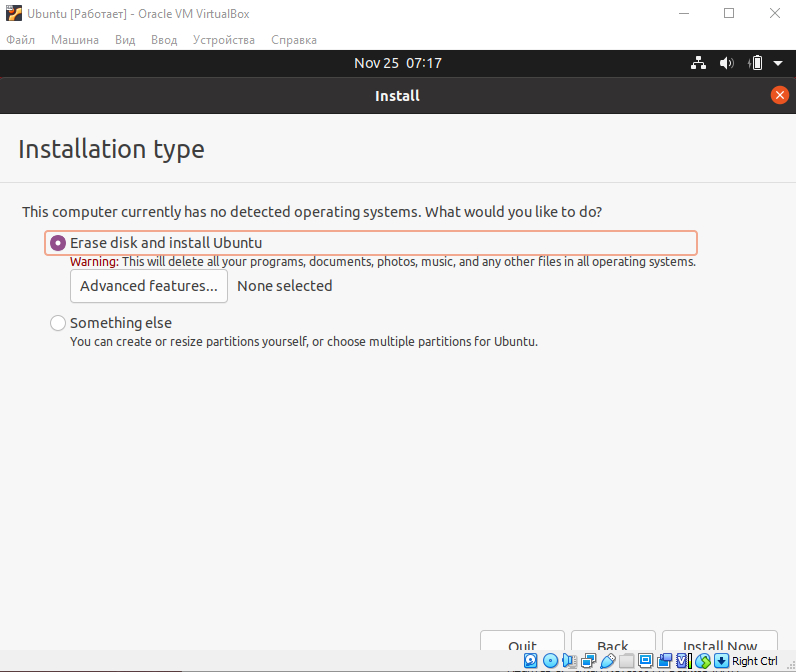


Рис.16 -Installation type

Выбираем свой регион



Рис. 17 -Регион

Далее вводим свой логин и пароль

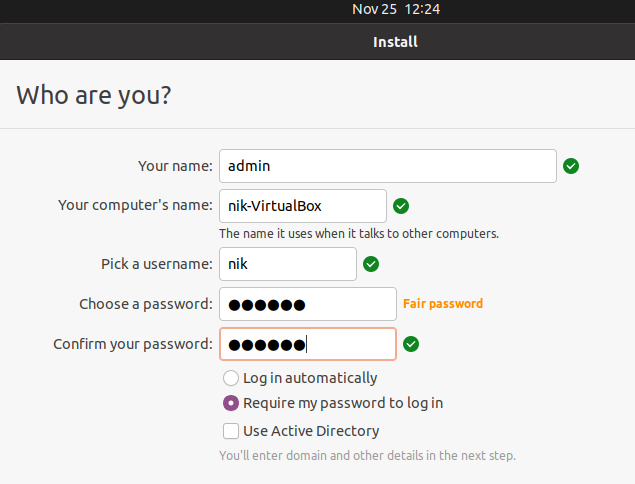


Рис. 18 -Авторизация Логин и пароль

Далее происходит установка

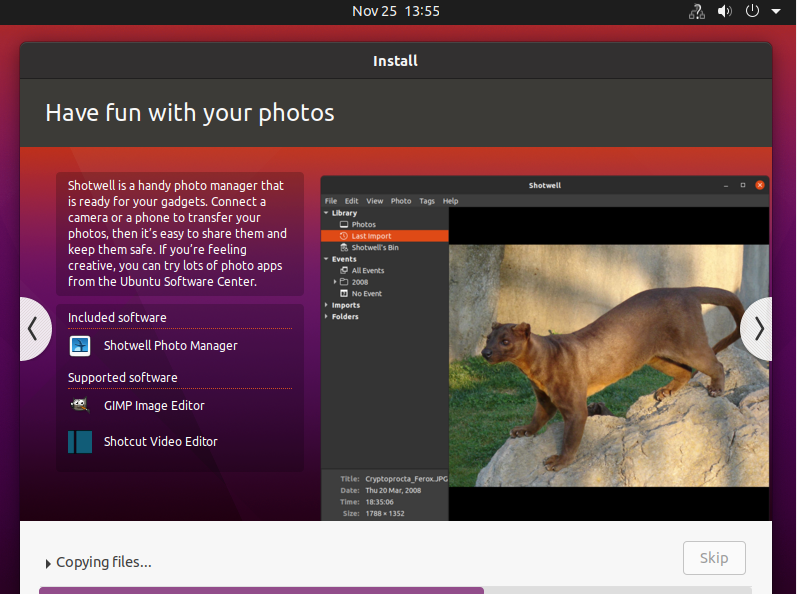
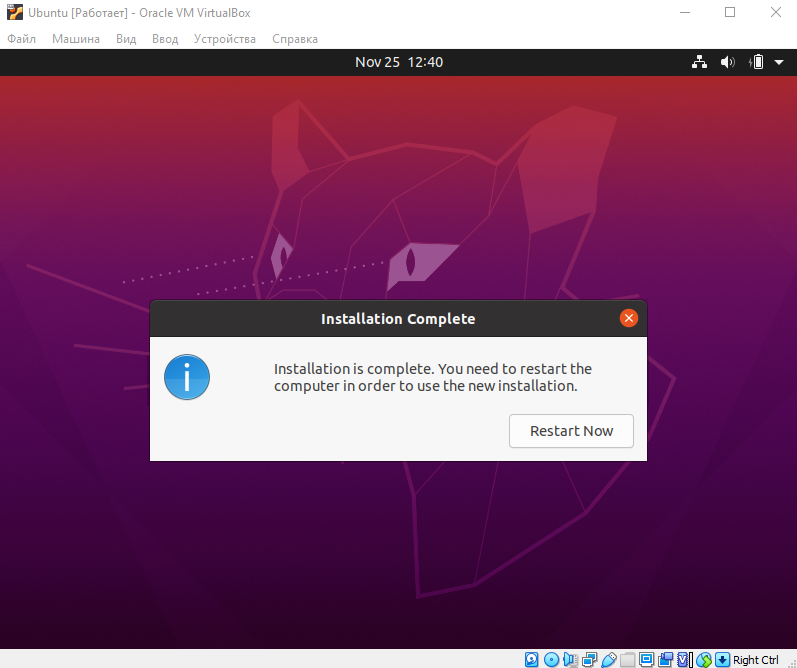


Рис. 19 -Установка

После установки нужно перезапустить машину

Рис. 20 -Перезагрузка



* 1. Установка остального ПО через терминал
     1. Устанавливаем компилятор GCC и все необходимые для него компоненты

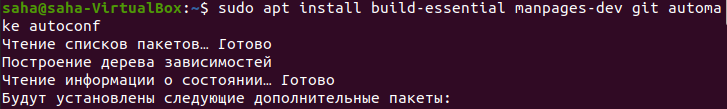


Рис. 21 -Установка GCC

* + 1. Установка radare2

Устанавливаем необходимые пакеты для работы с radare2

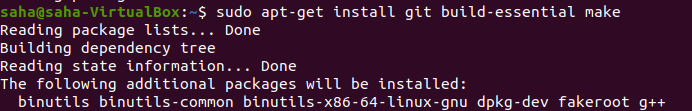


Рис. 22 -Установка пакетов

Так как используется компьютер в колледже прописываем:

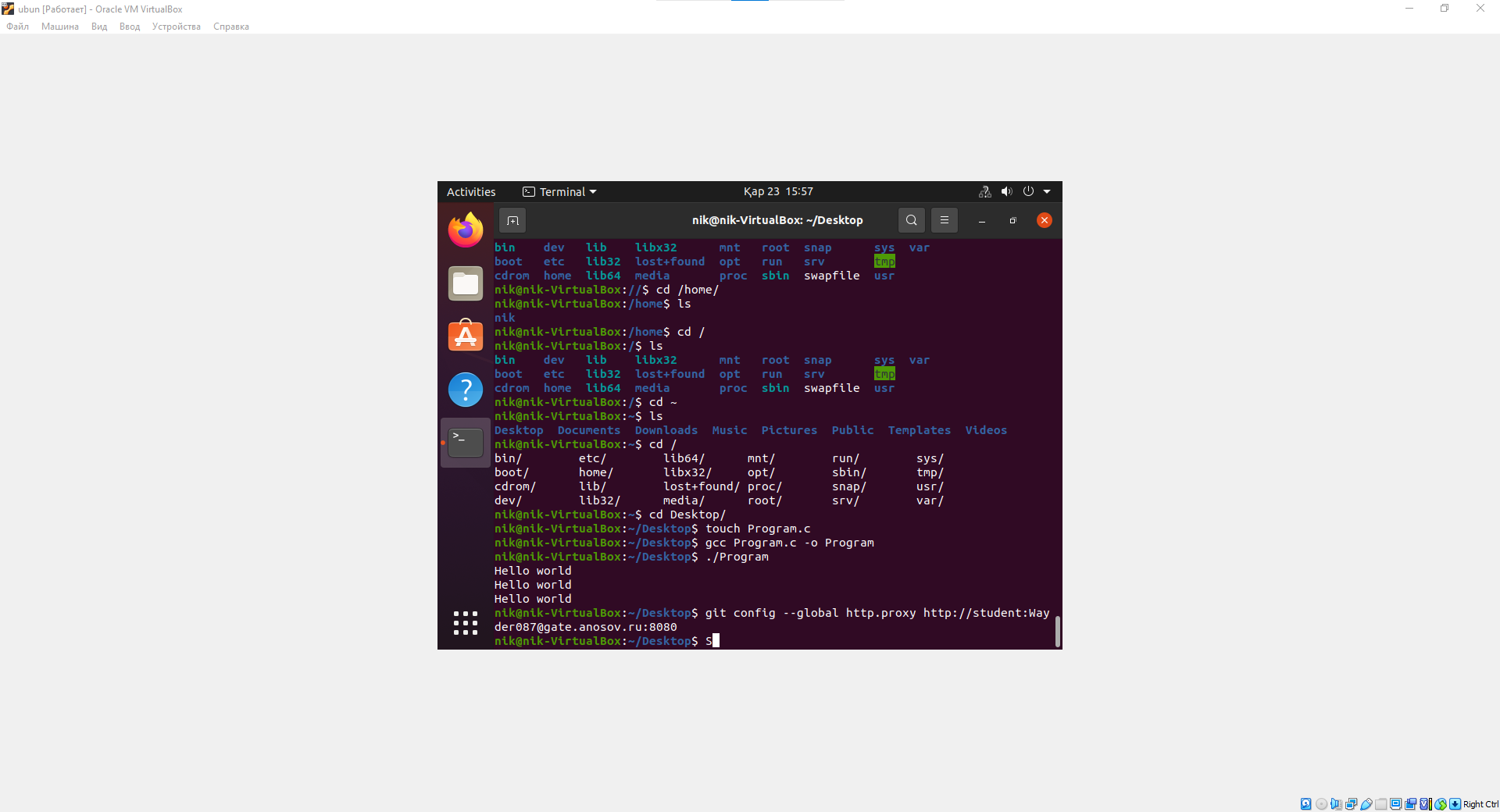


Рис. 23 -Прописываем прокси

sudo nano /etc/wgetrc

Ищем прокси и меняем на:

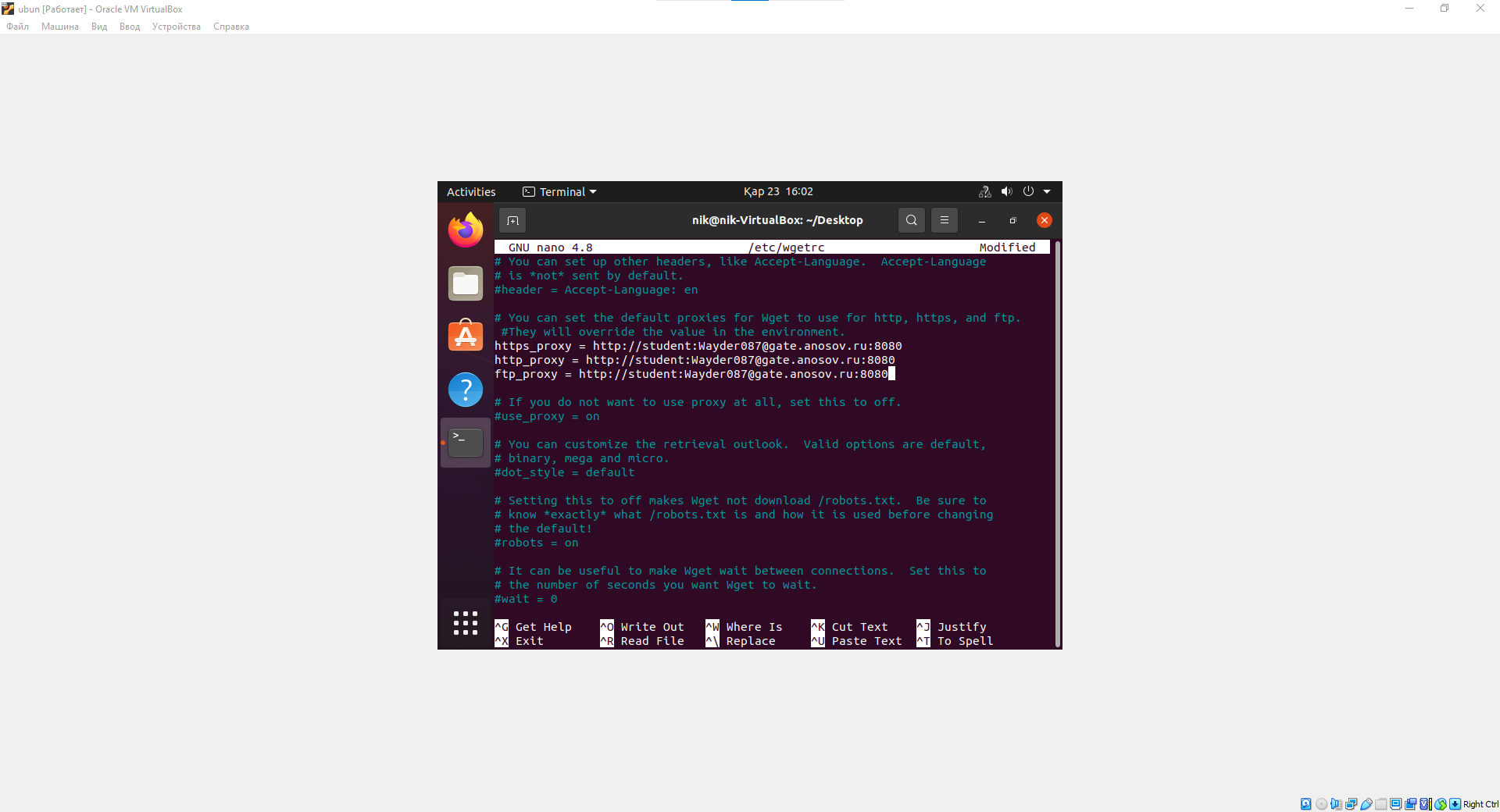


Рис. 24 -Меняем прокси

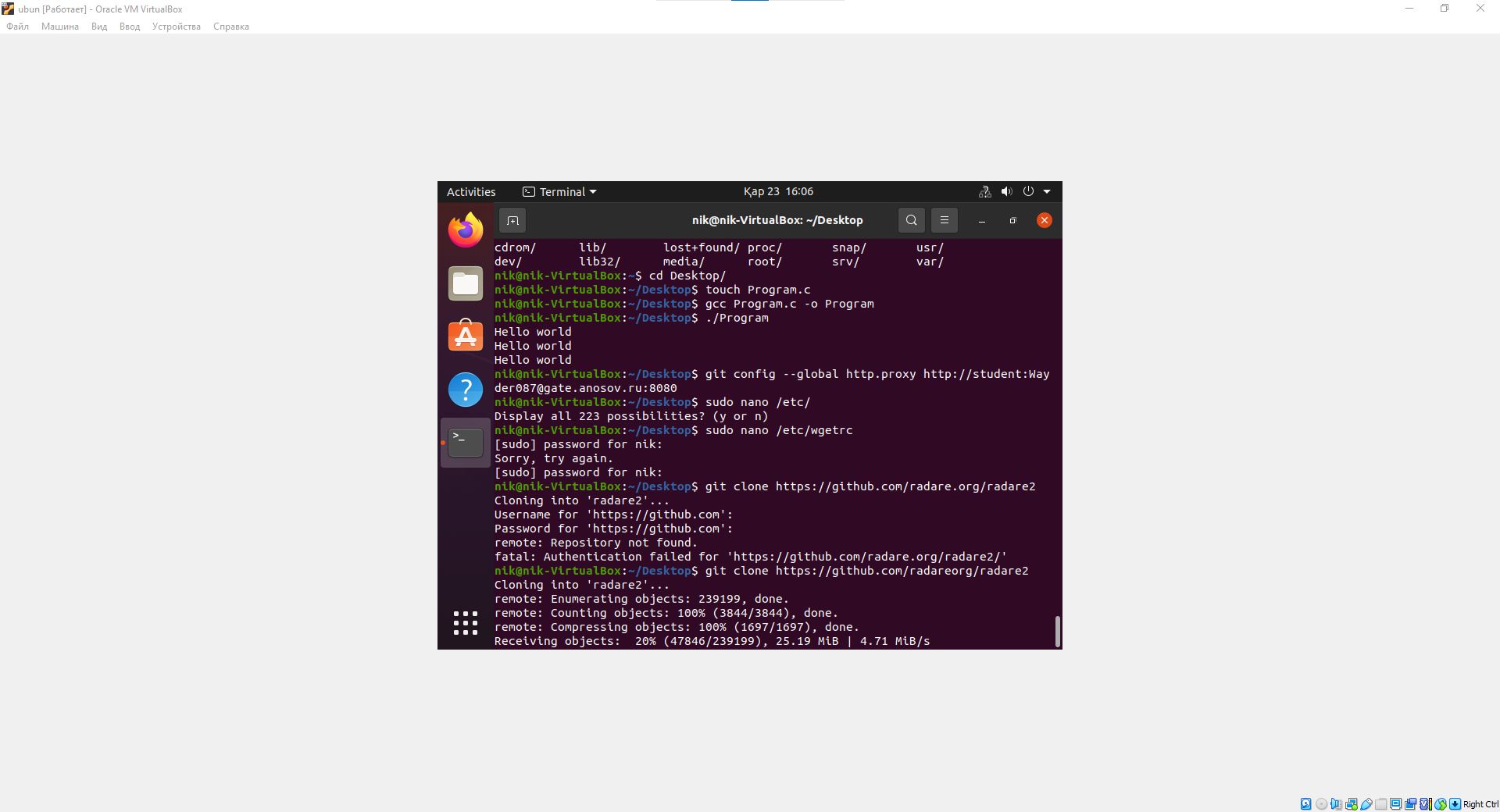


Рис.25-Клонирование radare2

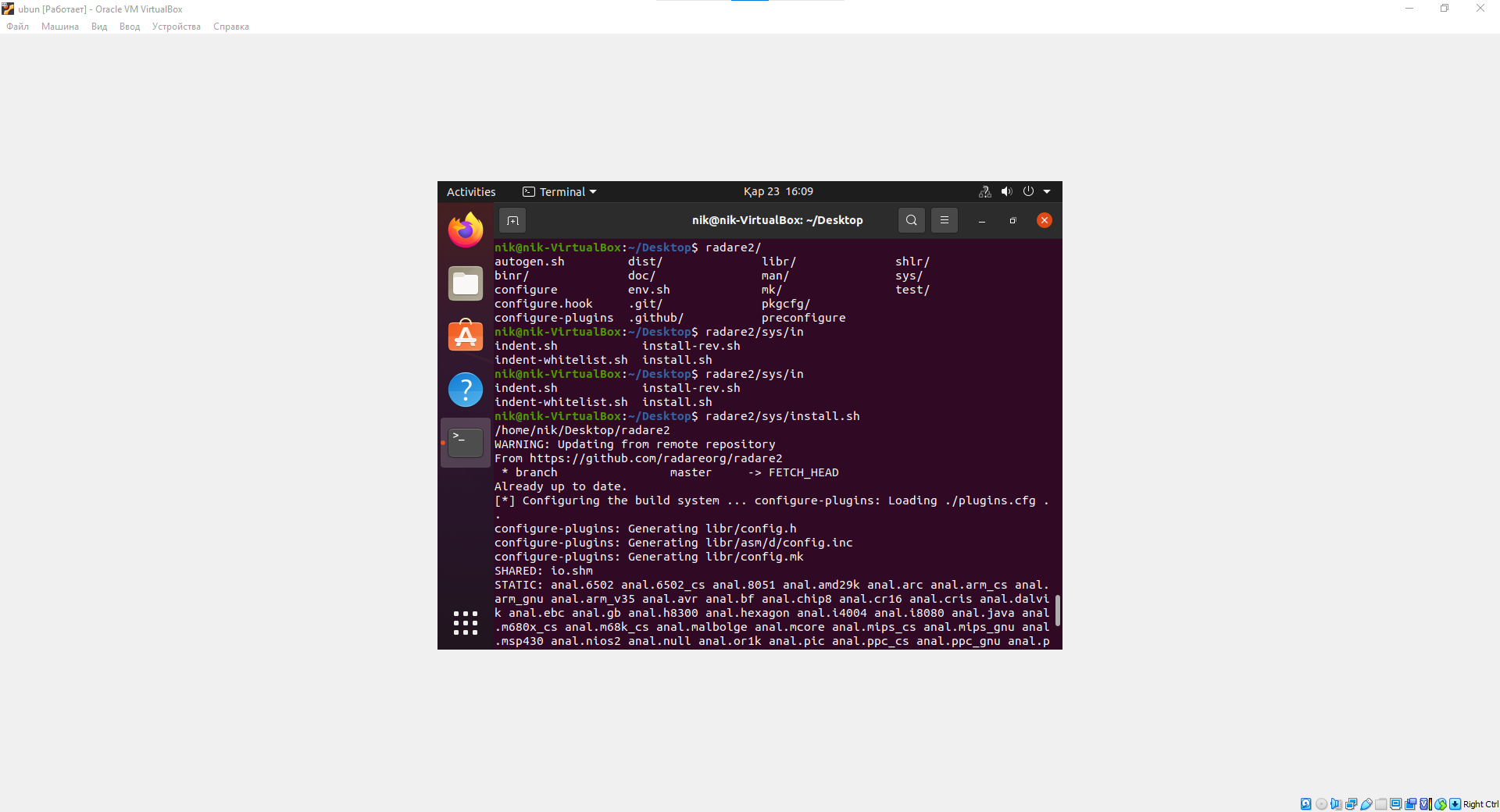


Рис. 26 -Установка radare2

* + 1. Устанавливаем необходимые компоненты для граф.оболочки «Iaito»



Рис. 27 -Установка компонентов

Переход в папку radare2 и открытие файла конфигурации

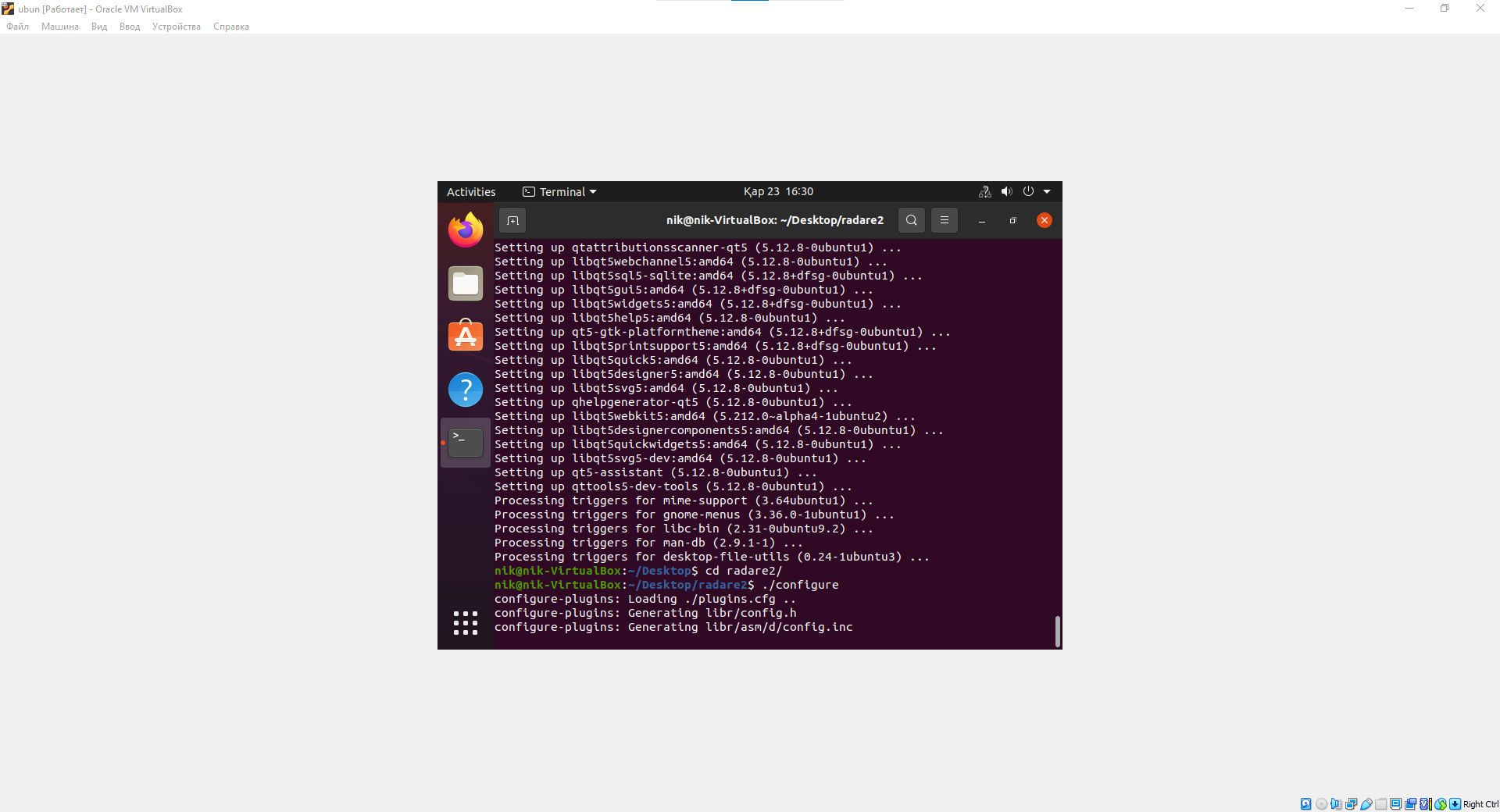


Рис.28-Конфигурация radare2

Распаковка radare2



Рис.29-Распаковка radare2

Установка графического интерфейса Iaito

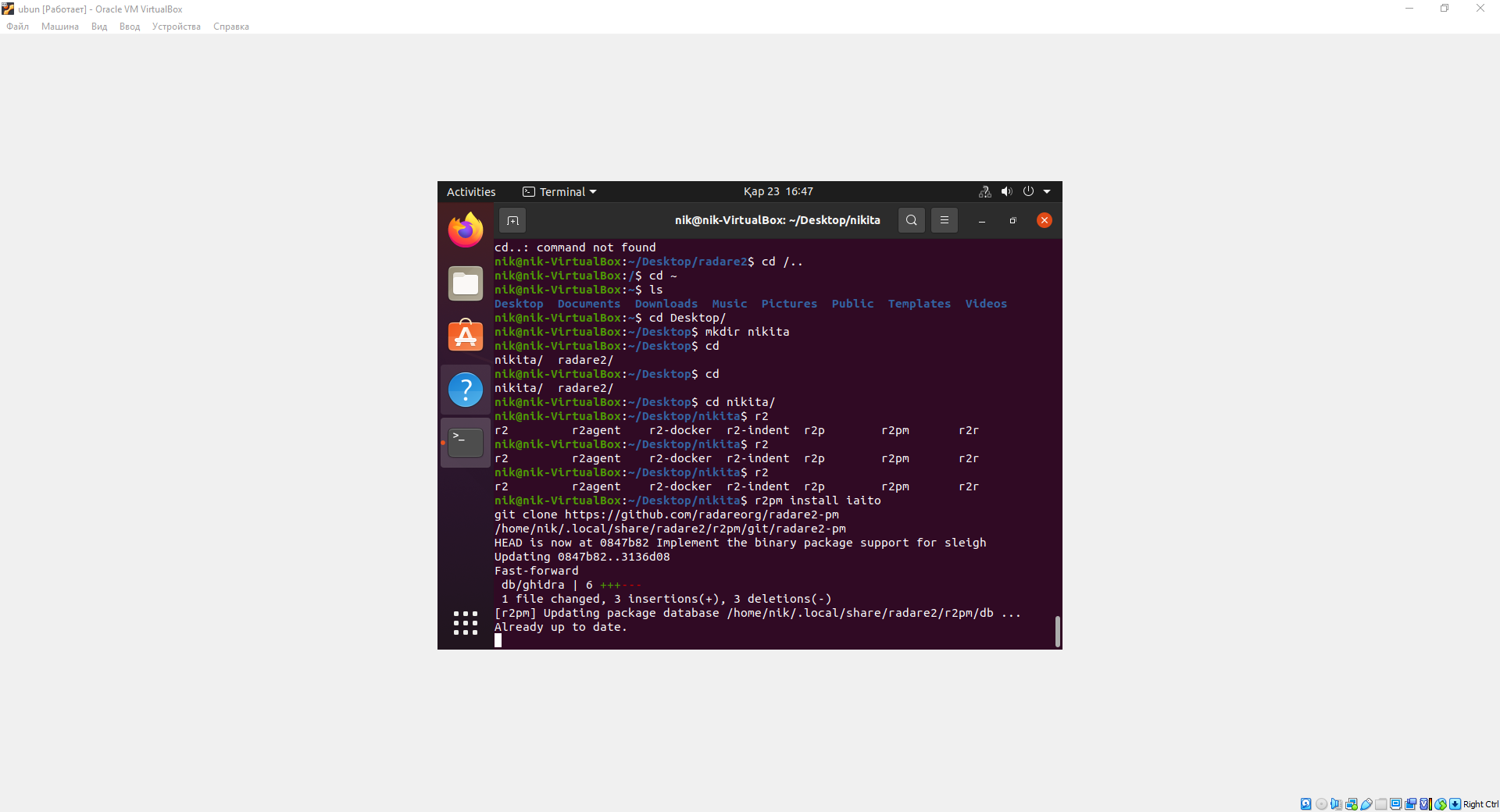


Рис.30-Графический интерфейс iaito

Установка декомпилятора r2ghidra



Рис.31-Декомпилятор r2ghidra

# **6. Дизассемблирование**

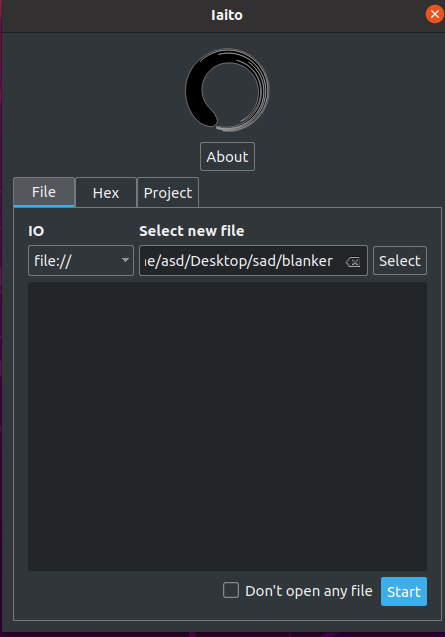


Рис.32-Вход в iaito

Запускаем iaito и переходим во вкладку main

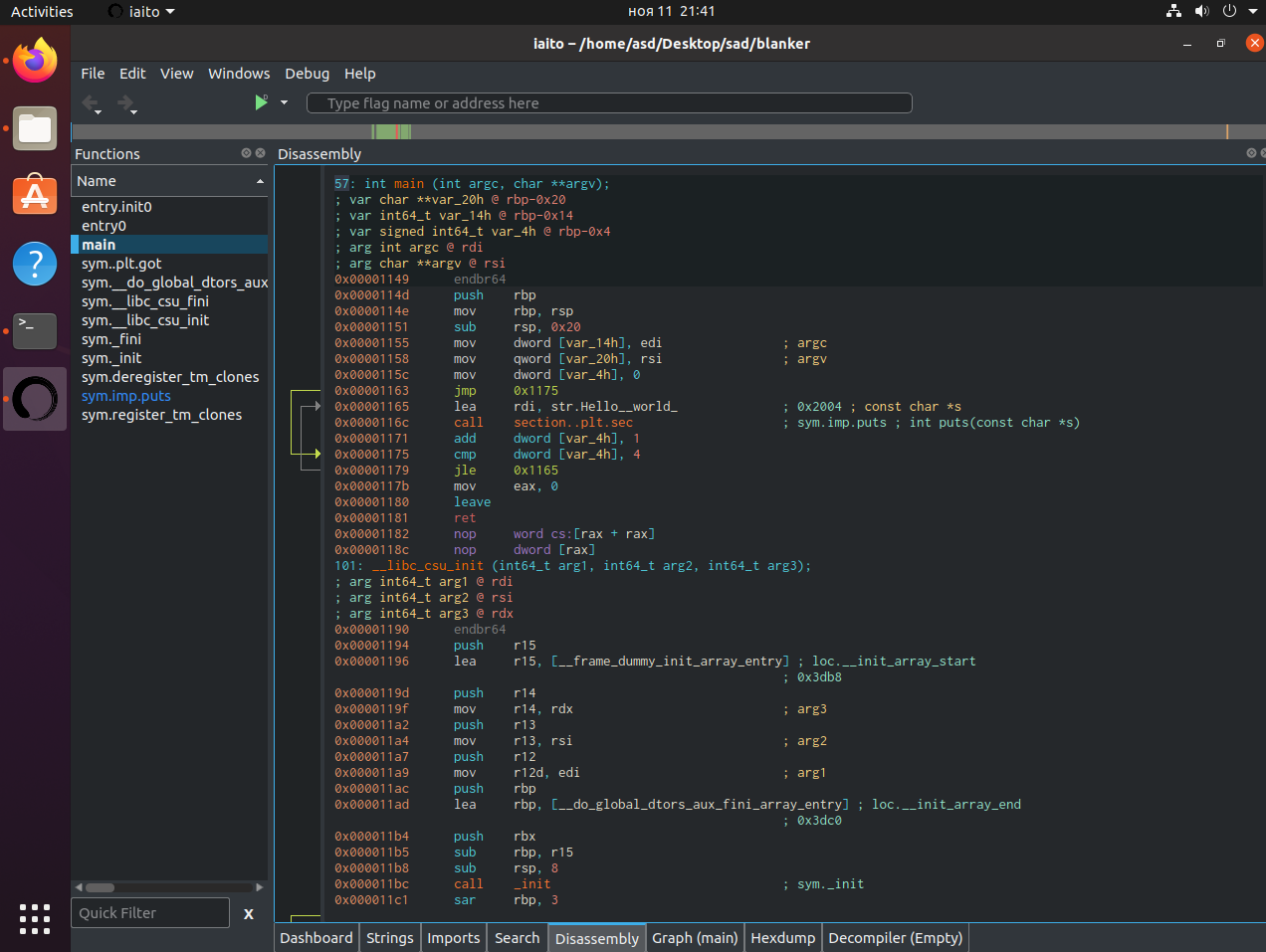


Рис-33.Iaito

Выбор элемента для изменения и его изменение

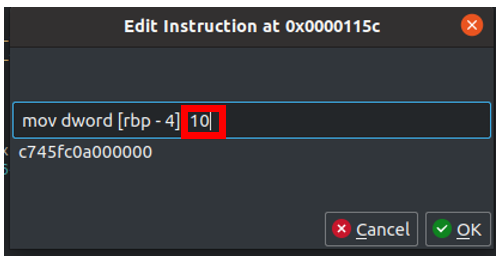


Рис.34-Изменения в коде

Проверка изменений

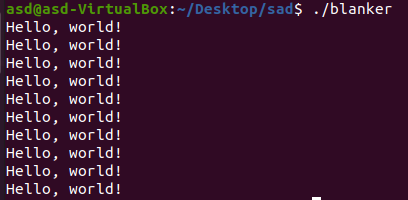


Рис.35-Проверка изменений

Создание файла с кодом для компиляции на рабочем столе



Рис36-Файл для компиляции

# **7. Обратное проектирование**

Файл для компиляции на рабочем столе

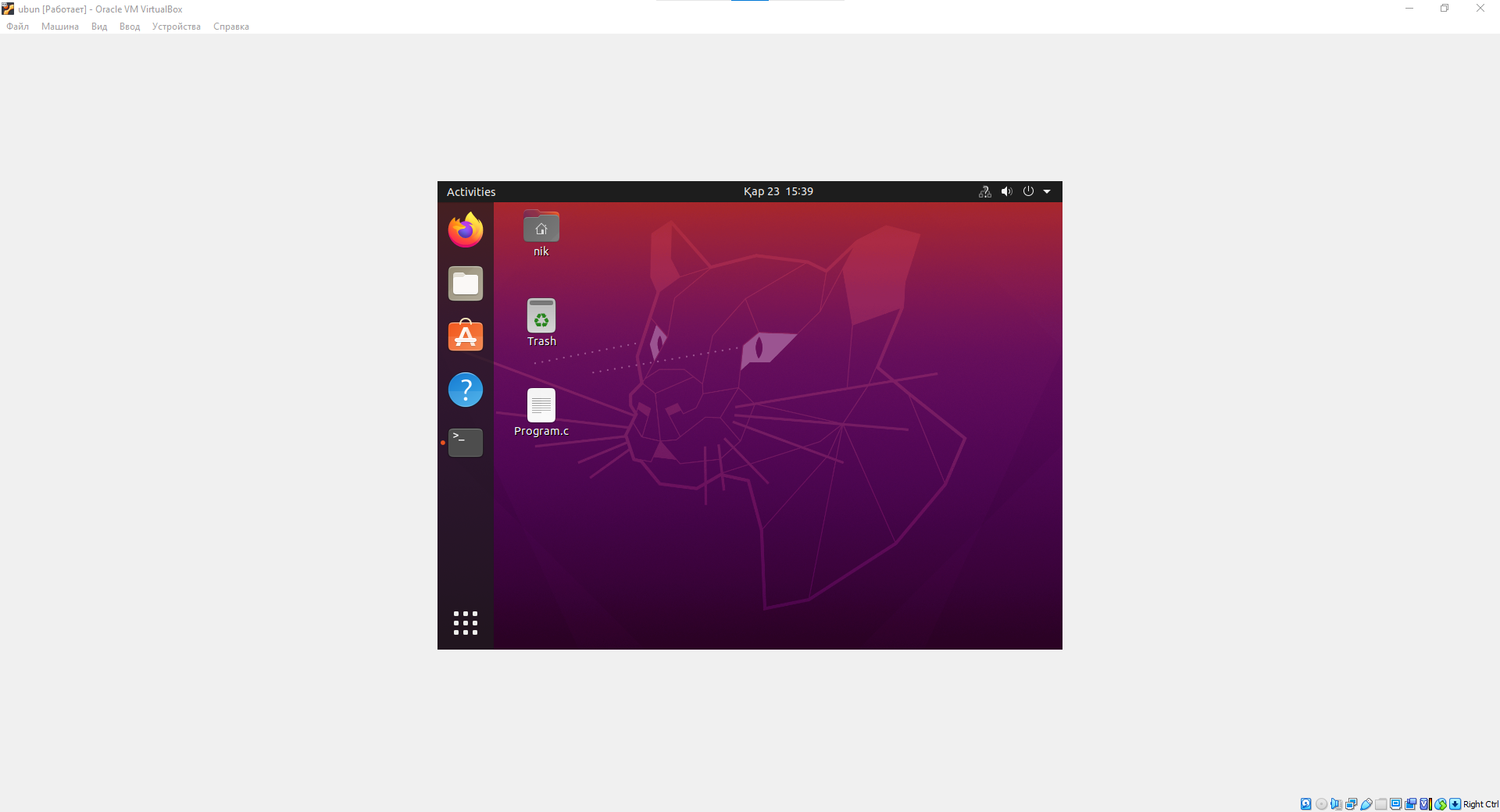
 Р

рис.37-Файл на рабочем столе

Исходный код

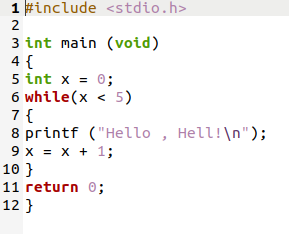


Рис.38-Исходный код

Код в файле для компиляции

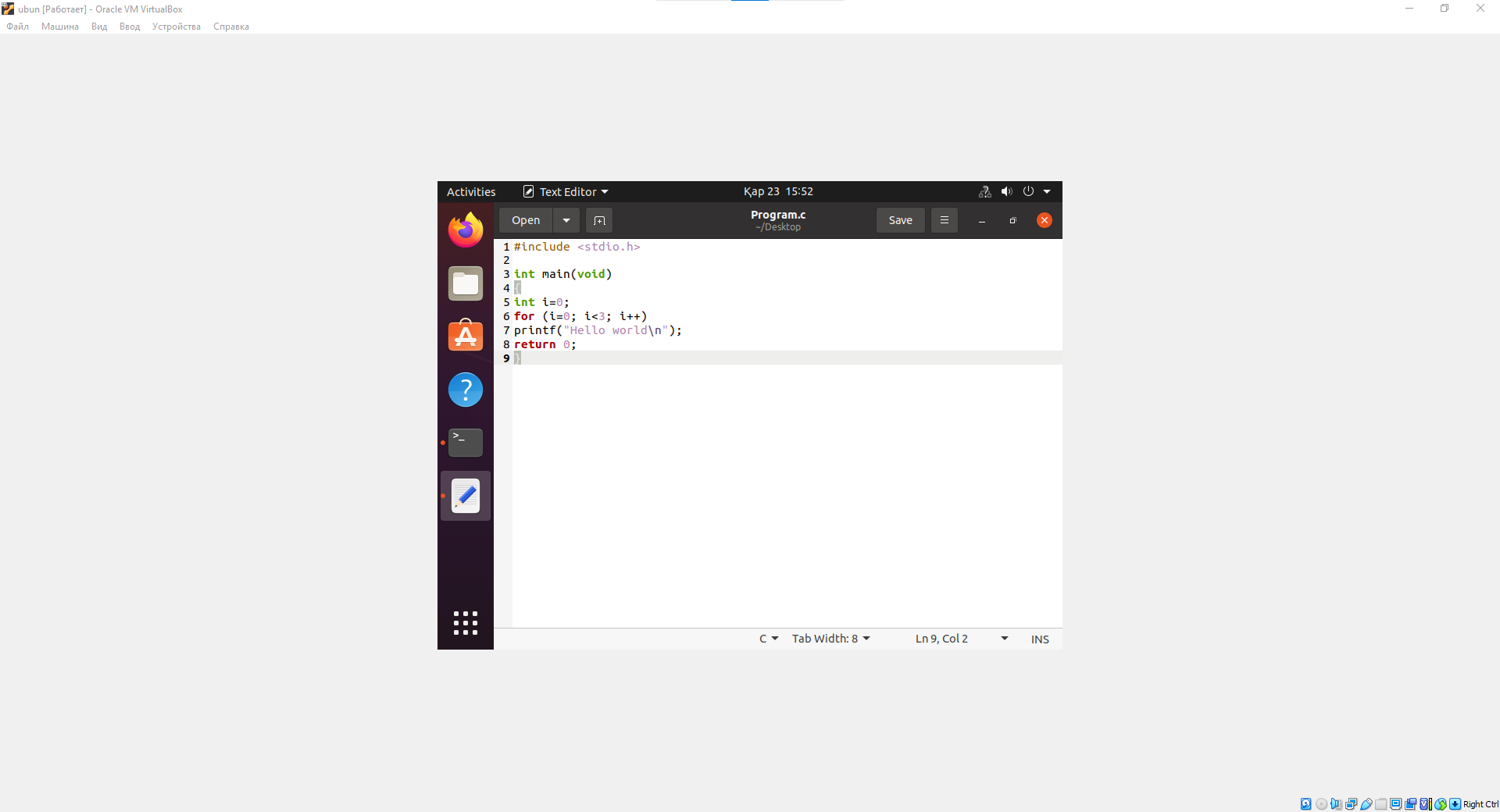


Рис.39-Код в файле

Компиляция файла с кодом



Рис.40-Компиляция файла

Запуск скомпилированного файла



Рис.41-Скомпилированный файл

# **ЗАКЛЮЧЕНИЕ**

Целью учебной практики по ПМ.03 «Ревьюирирование программных продуктов» являлось, научиться выполнять прямое и обратное проектирование ПО, овладеть навыками приостановки и возобновления работы с рабочими задачами иинспекцией кода в Visual Studio, совместно работать над проектом в системе контроля версий GIT.

В ходе выполнения заданий на учебную практику было выполнено:

1. Изучены такие термины как, обратное проектирование, репозиторий, дизассемблирование и т.д.
2. Измерена производительность приложения посредством анализа использования ЦП. Здесь мы узнали, как собирать и анализировать данные об использовании ЦП.
3. Выполнено обратное проектирование.
4. Выполнено дизассемблирование, где было произведено изменение кода с помощью дизассемблера.

**ЛИТЕРАТУРА**

1. (бакалавриат), 38.03.05 (бакалавриат) и 10.05.02 (специалитет) всех профилей подготовки / Юрий Владимирович Ланских ; ВятГУ, ФАВТ, каф. АТ. - Киров: [б. и.], 2015. - 138 с.
2. Золотов, С. Ю. Проектирование информационных систем [Электронный ресурс] / С.Ю. Золотов. - Томск: Эль Контент, 2013. - 88 с.
3. Карпенков, С. Х. Технические средства информационных технологий [Электронный ресурс] / С.Х. Карпенков. - 3-е изд., испр. и доп. - М.|Берлин: Директ-Медиа, 2015. - 376 с.
4. Коноплева, И. А. Информационные технологии [Электронный ресурс] / И.А. Коноплева. - 2-е изд., перераб. и доп. - Москва: Проспект, 2014. - 328 с.
5. Корячко, В. П. Процессы и задачи управления проектами информационных систем [Электронный ресурс] / В.П. Корячко. - Москва: Горячая линия - Телеком, 2014. - 376 с.
6. Проектирование информационных систем. Лекция 1. Презентация [Электронный ресурс]. - Москва: Национальный Открытый Университет «ИНТУИТ», 2014. - 27 с.
7. Советов, Борис Яковлевич. Информационные технологии [Электронный ресурс]: учебник / Б. Я. Советов, В. В. Цехановский. - 6-е изд. - Москва: Юрайт, 2015. - х эл. опт. диск (CD-ROM)
8. Советов, Борис Яковлевич. Информационные технологии [Электронный ресурс]: учебник / Б. Я. Советов, В. В. Цехановский. - 6-е изд. - Москва: Юрайт, 2015. - х эл. опт. диск (CD-ROM)
9. Страбыкин, Дмитрий Алексеевич. Организация ЭВМ: лабораторный практикум на компьютерах: учеб. пособие для студентов направления подготовки 09.03.01 (230100.62) / Д. А. Страбыкин; ВятГУ, ФАВТ, каф. ЭВМ. - 3-е изд., перераб. и доп. - Киров: [б. и.], 2013. - 62 с.