Титульный лист

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

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

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

Дата\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ОТЧЕТ**

По учебной практике

Специальность: 09.02.07 «Информационные системы и программирование»

ПМ.03 «Ревьюирирование программных продуктов»

Выполнил:

Студент группы\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_В.Е.Левин

\_\_\_\_\_\_\_\_\_\_\_дата

2020-2021 уч.г.

План прохождения учебной практики

по ПМ.03 «Ревьюирирование программных продуктов»

Студент Левин Валерий

Группа 1

Специальность Информационные системы и программирование

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

Руководитель практики \_\_\_Ю.В.Майер\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Сроки проведения практики\_\_\_\_22.11.-27.112021\_\_\_\_\_\_\_

Таблицу 1 – План прохождения практики

|  |  |  |
| --- | --- | --- |
| Этапы практики | Дата | Освоен  (да/нет) |
| Измерение производительности приложения посредством анализа использования ЦП | 22.11 |  |
| Установка ПО (Linux, компилятор GCC, radare2, iaito) для обратного проектирования | 23.12 |  |
| Обратное проектирование. | 24.12 |  |
| Дизассемблирование. | 25.12 |  |
| Разработать сопроводительную документацию | 26.12 |  |
| Защита практики | 27.12 |  |

**Отзыв руководителя**

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Группа\_\_\_\_\_\_\_\_\_\_

Специальность \_\_\_\_\_\_\_\_\_\_\_\_\_

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

Руководитель практики \_\_\_Ю.В.Майер\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Сроки проведения практики\_\_\_\_22.11– 27.11 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***

Введение6

Список терминов и определений8

Функциональные требования9

Измерение производительности приложения …………………………....9

Установка ПО ……………………………………………………………..13

Обратное проектирование ………………………………………………..17

Дизассемблирование …………………………………………………..…19

Заключение …………………………………………………………...…...26

Список литературы ……………………………………………………….27

**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

*1. Точка останова* - это преднамеренное прерывание выполнения программы, при котором выполняется вызов отладчика (одновременно с этим программа сама может использовать точки останова для своих нужд). После перехода к отладчику программист может исследовать состояние программы (логи, состояние памяти, регистров процессора, стека и т. п.), с тем чтобы определить, правильно ли ведёт себя программа. В отличие от полной остановки, с помощью останова, после работы в отладчике программа может быть завершена либо продолжена с того же места, где произошёл останов.

*2*. *Репозиторий* (от англ. Repository - хранилище) — место, где хранятся и поддерживаются какие-либо данные. Чаще всего данные в репозитории хранятся в виде файлов, доступных для дальнейшего распространения по сети.

*3.* GitHub - это система управления проектами и версиями кода, а также платформа социальных сетей, созданная для разработчиков.

4. *Radare2* (Также известный как r2) – свободный кроссплатформенный фреймворк для реверс-инжиниринга, написанный на Си, который включает дизассемблер, шестнадцатеричный редактор, анализатор кода и т.п. Используется при реверсе, отладке вредоносного ПО и прошивок

5. *Декомпилятор* — это программа, транслирующая [исполняемый модуль](https://ru.wikipedia.org/w/index.php?title=%D0%98%D1%81%D0%BF%D0%BE%D0%BB%D0%BD%D1%8F%D0%B5%D0%BC%D1%8B%D0%B9_%D0%BC%D0%BE%D0%B4%D1%83%D0%BB%D1%8C&action=edit&redlink=1) (полученный на выходе [компилятора](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80)) в эквивалентный [исходный код](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 программирования](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) высокого уровня.

6. *Декомпиляция* – получение кода языка высокого уровня из программы на машинном языке или ассемблере.

7. *Обратное проектирование (Reverse Engineering)* - исследование некоторого готового устройства или программы, а также документации на него с целью понять принцип его работы; например, чтобы обнаружить недокументированные возможности (в том числе программные закладки), сделать изменение или воспроизвести устройство, программу или иной объект

8. *Дизассемблирование* **–** преобразованиепрограммы на машинном языке к ее ассемблерному представлению.

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

**Функциональный требования**

В ходе работы было установлено:

1. Компилятор и GCC компоненты

2. Пакеты, необходимые для работы с Radare2

3. Сам Radare2

4. Компоненты для работы с программой Iaito

5. Установка самой Iaito

6. Декомпилятор r2ghidra

7. Visual Studio

8. OC Linux – Ubuntu

**Измерение производительности приложения посредством анализа использования ЦП**

**Шаг 1. Сбор данных профилирования**

1. Откройте проект для отладки в Visual Studio и установите точку останова в приложении в точке, где вы хотите проверить загрузку ЦП.
2. Установите вторую точку останова в конце функции или области кода, который требуется проанализировать.

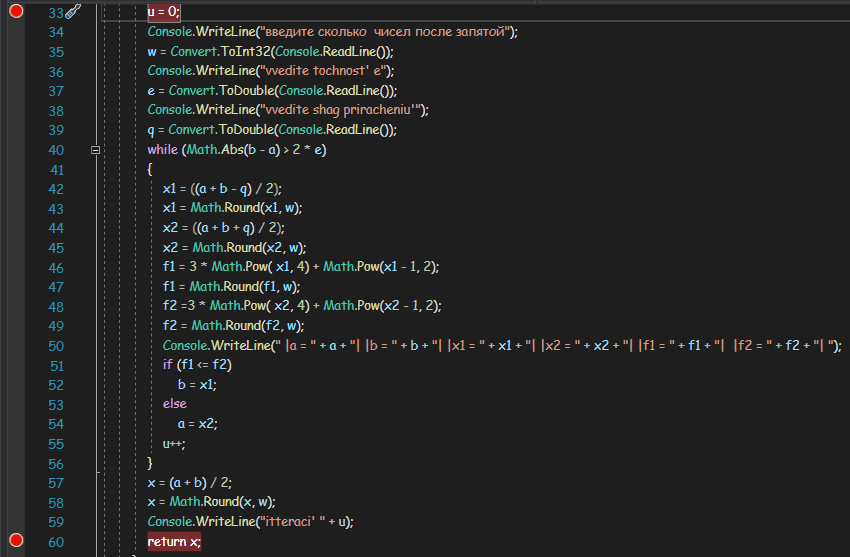


Рис.1 Точки останова

1. Открываем средства диагностики (Отладка -> Окна -> Показать средства диагностики)

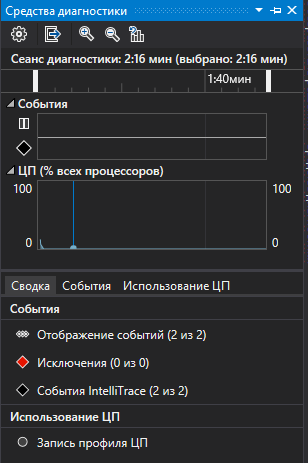


Рис.2 Средства диагностики

1. Мы можем выбрать, что посмотреть – использование памяти или использование ЦП, но нам необходимо выбрать Использование ЦП.

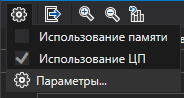


Рис.3 Параметры средств диагностики

1. Далее щелкаем «Начать отладку»
2. Сценарий, который вызвал срабатывание первой точки останова



Рис.4 Срабатывание первой точки останова

1. Делаем запись профиля ЦП

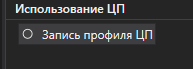


Рис.5 Запись профиля ЦП

1. Нажимаем F5 и идем по точкам останова, и смотрим результат записи

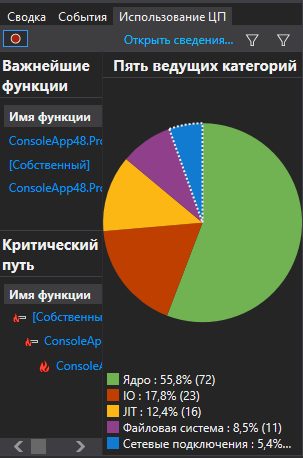


Рис.6 Результат записи

1. Также можем выбрать определенную область

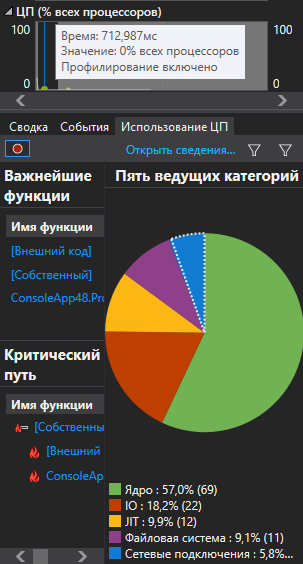


Рис.7 Выбор области

**Шаг 2. Анализ данных о загрузке ЦП**

1. Изучаем функции, которые выполняют большую часть работы

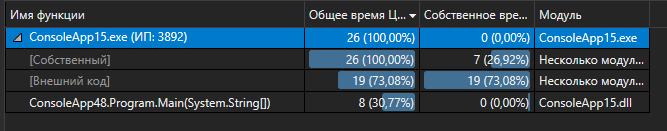
****

Рис.8 Функции

1. Открываем вызывающие/вызываемые функции

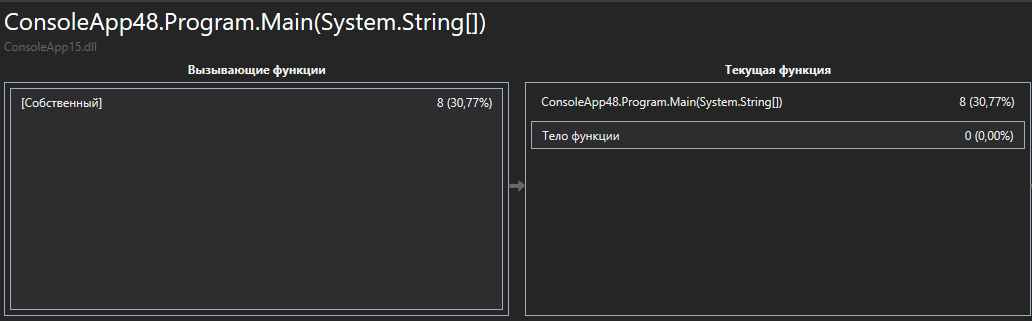


Рис.9 Вызывающие/вызываемые функции

1. Далее открываем дерево вызовов

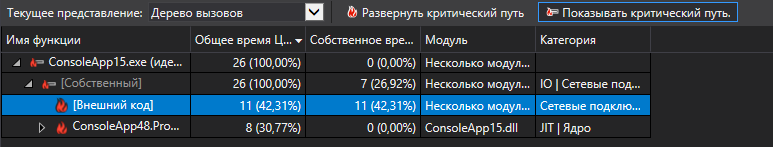


Рис.10 Дерево вызовов

Репозиторий проекта - https://github.com/Barsuk22/MDK-03.02

**Установка ПО**

1. Устанавливаем компилятор GCC компоненты



Рис. 11

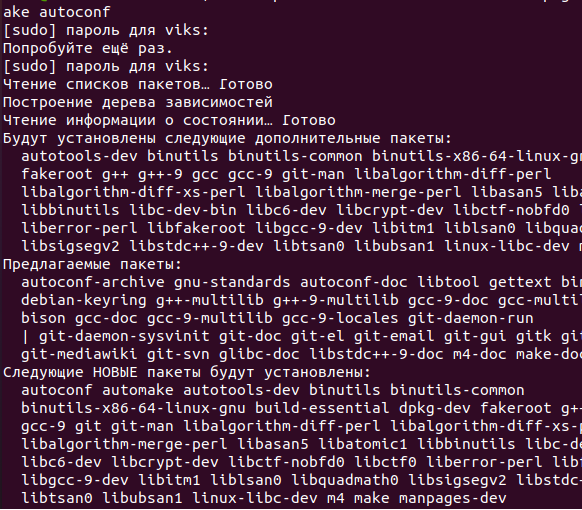


Рис. 12

1. Устанавливаем необходимые пакеты для работы с radare2

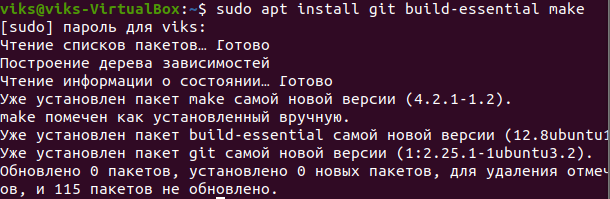


Рис. 13

1. Установка radare2:

git clone https://github.com/radareorg/radare2

radare2/sys/install.sh

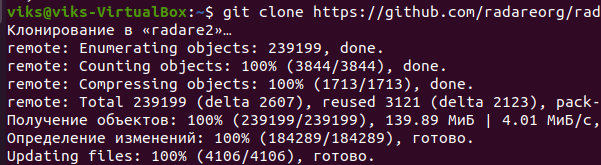


Рис. 14

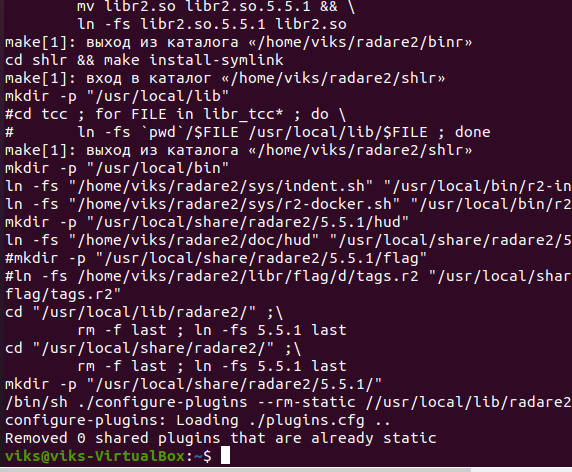


Рис. 15

3. Устанавливаем необходимые компоненты для граф.оболочки «Iaito»

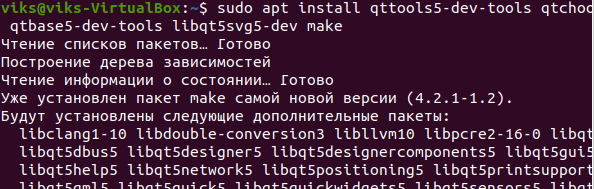


Рис. 16

4. Устанавливаем графический интерфейс «iaito»



Рис. 17

5. Устанавливаем декомпилятор «r2ghidra» 

Рис. 18

**Обратное проектирование**

1. Устанавливаем компилятор GCC компоненты



Рис. 19

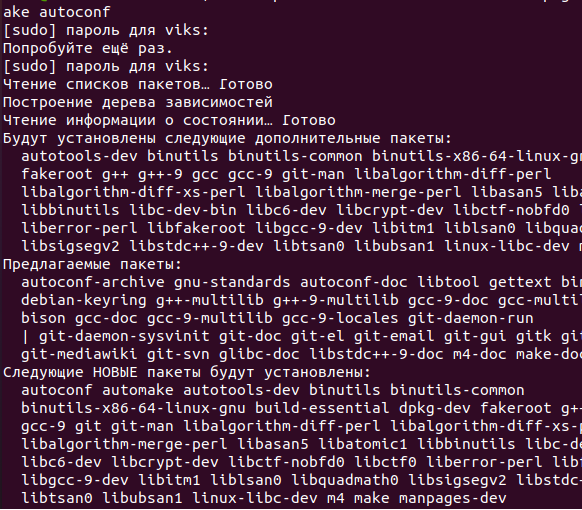


Рис. 20

1. Создаем файл



Рис. 21

1. Вставляем туда код

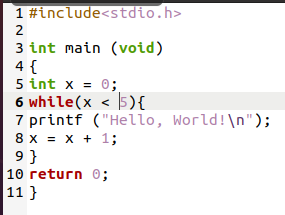


Рис. 22

1. Компилируем



Рис. 23

1. Открываем и проверяем наш файл

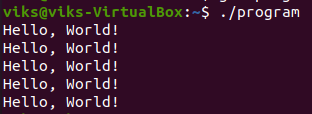


Рис. 24

**Дизассемблирование**

1. Устанавливаем необходимые пакеты для работы с radare2

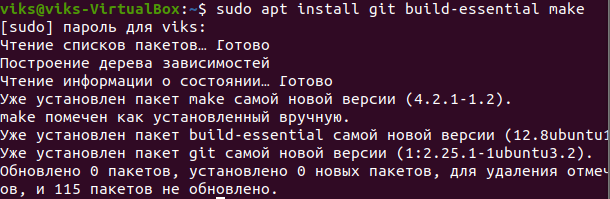


Рис. 25

1. Установка radare2:

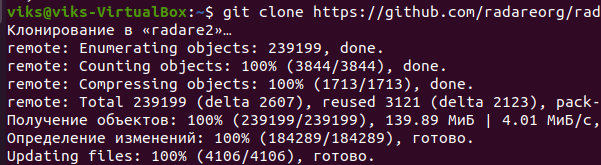


Рис. 26

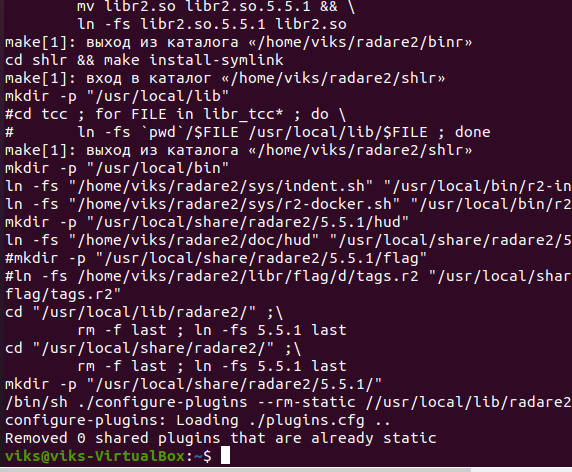


Рис. 27

1. Проверяем и обновляем radare2

Рис. 28

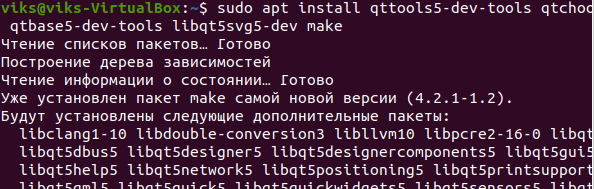
1. Устанавливаем необходимые компоненты для граф.оболочки «Iaito» 

Рис. 29

1. Открываем папку и открываем файл



Рис. 30

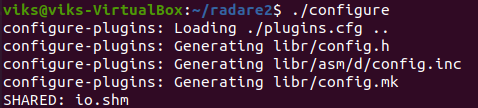


Рис. 31

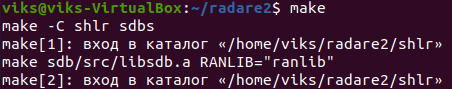


Рис. 32

1. Повторить

Устанавливаем графический интерфейс «iaito»



Рис. 33

1. Устанавливаем декомпилятор «r2ghidra» 

Рис. 34

1. Запускаем «Iaito»

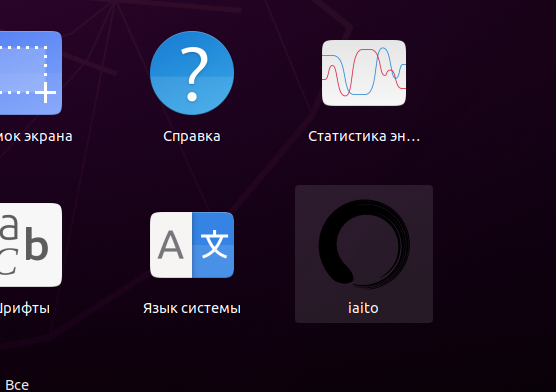


Рис. 35

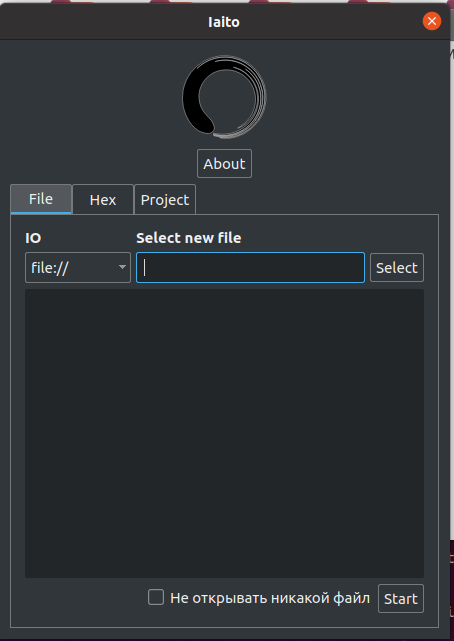


Рис. 36

1. Проверяем тестовый файл

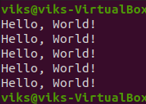


Рис. 37

1. Открываем тестовый файл в «radare2». Настройки оставляем стандартные

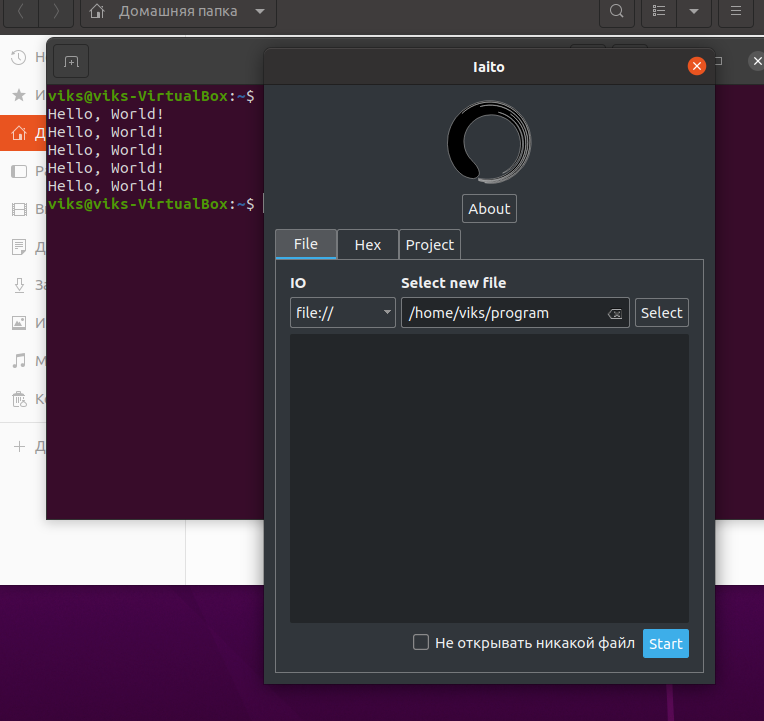


Рис. 38

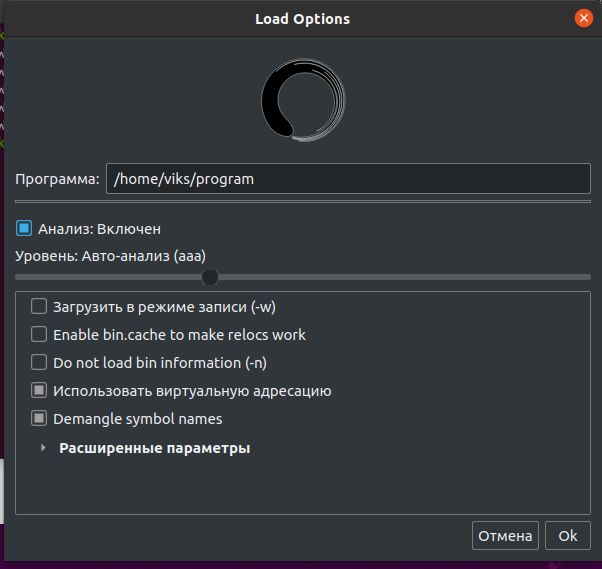


Рис. 39

1. Открываем слева «main»

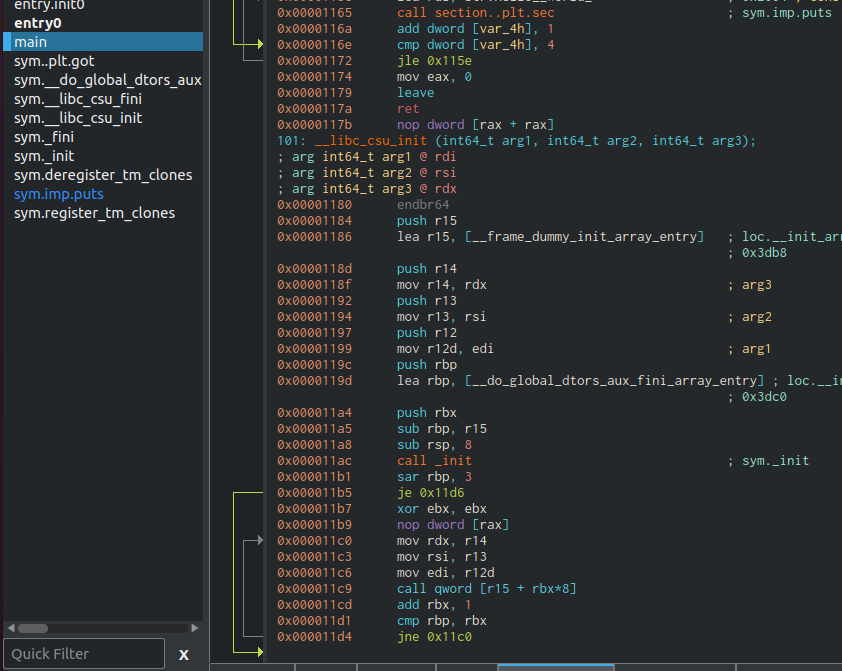


Рис. 40

1. Открываем вкладку декомпилятора. Видим, что код практически читаем, кроме переменных

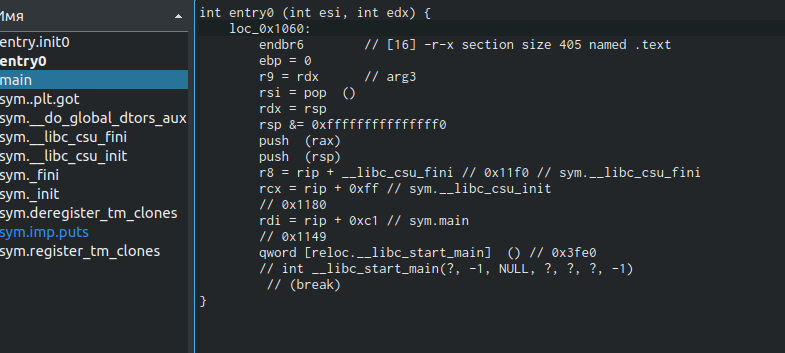


Рис. 41

1. Меняем режим работы программы (File > Set mode > Cache mode)

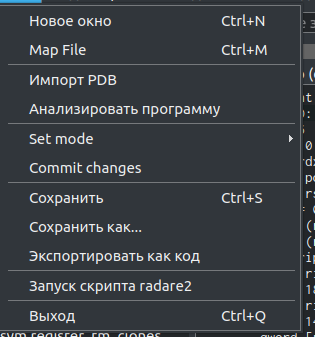


Рис. 42

1. Так как фраза «Hello world!» выводится 5 раз, а нам необходимо увеличить вывод до 10 раз, то смотря на код, можно увидеть, что используется цикл «while» по значение переменной [var4\_h] равной 4 (т.к While <=4, а цикл начинается с 0). Выделяем цифру 4 и нажимаем «Edit» меняя значение переменное с 4 до 10.



Рис. 43

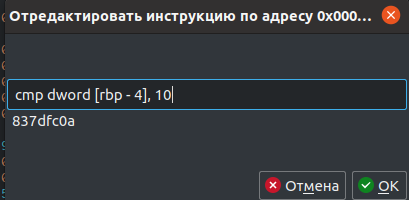


Рис. 44

1. Сохраняем наши изменения, нажав на кнопку «Commit changes»

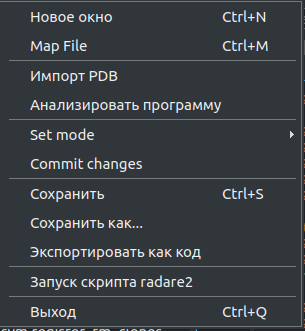


Рис. 45

1. Запускаем наш файл для проверки

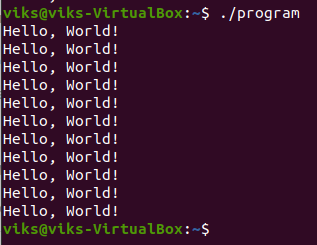


Рис. 46

**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 измерение производительности приложения посредством анализа использования ЦП. В ходе данной работы было выяснено как работать с точками останова, работать с ЦП.

В ходе выполнения заданий на учебную практику была выполнена установка ПО в Linux, что научило меня работать с различным программным обеспечением из консольного приложения.

Также было выполнено обратное проектирование. Необходимо было создавать и работать с исходным файлом.

Дизассемблирование – работа с radare2, Iaito.

**ЛИТЕРАТУРА**

1. 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.
2. 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)
3. 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)
4. Проектирование информационных систем. Лекция 1. Презентация [Электронный ресурс]. - Москва: Национальный Открытый Университет «ИНТУИТ», 2014. - 27 с.
5. Ланских, Юрий Владимирович Предметно-ориентированные информационные системы [Электронный ресурс] : учеб. пособие для студентов направления 09.03.02, 10.03.01,
6. Корячко, В. П. Процессы и задачи управления проектами информационных систем [Электронный ресурс] / В.П. Корячко. - Москва: Горячая линия - Телеком, 2014. - 376 с.
7. Коноплева, И. А. Информационные технологии [Электронный ресурс] / И.А. Коноплева. - 2-е изд., перераб. и доп. - Москва: Проспект, 2014. - 328 с.
8. 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.
9. 3.Золотов, С. Ю. Проектирование информационных систем [Электронный ресурс] / С.Ю. Золотов. - Томск: Эль Контент, 2013. - 88 с.
10. (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.