

Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №1

Менеджер виртуальных машин “Oracle VirtualBox”

Выполнил студент

группы ПИ-21-1

Шишкина Алина

Преподаватель:

Кургасов В.В

Цели работы:

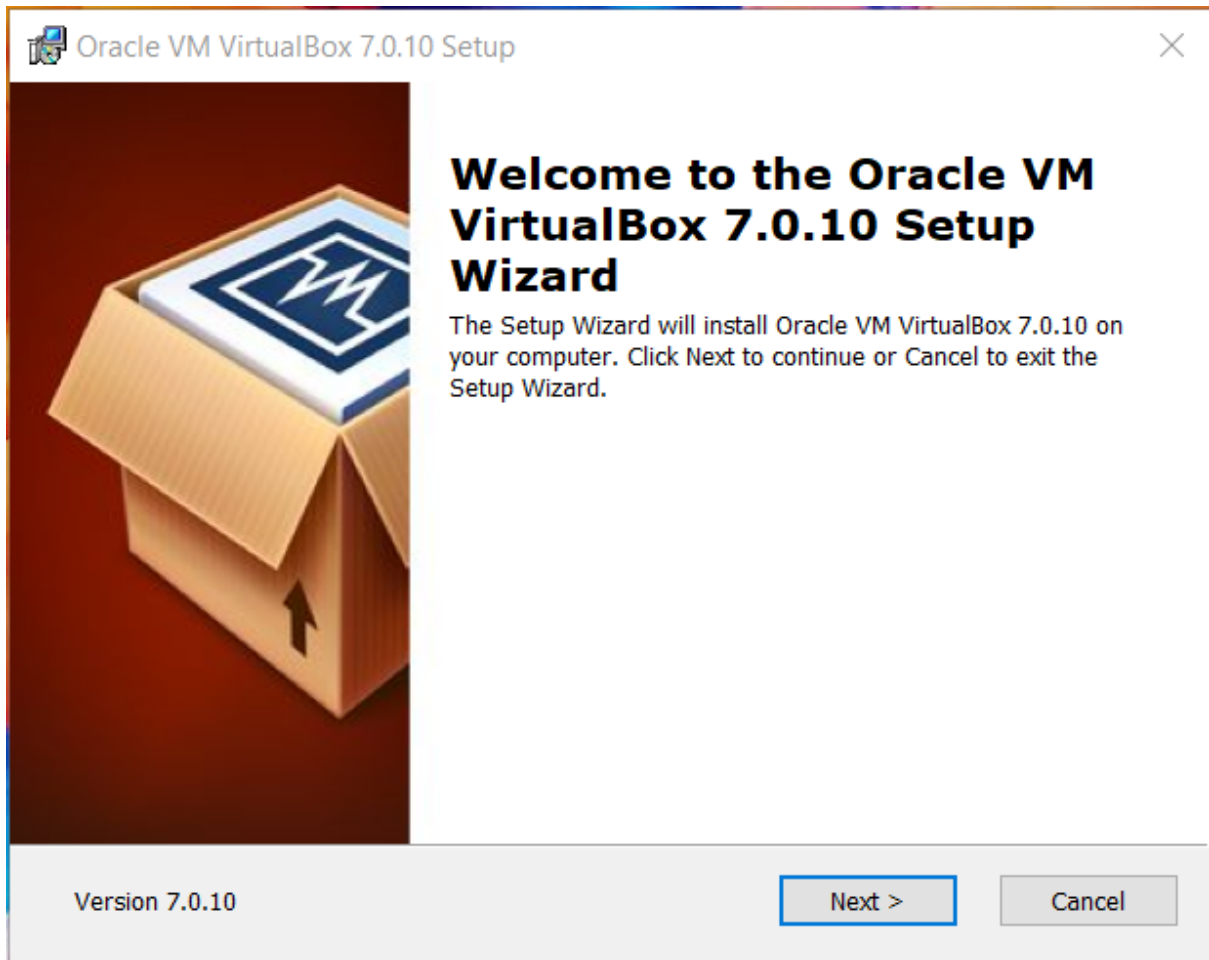
1. Изучить состав инструментальных средств создания и сопровождения аппаратных конфигураций виртуальных машин.
2. Получить практические навыки выполнения типовых операций мониторинга и управления состояниями виртуальных машин.
3. Изучить функциональные возможности интеграции виртуальных (гостевых) и физической (хостовой) машин.
4. Изучить компиляцию и компоновку программ на примере инструментального средства разработки программ GCC (<http://gcc.gnu.org>).
5. Изучить сборку проекта с использованием make (<http://www.gnu.org/software/make/>).
6. Изучить возможности отладки программ с использованием GDB (<http://www.gnu.org/s/gdb/>).
7. Ознакомиться с интегрированными средами разработки и используемыми редакторами.
8. Установить (если отсутствует в системе) редактор Vim. Провести его настройку, включая установку не менее 5 плагинов.
9. Напишите программу в соответствии с назначенным вариантом, используя редактор Vim.
10. Продемонстрируйте работу отладчика на примере написанной вами программы.

Вариант программы - номер 3.

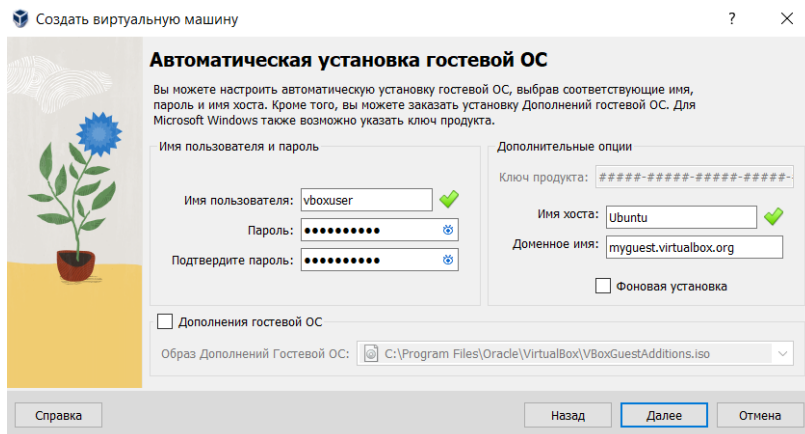
Ход работы:

1. Создание виртуальной машины

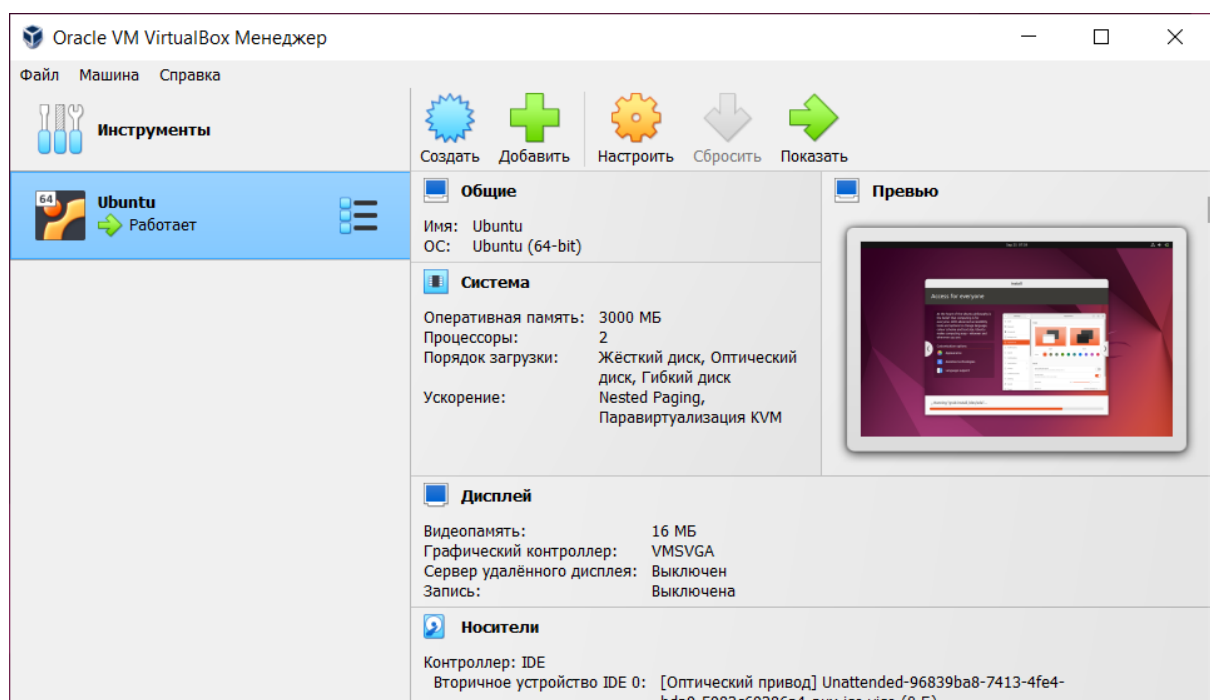
Установка менеджера виртуальных машин VirtualBox осуществляется через установку загрузочного файла, который можно найти на официальном сайте: [Oracle VM VirtualBox](https://www.oracle.com/uk/technetwork/topics/vm-virtualbox-088277.html).



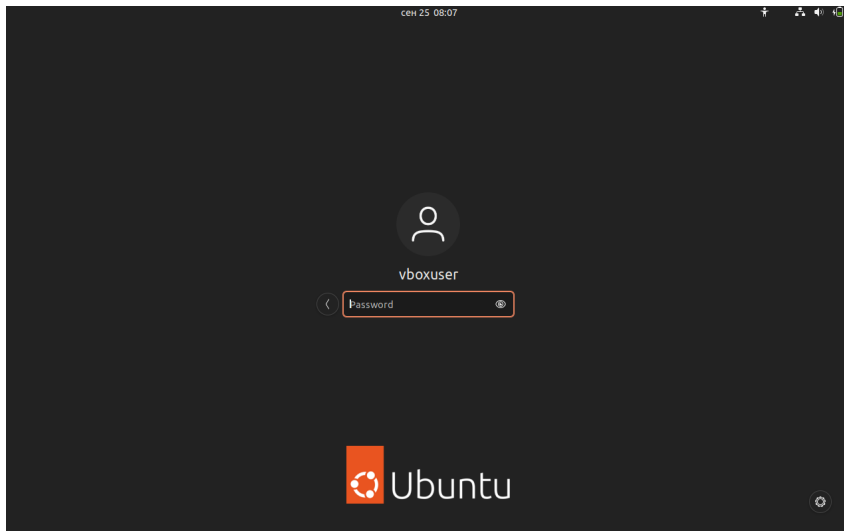
Основная задача программы VirtualBox - создание виртуальных (гостевых) машин, позволяющие совершать вход в другую ОС. Для того, чтобы создать виртуальную машину нужно открыть менеджер, найти кнопку *Создать*, при нажатии запускается мастер создания гостевых машин.



Далее записываем необходимые параметры, выбираем нужный ISO-образ диска (Ubuntu) и запускаем процесс.

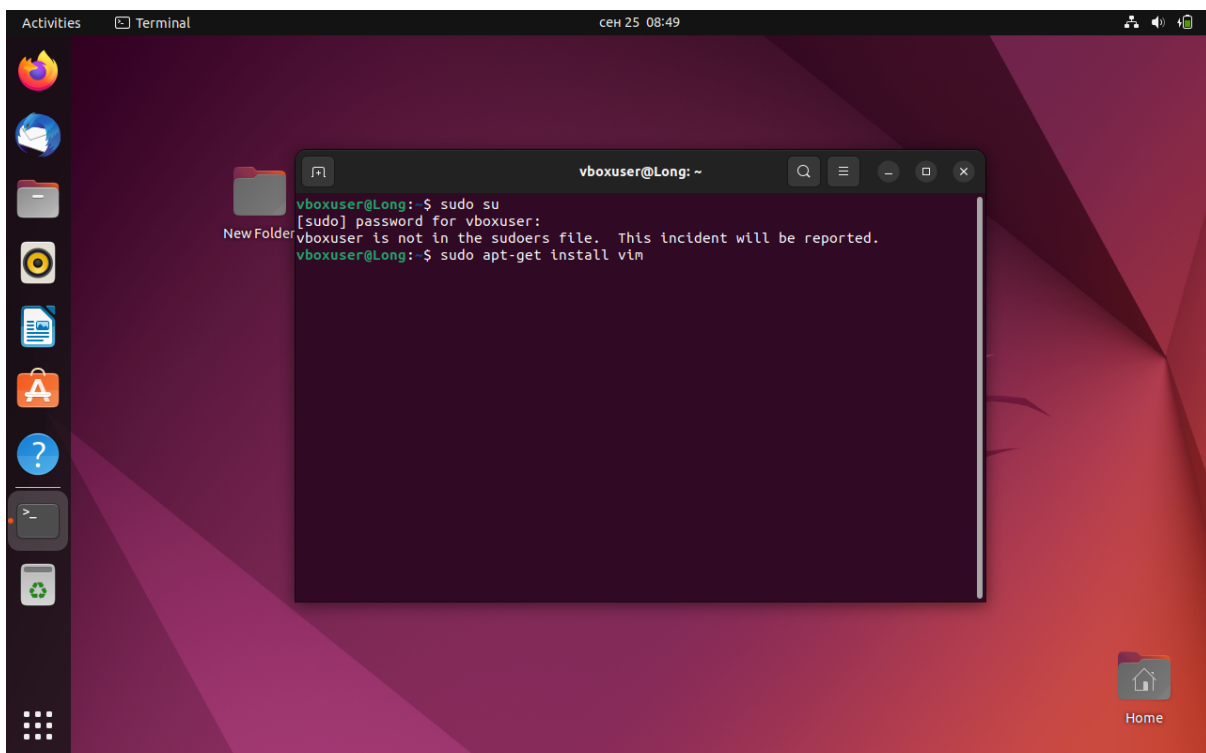


После установки операционной системы сразу видим пользователя, необходимо ввести пароль, чтобы войти в систему.



2. Установка редактора Vim

Vim – это модификация текстового редактора Vi. Это бесплатный кроссплатформенный текстовый редактор с открытым исходным кодом. Установка данного редактора в виртуальной машине представлена на рисунке ниже.



3. Работа с Vim и создание программы

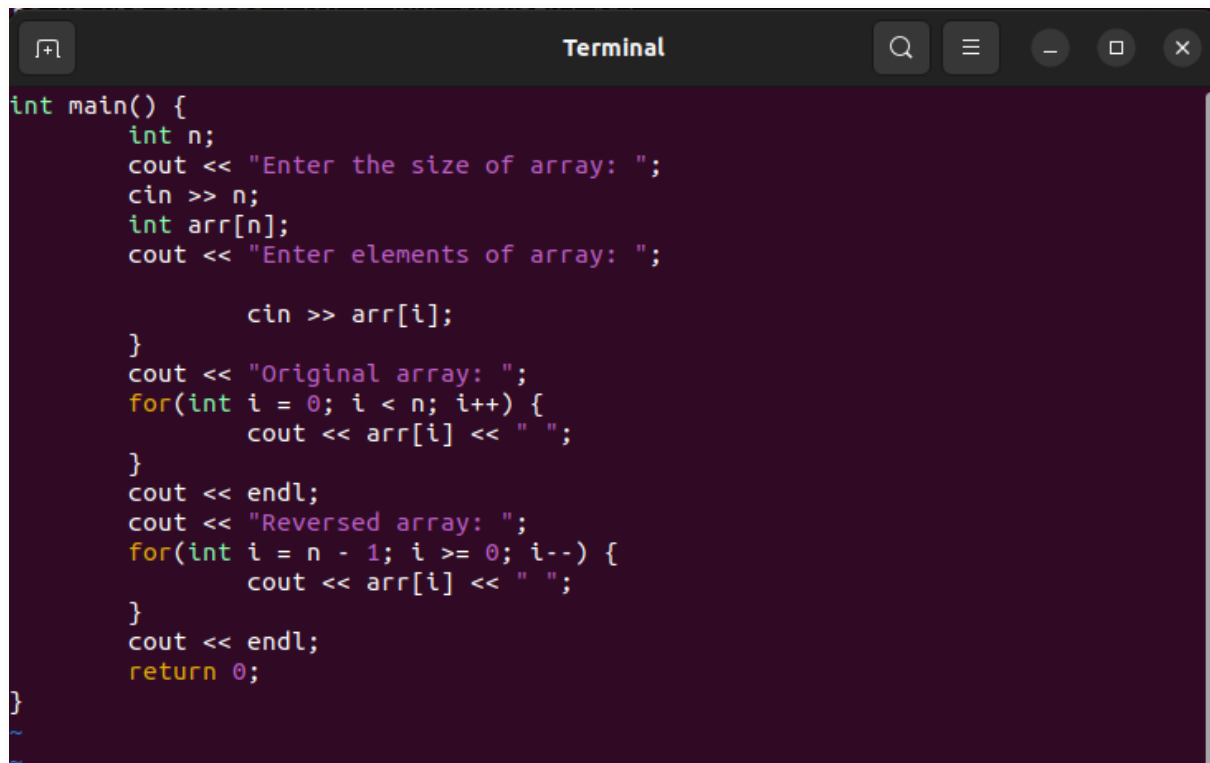
По заданию необходимо создать файл с расширением .cpp, написать код, решающий задачу по варианту, а также необходимо установить плагины для данного редактора.

а) Программная реализация задачи

Задача:

поменять порядок следования элементов массива на обратный

Код решения задачи приведен ниже:

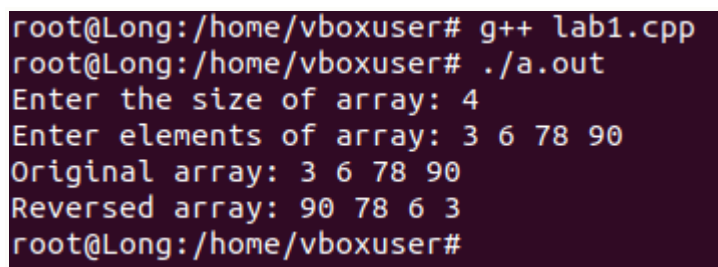
A screenshot of a terminal window with a dark background. The title bar says "Terminal". The code is written in C++ and is as follows:

```
int main() {
    int n;
    cout << "Enter the size of array: ";
    cin >> n;
    int arr[n];
    cout << "Enter elements of array: ";

    cin >> arr[i];
}
cout << "Original array: ";
for(int i = 0; i < n; i++) {
    cout << arr[i] << " ";
}
cout << endl;
cout << "Reversed array: ";
for(int i = n - 1; i >= 0; i--) {
    cout << arr[i] << " ";
}
cout << endl;
return 0;
}
```

Для того, чтобы сохранить файл, необходимо нажать Esc, затем написать :wq. Таким образом команда запишет файл (сохранит) и выйдет из редактора.

Компиляция и последующий результат программы представлен на рисунке

A screenshot of a terminal window showing the compilation and execution of the program. The output is as follows:

```
root@Long:/home/vboxuser# g++ lab1.cpp
root@Long:/home/vboxuser# ./a.out
Enter the size of array: 4
Enter elements of array: 3 6 78 90
Original array: 3 6 78 90
Reversed array: 90 78 6 3
root@Long:/home/vboxuser#
```

б) Установка плагинов

```
curl -fLo ~/.vim/autoload/plug.vim --create-dirs \
```

Эта команда позволяет установить vim-plug, который управляет плагинами.

```

root@Long: /home/vboxuser
root@Long: /home/vboxuser
root@Long: /home/vboxuser
call plug#begin()
    Plug 'pope/vim-dotenv'

    " Short .editorconfig file
    Plug 'editorconfig/editorconfig-vim'

    " Fast motions using f F like behavior
    Plug 'easymotion/vim-easymotion'

    " Outline 'preservim/tagbar'
    Plug 'majutsushi/tagbar', { 'on': 'TagbarToggle' }

    Plug 'luochengxu/vim-which-key'

    " Fuzzy finder
    Plug 'junegunn/fzf', { 'do': { -> fzf#install() } }
    Plug 'junegunn/fzf.vim'

    " Project root
    Plug 'dbakker/vim-projectroot'

    " Comment/uncomment blocks and selections
    Plug 'preservim/nerdcommenter'

    " More informative status bar
    Plug 'vim-airline/vim-airline'
    Plug 'vim-airline/vim-airline-themes'

    " Sonokai full color scheme
    Plug 'sainnhe/sonokai'

call plug#end()

```

~/.vimrc[+] vim utf-8[unix] 38% ln:13/34 361 [1] trailing

Плагин, позволяющий импортировать переменные и их значения из файла `.env` расположенного по тому же пути что и открываемый буфер. Полезно для некоторых плагинов, которые затем могут использовать их для инициализации.

Импорт универсальных настроек редактора <http://editorconfig.org/>: отступы, символы конца строк, кодировка и тому прочее. Устанавливает индивидуальные настройки редактора для проекта или директории.

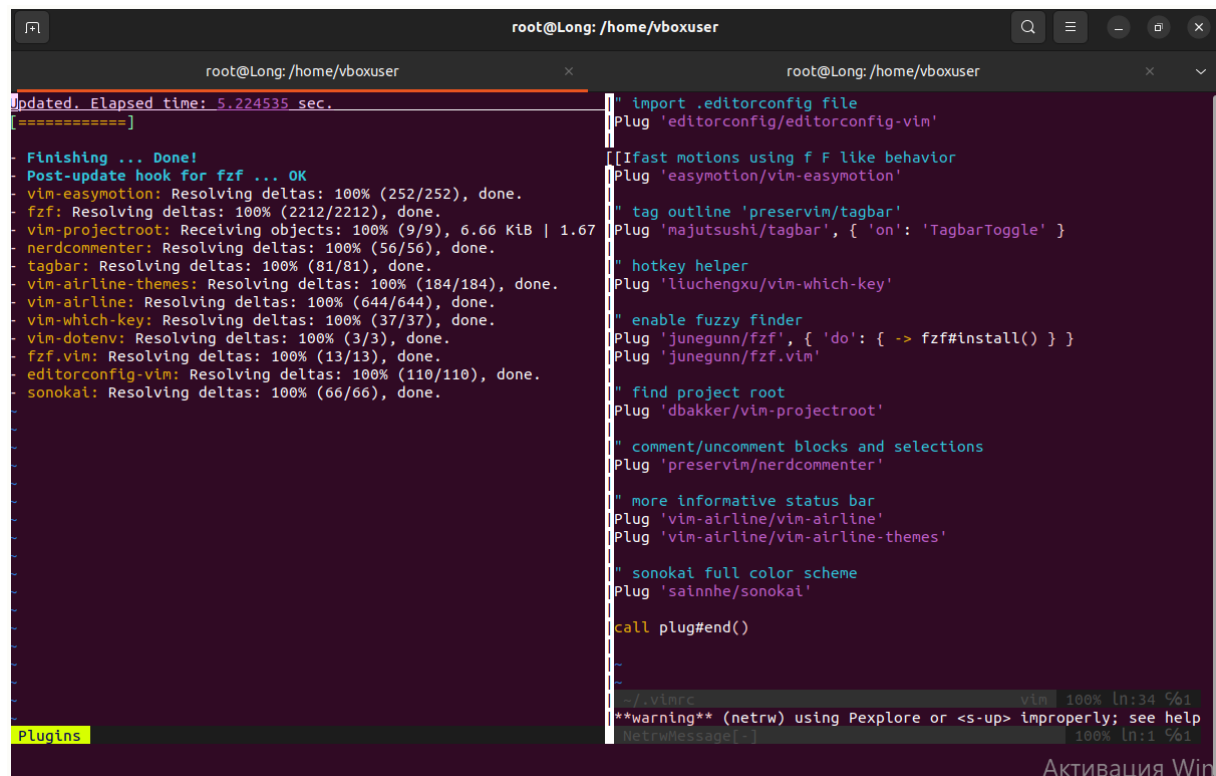
Он нужен для быстрого перемещения по документу.

tagbar

Теги - одно из универсальных понятий для всего что связано с исходным кодом. Плагин с говорящим названием просто отображает эти самые теги сбоку в виде списка в отдельном окне, позволяя передвигаться по ним.

Установка плагинов осуществляется командой `:PlugInstall`

Результат выполнения команды



```
root@Long: /home/vboxuser
Updated. Elapsed time: 5.224535 sec.
=====]
- Finishing ... Done!
- Post-update hook for fzf ... OK
vim-easymotion: Resolving deltas: 100% (252/252), done.
fzf: Resolving deltas: 100% (2212/2212), done.
vim-projectroot: Receiving objects: 100% (9/9), 6.66 KiB | 1.67
nerdcommenter: Resolving deltas: 100% (56/56), done.
tagbar: Resolving deltas: 100% (81/81), done.
vim-airline-themes: Resolving deltas: 100% (184/184), done.
vim-airline: Resolving deltas: 100% (644/644), done.
vim-which-key: Resolving deltas: 100% (37/37), done.
vim-dotenv: Resolving deltas: 100% (3/3), done.
fzf.vim: Resolving deltas: 100% (13/13), done.
editorconfig-vim: Resolving deltas: 100% (110/110), done.
sonokai: Resolving deltas: 100% (66/66), done.

" import .editorconfig file
Plug 'editorconfig/editorconfig-vim'

[[Ifast motions using f F like behavior
Plug 'easymotion/vim-easymotion'

" tag outline 'preservim/tagbar'
Plug 'majutsushi/tagbar', { 'on': 'TagbarToggle' }

" hotkey helper
Plug 'liuchengxu/vim-which-key'

" enable fuzzy finder
Plug 'junegunn/fzf', { 'do': { -> fzf#install() } }
Plug 'junegunn/fzf.vim'

" find project root
Plug 'dbakker/vim-projectroot'

" comment/uncomment blocks and selections
Plug 'preservim/nerdcommenter'

" more informative status bar
Plug 'vim-airline/vim-airline'
Plug 'vim-airline/vim-airline-themes'

" sonokai full color scheme
Plug 'sainnhe/sonokai'

call plug#end()

~/vimrc vim 100% ln:34 %1
**warning** (netrw) using Pexlore or <s-u> improperly; see help
netrwmessage[-] 100% ln:1 %1

Plugins
Активация Win
```

3. IDE: сравнение характеристик

В данной работе будут сравниваться следующие редакторы кода: Code::Blocks и JetBrains CLion

1. Интерфейс и удобство использования:

Code::Blocks имеет простой и интуитивно понятный интерфейс, который легко освоить даже начинающим программистам. Он предлагает широкий набор инструментов и настраиваемых опций. CLion, с другой стороны, имеет более современный и эргономичный интерфейс, который может быть более привлекательным для опытных разработчиков.

2. Поддержка языков программирования:

Code::Blocks предназначен преимущественно для разработки на C и C++, но также поддерживает другие языки, такие как Java и Python. CLion специализируется на разработке на C и C++, и предлагает более глубокую интеграцию с этими языками.

3. Функциональность и возможности:

Code::Blocks предлагает базовый набор функций, таких как автодополнение кода, отладка и интеграция с компиляторами. CLion, с другой стороны, предлагает более широкий набор инструментов, включая интеллектуальное автодополнение, статический анализ кода, поддержку системы сборки CMake и многое другое.

4. Скорость и производительность:

Code::Blocks обычно считается легковесной IDE, которая работает быстро и эффективно. CLion, с другой стороны, может быть немного тяжелее и требовать больше ресурсов, но он предлагает более мощные функции и инструменты.

5. Поддержка плагинов и расширений:

Code::Blocks имеет широкий выбор плагинов, которые могут быть установлены для расширения функциональности IDE. CLion также поддерживает плагины.

6. Интеграция с системами контроля версий:

Code::Blocks имеет базовую поддержку систем контроля версий, таких как Git, SVN и CVS. Однако, CLion предлагает более глубокую интеграцию с Git и другими системами контроля версий, включая возможность просмотра истории изменений, слияния веток и управления ветками.

7. Поддержка отладки:

Code::Blocks и CLion оба предлагают возможности отладки, такие как точки останова, просмотр переменных и выполнение по шагам. Однако, CLion имеет более продвинутые инструменты отладки, такие как режим отладки построчно и поддержка LLDB для отладки на macOS.

8. Сообщество и поддержка:

Code::Blocks является открытым исходным кодом и имеет активное сообщество пользователей, которые предлагают поддержку и обновления. CLion, с другой стороны, является коммерческим продуктом и имеет поддержку от JetBrains, которая предлагает регулярные обновления и техническую поддержку.

Контрольные вопросы

1. Что такое IDE?

IDE – это ПО, которое объединяет инструменты для разработки приложений и их тестирования в едином интерфейсе

2. Что такое API?

(англ. Application Programming Interface) — это интерфейс программирования приложений. API сервиса предоставляет набор готовых процедур, функций и структур, с помощью которых разработчики могут создавать свои программы, приложения, скрипты (далее — приложения) для работы с сервисом.

3. Что такое библиотека в программировании?

Библиотека в программировании - это готовый набор функций, классов и объектов для какого-либо языка программирования. Обычно такие наборы объединены назначением или сферой использования, например, математические библиотеки или библиотеки для работы с графикой.

4. Понятия статической и динамической библиотек

Статическая библиотека - это файл с программным кодом, который описывает нужные сущности. Ее подключают к программе с помощью специальной команды. При запуске код из библиотеки как бы «вставляется» в программу и становится ее частью.

Динамическая библиотека - это файл с программным кодом, который загружается в программу во время ее выполнения. Когда вы компилируете программу, использующую динамическую библиотеку, библиотека не становится частью вашего исполняемого файла – она остается отдельной единицей.

5. Что такое плагин?

Плагин - это программный модуль, который добавляет определенную функцию в уже существующую программу. Плагины являются независимыми модулями, которые динамически подключаются к основной программе и предназначены для расширения и/или улучшения ее функциональности.

6. Назовите несколько консольных текстовых редакторов для Linux.

Nano, Vim, Emacs, Spacemacs, Jed

7. Что делает команда gcc?

Команда gcc используется для компиляции исходного кода на языках C, C++, Objective-C, Fortran, Ada, Go и D.

8. Что делает команда make?

Команда make - это утилита, предназначенная для автоматизации процесса сборки программ из исходных файлов. Она считывает инструкции из файла с именем Makefile или makefile в текущей директории и выполняет команды, необходимые для компиляции и сборки программы. В файле Makefile описываются зависимости между файлами и команды для обновления каждого файла.

9. Что делает команда gdb?

Команда gdb - это отладчик, который используется для отладки программ на языках C, C++, Objective-C, Fortran, Ada, Go и других. Он позволяет запускать программу в режиме отладки, останавливать ее выполнение в любой момент и анализировать состояние программы.

10. Дайте определение заголовочного файла и файла реализации

Заголовочный файл - это файл, который содержит объявления функций, классов, переменных и других элементов программы, но не содержит их реализации. Он используется для описания интерфейса программы и для того, чтобы другие файлы могли использовать его содержимое.

Файлы реализации - это файлы, которые содержат определения функций, классов, переменных и других элементов программы. Они используются для реализации функциональности, описанной в заголовочных файлах.

11. Что означает единица трансляции? В чем особенность разработки программ из нескольких единиц трансляции?

Единица трансляции - это минимальный блок исходного кода, который может быть скомпилирован в объектный файл. В языках программирования единицей трансляции может быть отдельная программа, подпрограмма, функция или модуль вместе с включенными файлами. Особенностью разработки программ из нескольких единиц трансляции является возможность разбиения программы на логически связанные фрагменты, что упрощает ее разработку и поддержку. Каждая единица трансляции компилируется отдельно, а затем объединяется вместе компоновщиком.

12. Дайте краткую характеристику каждому этапу трансляции программ, написанных на Си.

Процесс трансляции программы на языке Си состоит из следующих этапов:

1. Текстовый препроцессор - обрабатывает исходный код программы, выполняя директивы препроцессора, такие как `#include`, `#define` и т.д.
2. Компилятор - преобразует исходный код программы на языке Си в объектный код, который представляет собой машинный код, но еще не зависит от конкретной платформы.
3. Линковщик - объединяет объектные файлы и библиотеки в один исполняемый файл, разрешая ссылки на функции и переменные, которые определены в других файлах.

Вывод:

Получила навыки выполнения типовых операций мониторинга и управления состояниями виртуальных машин, изучила компиляцию и компоновку программ на примере инструментального средства разработки программ GCC, ознакомилась с интегрированными средами разработки и используемыми редакторами, установила редактор Vim и ознакомилась с ним.