

Липецкий государственный технический университет

Кафедра прикладной математики

Отчет по лабораторной работе № 2
«Средства разработки ПО в ОС Unix»
по курсу «Операционные системы»

Студентка

подпись, дата

Пестова А.Ю.
фамилия, инициалы

Группа ПМ-19-1

Руководитель

Доцент, к. пед. наук
ученая степень, ученое звание

подпись, дата

Кургасов В.В.
фамилия, инициалы

Липецк 2022 г.

Содержание

Цель работы	3
Задание кафедры	4
1. Ход работы	5
1.1. Упомянутое программное обеспечение	5
1.2. Сравнительная характеристика популярных IDE	6
1.3. Установка и настройка Vim	16
1.4. Программа на языке C	20
1.5. Демонстрация работы отладчика на основе программы на языке C	22
Выводы	23
Контрольные вопросы	24

Цель работы

Получить представление о технологиях и средствах разработки программного обеспечения в ОС Unix.

Задание кафедры

1. Ознакомиться с представленным теоретическим материалом. Составьте таблицу, с упомянутым программным обеспечением в материале данной работы.
2. Выполните сравнительную характеристику популярных IDE.
3. Установить (если отсутствует в системе) редактор Vim. Провести его настройку, согласно предложенному материалу лабораторной работы.
4. Напишите программу в соответствии с назначенным вариантом используя редактор Vim.
5. Продемонстрируйте работу отладчика на примере написанной вами программе.
6. Подготовьте отчёт о выполненной работе содержащий скриншоты всех этапов.

1. Ход работы

1.1. Упомянутое программное обеспечение

№	ПО
1	GNU Compiler Collection
2	The GNU Project Debugger
3	GNU Make
4	Code::Blocks
5	Netbeans
6	Eclipse CDT
7	CodeLite IDE
8	Bluefish
9	Brackets
10	Atom
11	Sublime
12	JetBrains CLion
13	Visual Studio Code
14	KDevelop
15	Anjuta DevStudio
16	GNAT Programming Studio
17	Qt Creator
18	Emacs
19	SlickEdit
20	VI/VIM

Таблица 1. Упомянутое программное обеспечение

1.2. Сравнительная характеристика популярных IDE

CLion



Рисунок 1 – CLion

Официальный сайт: <https://www.jetbrains.com/ru-ru/clion/>

Платформы: Windows/Linux/macOS

Поддерживаемые языки: C++, C, Objective C, Kotlin, Python, Swift, Fortran, JavaScript, CSS и другие.

Стоимость: от 199\$ в год (последующие года будут стоить дешевле).

И снова продукт JetBrains. CLion – идеальное кроссплатформенное решение для тех, кто работает на C и C++ (и не только). Умный редактор, удобный генератор кода, статический и динамический анализ, безопасный рефакторинг... Особенности данной среды разработки можно перечислять бесконечно.

Преимущества:

- Поддержка удаленной разработки по SSH.
- Просмотр значений переменных прямо в редакторе.
- Умная помощь при написании кода.
- Возможность кастомизировать редактор.
- Быстрый и безопасный рефакторинг.

- Широкий функционал. IDE можно использовать даже для программирования микроконтроллеров.

Недостатки :

- Нет бесплатной версии. Но можно скачать пробную версию.

Eclipse



Рисунок 2 – Eclipse

Официальный сайт: <https://www.eclipse.org/ide/>

Платформы: Windows/Linux/macOS

Поддерживаемые языки: C, C++, Java, Perl, PHP, Python, Ruby и другие.

Это бесплатная опенсорсная среда разработки, которая хорошо подойдет как новичкам, так и опытным разработчикам. Помимо инструментов отладки и поддержки Git/CVS, Eclipse поставляется с Java и инструментом для создания плагинов. Изначально Eclipse использовалась только для Java, но сейчас, благодаря плагинам и расширениям, ее функции значительно расширились. Именно из-за возможности расширить Eclipse своими модулями эта платформа и завоевала свою популярность среди разработчиков. Функционал Eclipse не такой большой, как у IntelliJ IDEA, зато эта среда разработки распространяется с открытым исходным кодом.

Преимущества:

- Возможность программировать на множестве языков.
- Значительная гибкость среды за счет модульности.
- Возможность интеграции JUnit.
- Удаленная отладка (при использовании JVM).

Недостатки:

- Новичкам может быть сложно разобраться в многообразии возможностей.

NetBeans



Рисунок 3 – NetBeans

Официальный сайт: <https://netbeans.org/>

Платформы: Windows/Linux/macOS/BSD

Поддерживаемые языки: C, C++, C++ 11, Fortan, HTML 5, Java, PHP и другие.

Это бесплатная опенсорсная IDE. Прекрасно подойдет как для работы с уже имеющимися проектами, так и для создания нового. Это одна из лучших IDE для разработки Java-приложений, в которую можно установить пакеты, обеспечивающие и поддержку других языков.

Преимущества:

- Интуитивно понятный интерфейс drag-and-drop.
- Динамические и статические библиотеки.
- Возможность удаленной разработки.
- Совместима с Windows, Linux, macOS и Solaris.
- Поддержка Qt.
- Поддерживает различные компиляторы, в том числе CLang/LLVM, Cygwin, GNU, MinGW и Oracle Solaris Studio.
-

Недостатки:

- NetBeans требуется много памяти, поэтому на некоторых машинах эта среда может подтормаживать.

Code::Blocks



Рисунок 4 – Code::Blocks

Официальный сайт: <http://www.codeblocks.org>

Платформы: Windows/Linux/macOS

Поддерживаемые языки: C, C++, Fortran

Опенсорсная среда разработки – простая, нетребовательная к ресурсам и очень производительная. Поддерживает огромное количество компиляторов и отладчиков. Расширить функционал можно с помощью бесплатных плагинов.

Преимущества:

- Удобная структура меню.
- Высокая производительность.
- Встроенная система быстрой сборки.

Недостатки:

- Не подойдет профессионалам.
- Много багов.
- Несколько устаревший интерфейс

Visual Studio Code



Рисунок 5 – Visual Studio Code

Тип: IDE

Цена: Бесплатно

Поддерживаемые платформы: Windows, Linux, macOS

Это популярнейший редактор текста для программистов, который можно превратить в мощную IDE, установив дополнительные плагины. Популярность VS Code обоснована его производительностью, открытым исходным кодом и неограниченной функциональностью.

Удобный, современный интерфейс вкупе с высокой скоростью работы делают VS Code идеальным инструментом для разработки программного обеспечения любого формата, в том числе и на многих языках.

В нем есть подсветка синтаксиса языка по умолчанию, автоматическое дополнение кода, а также система IntelliSense, помогающая находить ошибки в коде, взаимодействовать с API и дополнять код элементами из подключенных к проекту файлов.

Все это удобство дополняется функцией компиляции языка с помощью специализированного плагина. Все инструменты, необходимые для разработки, при этом доступны в едином интерфейсе.

Sublime Text



Рисунок 6 – Sublime Text

Цена: 80 \$

Поддерживаемые платформы: Windows, Linux, macOS

Легковесный проприетарный редактор со всеми необходимыми для разработки функциями. Написан на C++ и Python и поддерживает плагины на Python. Sublime Text дает возможность вносить изменения сразу в несколько строчек кода, имеет быструю навигацию и хорошо кастомизируется под собственные нужды.

Его можно установить бесплатно для того, чтобы познакомиться со всеми основными функциями, но для дальнейшего использования нужно будет купить лицензию.

Vim



Рисунок 7 – Vim

Цена: Бесплатно

Поддерживаемые платформы: Windows, Linux, macOS

Vim – это древняя модификация текстового редактора Vi, созданного Биллом Джоем в 1976 году. Vim, в свою очередь, появился спустя 15 лет и стал одним из наиболее значимых приложений в истории компьютеров.

Vim использует довольно специфичную по нынешним временам модель управления, полностью построенную на идее модальности в угоду устаревшим клавиатурам, на которых не было стрелок и других уже привычных для нас клавиш.

Несмотря на то, что нужды в Vim сейчас нет, он все еще популярен. Его поклонники образовали чуть ли не религиозный культ, строго верующий в Vim, защищающий его от противников и всячески продвигающий в обществе как лучшее, что могло быть изобретено для редактирования текста.

99% поклонников Vim – это разработчики и гики, пытающиеся добиться максимальной эффективности от используемых инструментов, и Vim в этом всяческим им помогает за счет гибкой системы настроек, такого же гибкого управления и бесконечной модифицируемости.

Давайте же подробнее разберемся в том, чем примечателен Vim и стоит ли вообще его пробовать.

Досконально изучив инструмент и натренировавшись в его использовании, вы становитесь на голову выше других спецов, потому что попросту

тратите меньше времени на выполнение рутинных задач. Быстрее удаляете строки, быстрее перемещаетесь по коду, быстрее вносите изменения в разных участках файла и приложения целиком. При этом тратите не только меньше времени, но и меньше сил.

Вы учите Vim один раз и пользуетесь им везде. Независимо от выбранной ОС, у вас всегда будет доступ к Vim, и он будет таким же, как Vim в любой другой системе (если, конечно, вы не обвешаете редактор на своем ПК плагинами и темами).

Возникает вопрос, почему Vim не используют все разработчики, раз он такой хороший. Ответ кроется в пороге вхождения. Он настолько высок для нового поколения пользователей ПК, что в интернете до сих пор гуляют мемы про то, как сложно закрыть Vim, если случайно его запустил.

Vim, без преувеличения, сложен в освоении. Даже заставить человека отказаться от использования стрелок и перейти на так называемый home-блок клавиш – тот еще квест.

1.3. Установка и настройка Vim

Переменная `expandtab` включает замену табов на пробелы, `tabstop` - количество пробелов в одном обычном табе, `softtabstop` - количество пробелов в табе при удалении, `smarttab` - при нажатии таба в начале строки добавляет количество пробелов равное `shiftwidth`.

Чтобы добавить нумерацию строк добавим команду `number`.

Сделаем ещё небольшой отступ между левой частью окна: `foldcolumn=2`.

Изменим цветовую схему: `colorscheme industry`.

Включим подсветку синтаксиса: `syntax on`.

При нажатии неверной клавиши или ошибке в Vim проигрывается специальный звук. Отключим его: `set noerrorbells`, `set novisualbell`.

Включим мышку во всех режимах программы, добавив такую строчку: `set mouse=a`.

Добавляем горячую клавишу на открытие панели, добавив строчку: `nnmap <F6> :NERDTreeToggle<CR>`.

Меняем кодировку с помощью строчки: `set encoding = utf8`.

Также устанавливаем стандарт использования символов переноса строки в файлах: `set ff=unix,dos,mac`.

1. NERDTree – это проводник файловой системы для редактора Vim. С помощью этого плагина пользователи могут визуально просматривать сложные иерархии каталогов, быстро открывать файлы для чтения или редактирования и выполнять основные операции с файловой системой.
2. Vim-airline — легкий плагин, заменяющий строку статуса в Vim с широкими возможностями для кастомизации. Получил свое название по причине того, что первую его версию автор написал, летя в самолете.
3. vim-css-color – очень быстрый цветной маркер ключевых слов с контекстно-зависимой поддержкой многих языковых синтаксисов.
4. lightline – легкий и настраиваемый плагин строки состояния/таблицы для Vim.
5. flattened – цветовая схема для редакторов кода и эмуляторов терминалов.

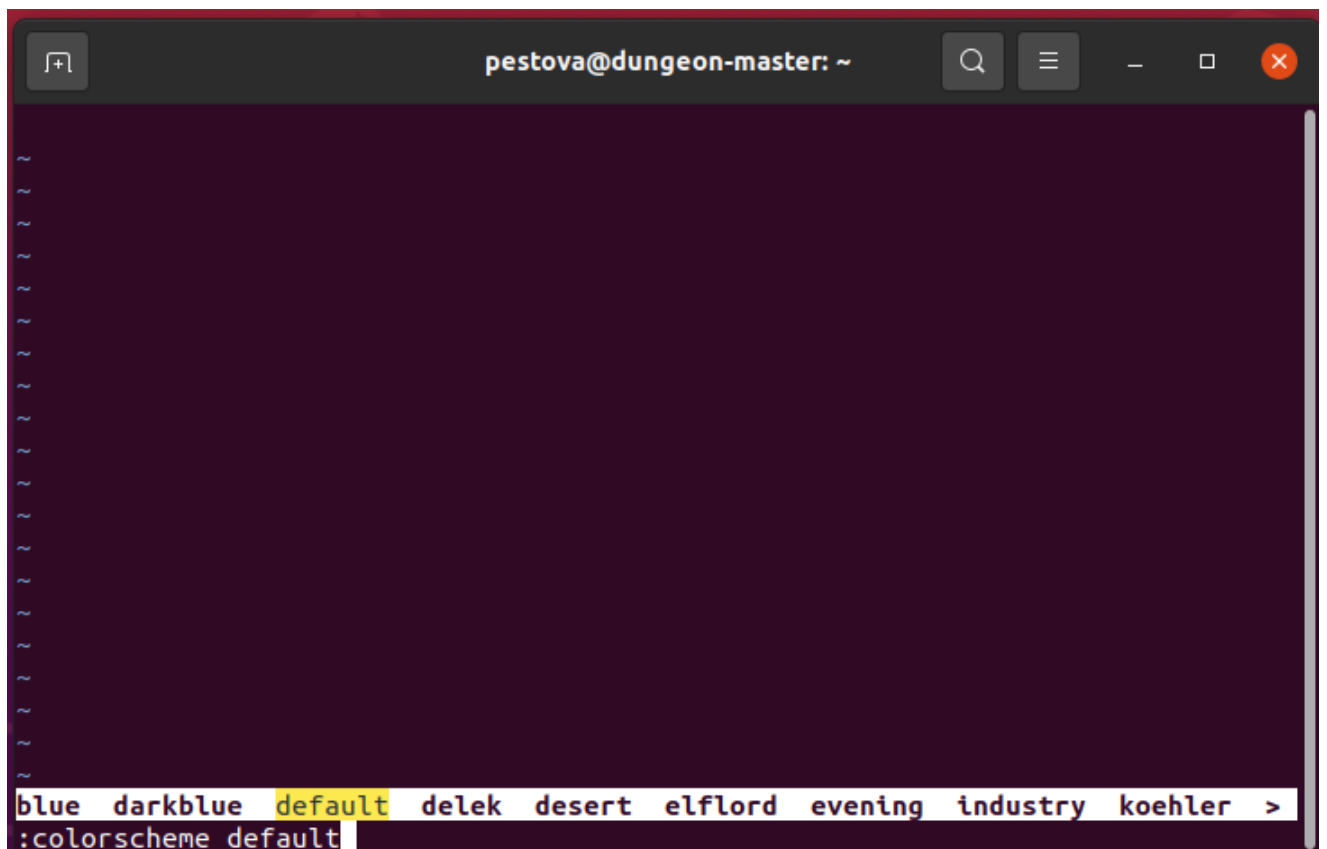


Рисунок 8 – Смена цветовой схемы

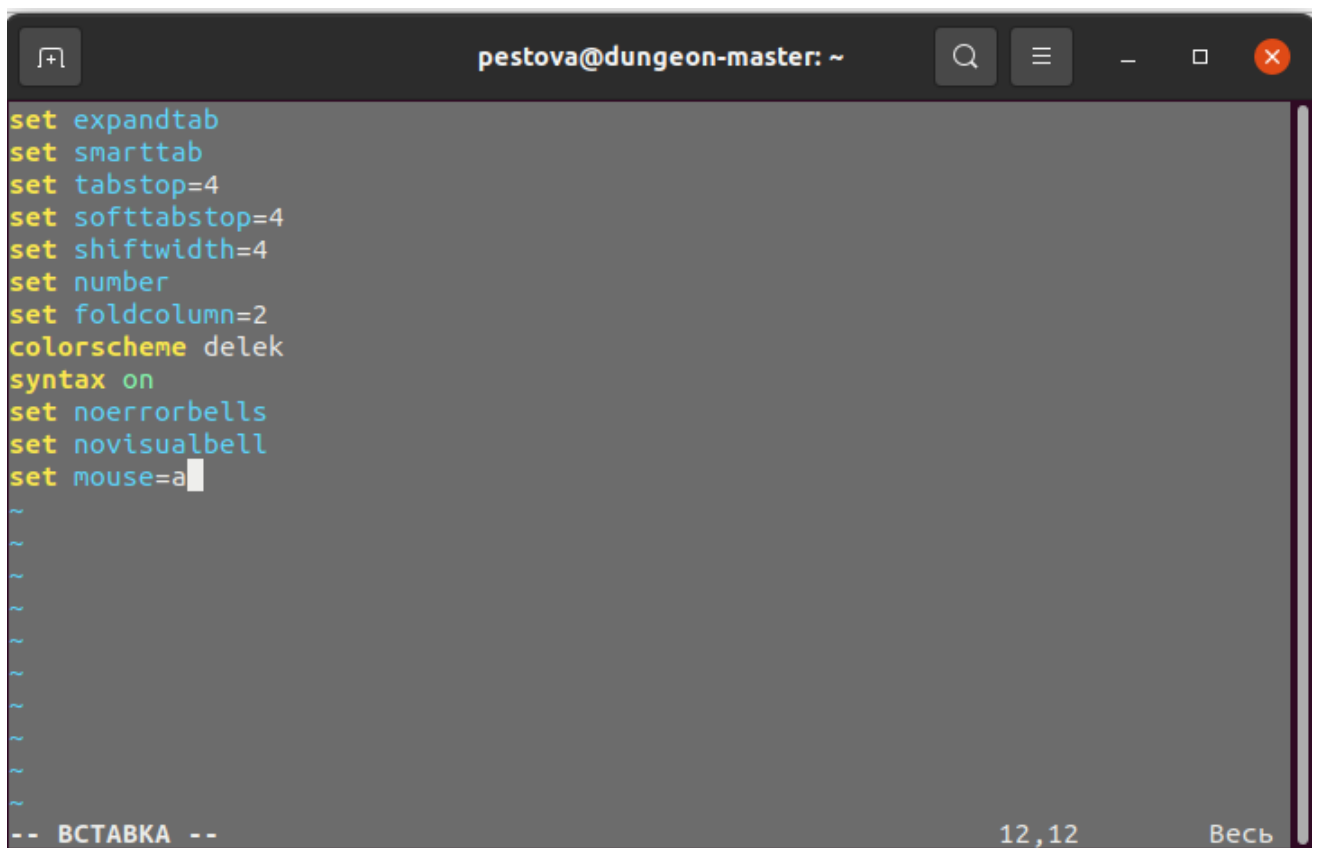


Рисунок 9 – Настройка конфигурационного файла

```

pestova@dungeon-master:~$ git clone https://github.com/preservim/nerdtree.git ~/.vim/pack/vendor/start/nerdtree
Клонирование в «/home/pestova/.vim/pack/vendor/start/nerdtree»...
remote: Enumerating objects: 6078, done.
remote: Counting objects: 100% (70/70), done.
remote: Compressing objects: 100% (61/61), done.
remote: Total 6078 (delta 24), reused 26 (delta 4), pack-reused 6008
Получение объектов: 100% (6078/6078), 1.90 МиБ | 1.71 МиБ/с, готово.
Определение изменений: 100% (2759/2759), готово.
pestova@dungeon-master:~$ vim -u NONE -c "helptags ~/.vim/pack/vendor/start/nerdtree/doc" -c q
pestova@dungeon-master:~$

```

Рисунок 10 – Установка плагина NerdTree

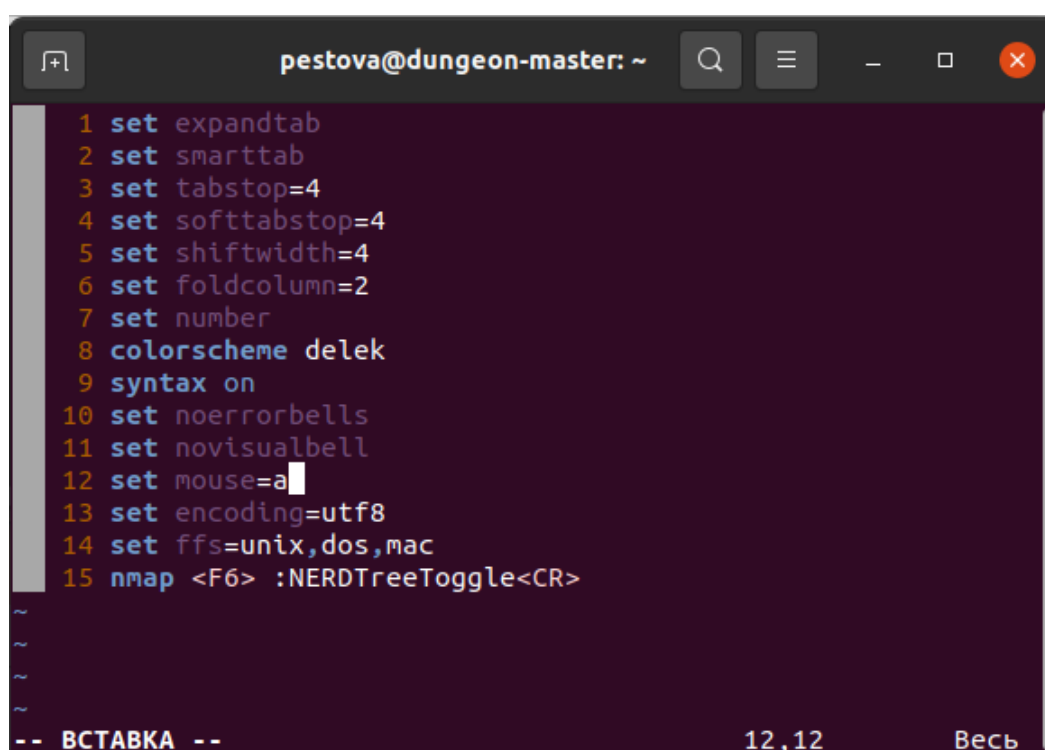


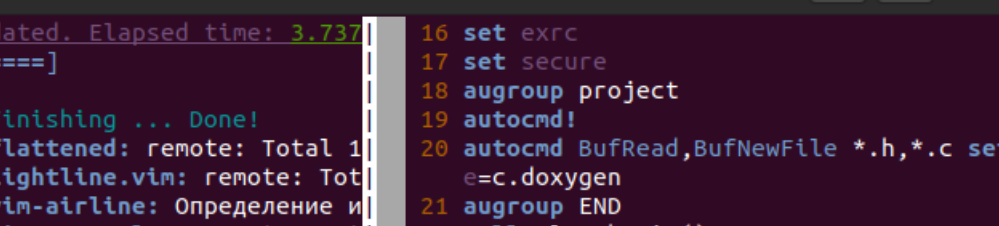
Рисунок 11 – Дополнительная настройка конфигурационного файла

```

pestova@dungeon-master:~$ git clone https://tpope.io/vim/eunuch.git ~/.vim/pack/vendor/start/eunuch
Клонирование в «/home/pestova/.vim/pack/vendor/start/eunuch»...
warning: переадресация на https://github.com/tpope/vim-eunuch.git/
remote: Enumerating objects: 403, done.
remote: Counting objects: 100% (40/40), done.
remote: Compressing objects: 100% (19/19), done.
remote: Total 403 (delta 18), reused 31 (delta 11), pack-reused 363
Получение объектов: 100% (403/403), 70.25 КиБ | 749.00 КиБ/с, готово.
Определение изменений: 100% (178/178), готово.
pestova@dungeon-master:~$ vim -u NONE -c "helptags ~/.vim/pack/vendor/start/eunuch/doc" -c q
pestova@dungeon-master:~$

```

Рисунок 12 – Установка плагина eunuch



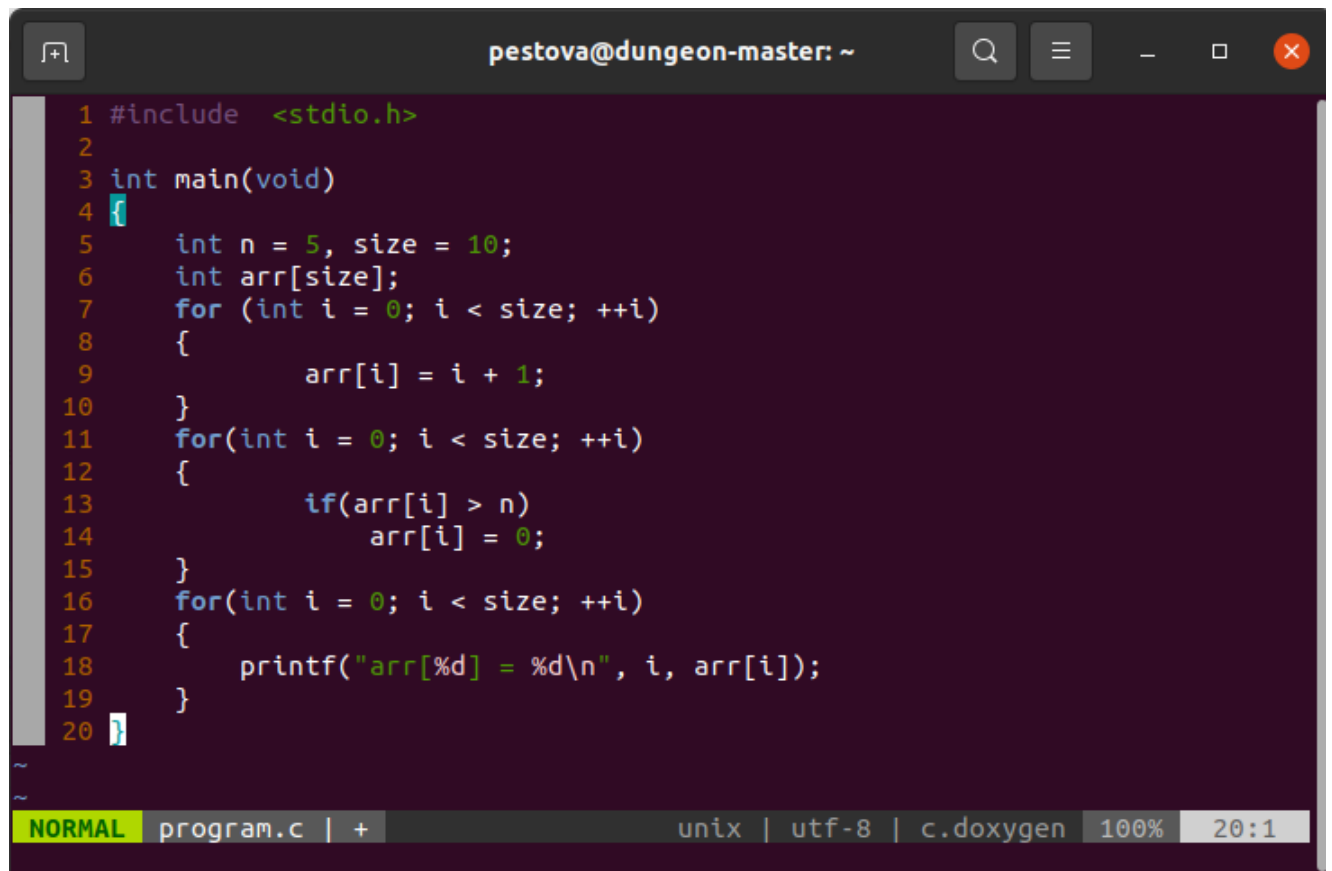
```
pestova@dungeon-master: ~
1 Updated. Elapsed time: 3.737
2 [=====]
3
4 - Finishing ... Done!
5 - flattened: remote: Total 1
6 - lightline.vim: remote: Tot
7 - vim-airline: Определение и
8 - vim-css-color: remote: Tot
9 - NERDTree: Определение изме
16 set exrc
17 set secure
18 augroup project
19 autocmd!
20 autocmd BufRead,BufNewFile *.h,*.c set filetype=c.doxygen
21 augroup END
22 call plug#begin()
23 Plug 'preservim/NERDTree'
24 Plug 'vim-airline/vim-airline'
25 Plug 'https://github.com/ap/vim-css-color'
26 Plug 'itchyny/lightline.vim'
27 Plug 'romainl/flattened'
28 call plug#end()

Plugins .vimrc 100% ln:28 %15
```

1.4. Программа на языке C

Вариант №8

Задание: задать произвольное число f . Обнулить элементы массива, которые больше значения f .



```
1 #include <stdio.h>
2
3 int main(void)
4 {
5     int n = 5, size = 10;
6     int arr[size];
7     for (int i = 0; i < size; ++i)
8     {
9         arr[i] = i + 1;
10    }
11    for(int i = 0; i < size; ++i)
12    {
13        if(arr[i] > n)
14            arr[i] = 0;
15    }
16    for(int i = 0; i < size; ++i)
17    {
18        printf("arr[%d] = %d\n", i, arr[i]);
19    }
20 }
```

~

NORMAL program.c | + unix | utf-8 | c.doxygen 100% 20:1

Рисунок 14 – Написание программы в редакторе vim

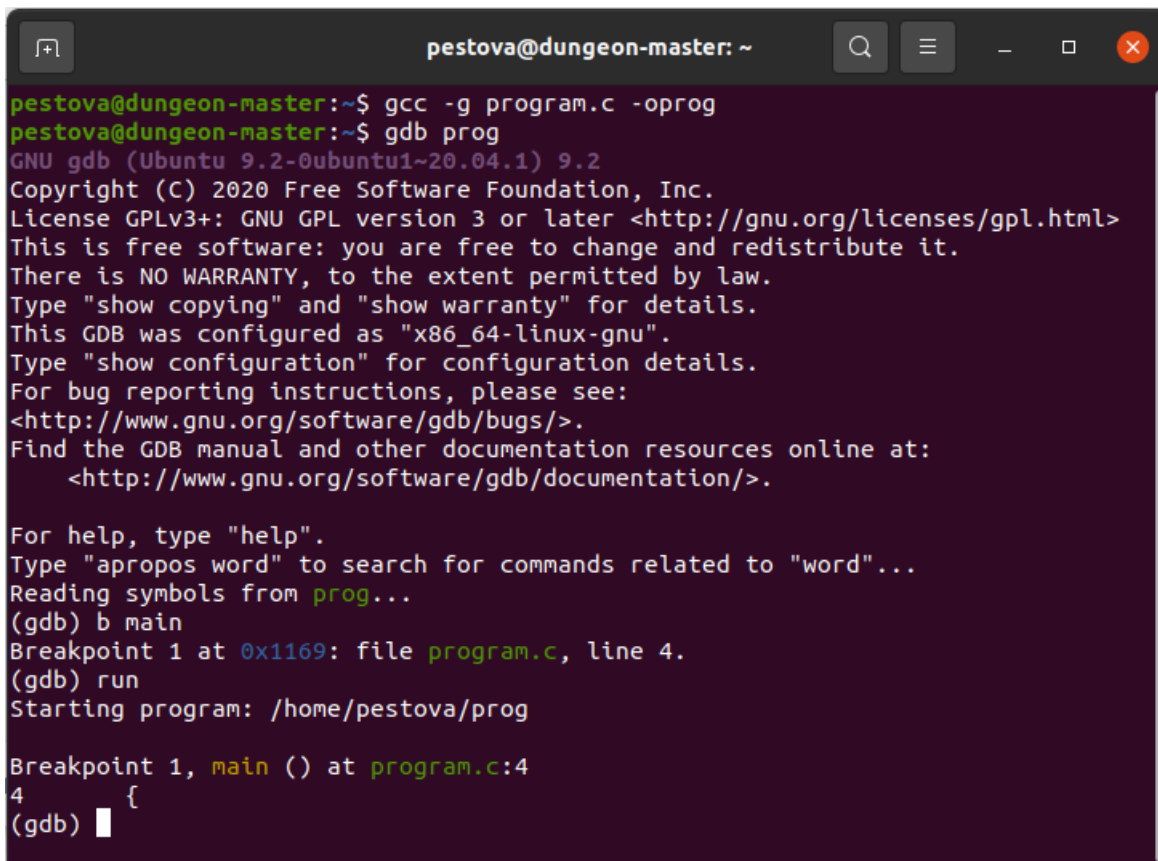
```
pestova@dungeon-master:~$ ./program
arr[0] = 1
arr[1] = 2
arr[2] = 3
arr[3] = 4
arr[4] = 5
arr[5] = 0
arr[6] = 0
arr[7] = 0
arr[8] = 0
arr[9] = 0
pestova@dungeon-master:~$
```

Рисунок 15 – Просмотр результата программы

```
pestova@dungeon-master:~$ gcc program.c -o program
pestova@dungeon-master:~$ ls -l
итого 56
-rwxrwxr-x 1 pestova pestova 16752 фев 22 11:57 program
-rw-rw-r-- 1 pestova pestova 352 фев 22 11:55 program.c
drwxr-xr-x 2 pestova pestova 4096 фев 22 11:08 Видео
drwxr-xr-x 2 pestova pestova 4096 фев 22 11:08 Документы
drwxr-xr-x 2 pestova pestova 4096 фев 22 11:08 Загрузки
drwxr-xr-x 2 pestova pestova 4096 фев 22 11:55 Изображения
drwxr-xr-x 2 pestova pestova 4096 фев 22 11:08 Музыка
drwxr-xr-x 2 pestova pestova 4096 фев 22 11:08 Общедоступные
drwxr-xr-x 2 pestova pestova 4096 фев 22 11:08 'Рабочий стол'
drwxr-xr-x 2 pestova pestova 4096 фев 22 11:08 Шаблоны
pestova@dungeon-master:~$
```

Рисунок 16 – Запуск результата программы

1.5. Демонстрация работы отладчика на основе программы на языке C



```
pestova@dungeon-master: ~  
pestova@dungeon-master:~$ gcc -g program.c -oprogram  
pestova@dungeon-master:~$ gdb program  
GNU gdb (Ubuntu 9.2-0ubuntu1~20.04.1) 9.2  
Copyright (C) 2020 Free Software Foundation, Inc.  
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>  
This is free software: you are free to change and redistribute it.  
There is NO WARRANTY, to the extent permitted by law.  
Type "show copying" and "show warranty" for details.  
This GDB was configured as "x86_64-linux-gnu".  
Type "show configuration" for configuration details.  
For bug reporting instructions, please see:  
<http://www.gnu.org/software/gdb/bugs/>.  
Find the GDB manual and other documentation resources online at:  
  <http://www.gnu.org/software/gdb/documentation/>.  
  
For help, type "help".  
Type "apropos word" to search for commands related to "word"..  
Reading symbols from program..  
(gdb) b main  
Breakpoint 1 at 0x1169: file program.c, line 4.  
(gdb) run  
Starting program: /home/pestova/program  
  
Breakpoint 1, main () at program.c:4  
4      {  
(gdb) █
```

Рисунок 17 – Демонстрация работы отладчика

Выводы

В ходе выполнения данной лабораторной работы была ознакомлена с теоретическим материалом, а также, с упомянутым в данной работе программным обеспечением. На основе полученных данных была выполнена сравнительная характеристика IDE и был получен опыт работы с редактором Vim. Кроме того, были получены навыки по его настройке и установке плагинов. Была ознакомлена с работой отладчика на примере написанной программы, в соответствии с вариантом.

Контрольные вопросы

1. Что такое IDE?

Интегрированная среда разработки, ИСР (англ. Integrated development environment — IDE), также единая среда разработки, ЕСР — комплекс программных средств, используемый программистами для разработки программного обеспечения (ПО).

2. Что такое API?

API (Application Programming Interface или интерфейс программирования приложений) — это совокупность инструментов и функций в виде интерфейса для создания новых приложений, благодаря которому одна программа будет взаимодействовать с другой.

3. Что такое библиотека в программировании?

Библиотека (от англ. library) в программировании — сборник подпрограмм или объектов, используемых для разработки программного обеспечения (ПО). В некоторых языках программирования (например, в Python) то же, что и модуль, в некоторых — несколько модулей. С точки зрения операционной системы (ОС) и прикладного ПО, библиотеки разделяются на динамические и статические.

4. Понятия Статической и Динамической библиотек.

Статическая библиотека - это просто архив объектных файлов, который подключается к программе во время линковки (сборки). Эффект тот же самый, как если бы вы подключали каждый из файлов отдельно, не делая из них библиотеку.

Динамическая библиотека - это созданная специальным образом библиотека, которая присоединяется к результирующей программе в два этапа. Первый этап, это естественно этап компиляции. В отличие от статических библиотек, код совместно используемых (динамических) библиотек не включается в исполняемый (бинарный) файл. Вместо этого в нем присутствует только ссылка на библиотеку.

5. Что такое плагин?

Плагин - независимо компилируемый программный модуль, динамически подключаемый к основной программе и предназначенный для расширения и/или использования её возможностей. Плагины обычно выполняются в виде библиотек общего пользования. Для возможности подключения плагинов разработчик основного приложения должен предусмотреть в нем некоторый программный интерфейс, а также хотя бы минимальные возможности по управлению набором плагинов.

6. Назовите несколько консольных текстовых редакторов для Linux.

- (a) VIM
- (b) Geany
- (c) Sublime Text Editor
- (d) Brackets
- (e) Eclipse
- (f) Kwrite
- (g) Nano
- (h) GNU Emacs

7. Что делает команда gcc?

GCC - это свободно доступный оптимизирующий компилятор для языков C, C++. Программа gcc, запускаемая из командной строки, представляет собой надстройку над группой компиляторов. В зависимости от расширений имен файлов, передаваемых в качестве параметров, и дополнительных опций, gcc запускает необходимые препроцессоры, компиляторы, линкеры.

8. Что делает команда make?

Команда make позволяет задействовать одноименную утилиту, предназначенную для компиляции программного обеспечения из исходных кодов

9. Что делает команда gdb?

GDB (GNU Debugger — отладчик проекта GNU) работает на многих UNIX-подобных системах и умеет производить отладку многих языков программирования.

10. Дайте определение заголовочного файла и файла реализации. Приведите пример.

Заголовочный файл (англ. header file) или подключаемый файл — файл, содержимое которого автоматически добавляется препроцессором в исходный текст в том месте, где располагается некоторая директива. В языках программирования Си и C++ заголовочные файлы — основной способ подключить к программе типы данных, структуры, прототипы функций, перечисляемые типы и макросы, используемые в другом модуле. (\$I file.inc в Паскале, include <file.h> в Си)

Файлы реализации - это отдельные модули, которые разрабатываются и транслируются независимо друг от друга и объединяются при создании выполняемой программы. Файлы реализации могут подключать описания, содержащиеся в заголовочных файлах. Сами заголовочные файлы также могут использовать другие заголовочные файлы.

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

Единица трансляции — максимальный блок исходного текста, который физически можно оттранслировать (результат препроцессирования).

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

Препроцессирование (1) просматривает входной .c файл, выполняет содержащиеся в нём директивы препроцессора, в частности, включает в него содержимое других файлов, указанных в директивах include.;

Трансляция в ассемблер (2) - на вход подаётся одна единица трансляции, а на выходе (при отсутствии синтаксических и семантических ошибок) выдаётся файл на языке ассемблера для (как правило) машины, на которой ведётся трансляция. Файл с оттранслированной программой на языке ассемблера имеет суффикс имени .s, но точно так же, как и результат работы препроцессора, он по умолчанию удаляется.;

Ассемблирование (3) - он получает на входе результат работы предыдущей стадии и генерирует на выходе объектный файл. Объектные файлы в UNIX имеют суффикс .o.;

Компоновка (4) - компоновщик получает на вход набор объектных файлов, соответствующим единицам трансляции, составляющим программу, подключает к ним стандартную библиотеку языка Си и библиотеки, указанные пользователем, и на выходе получает исполняемую программу..