

## Учебная практика №1 (Задание №2)

### «Получение основных навыков для работы на платформе Linux Ubuntu»

#### N. План хода работы в учебной практике.

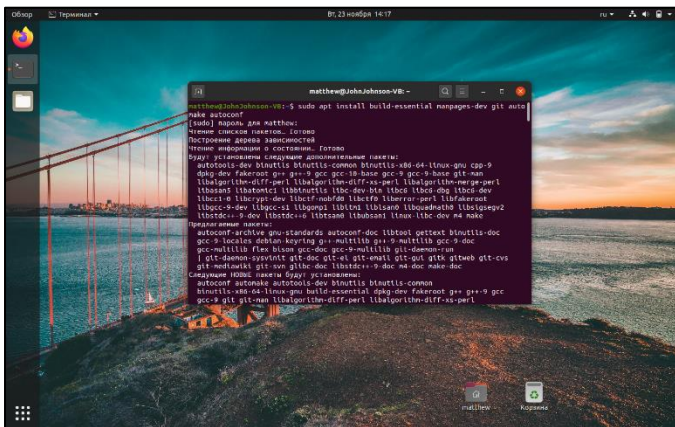
N.I. Этап 1. Создание исходного файла через терминал, на платформе Linux Ubuntu. (I)

N.II. Этап 2. Установка нужного ПО через терминал, на платформе Linux Ubuntu. (II)

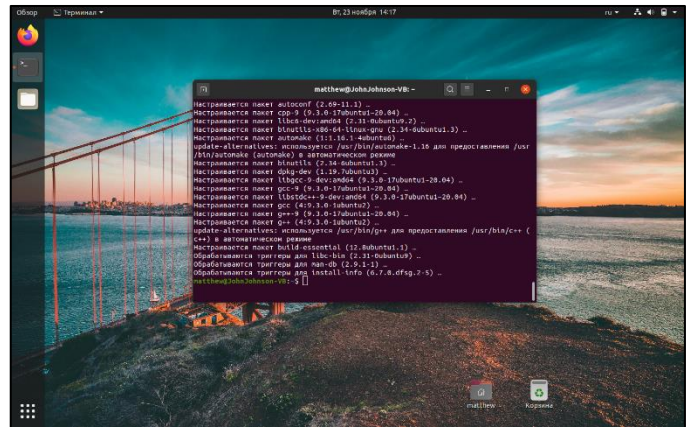
N.III. Этап 3. Выполнение задания в ПО Radare2 (iaito), на платформе Linux Ubuntu. (III)

#### I. Этап 1. Создание исходного файла через терминал, на платформе Linux Ubuntu. (→)

I.I. Открываем терминал и с помощью специальных команд, устанавливаем компилятор на Linux Ubuntu.



```
matthew@johnjohnson-VB:~$ sudo apt install build-essential nasm gcc git autoconf
[...]
```

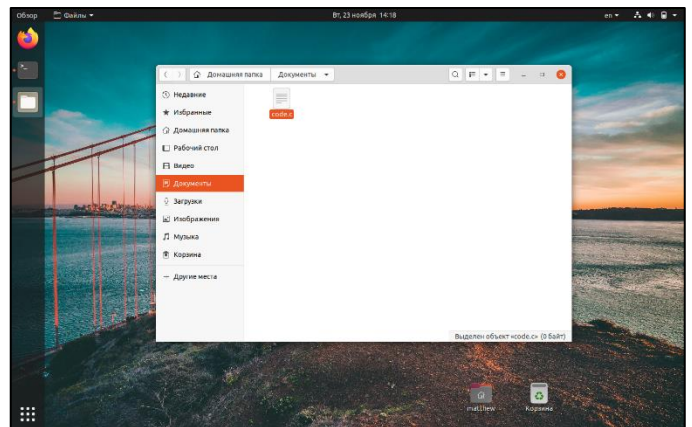


```
matthew@johnjohnson-VB:~$ sudo apt install gcc
[...]
```

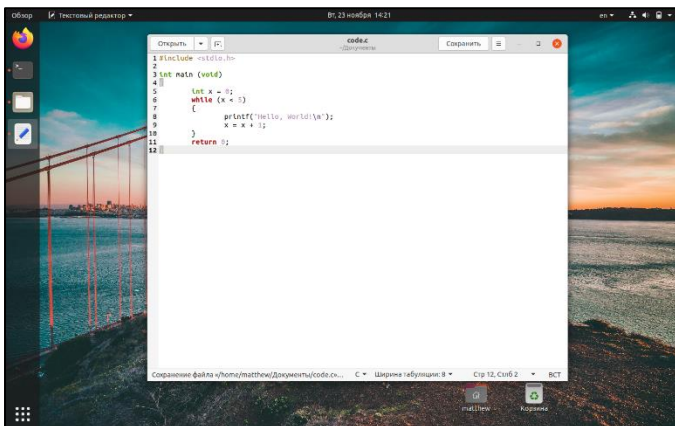
I.II. С помощью специальной команды создаем файл, в нем будет код для компиляции.



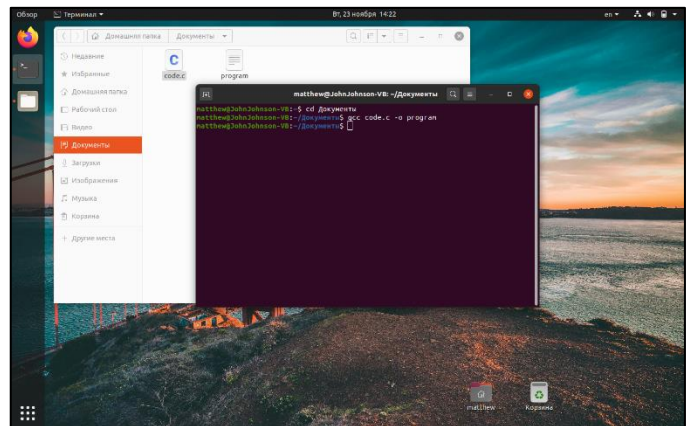
```
matthew@johnjohnson-VB:~$ touch code.c
```



I.III. Открываем исходный файл, пишем программный код, а затем производим компиляцию.



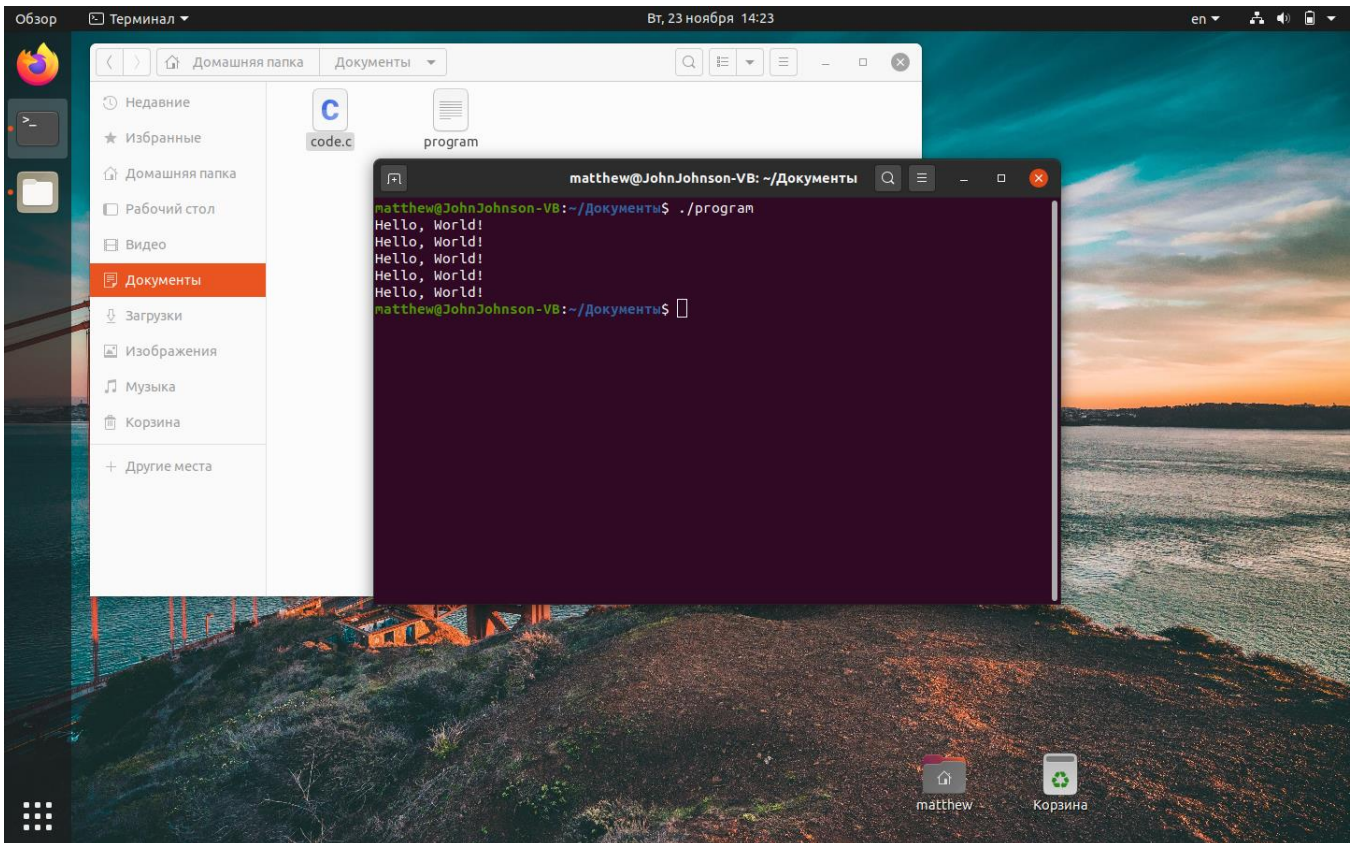
```
1 #include <stdio.h>
2
3 int main(void)
4 {
5     int x = 1;
6     while (x < 10)
7     {
8         printf("Hello, world!\n");
9         x = x + 1;
10    }
11    return 0;
12 }
```



```
matthew@johnjohnson-VB:~$ gcc code.c -o program
```



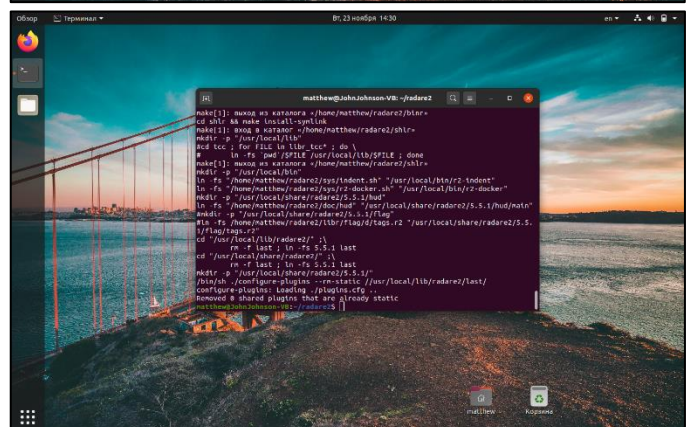
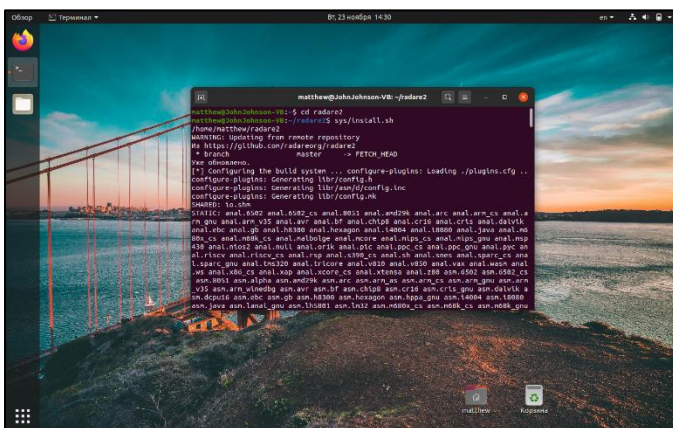
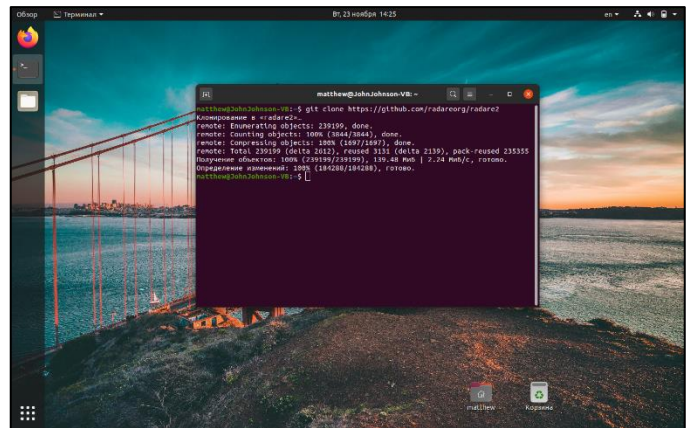
## I.IV. С помощью терминала открываем и проверяем наш исходный файл (код).



*Итог: Входной файл создан успешно. Данные программного кода работают и не требуют исправлений!*

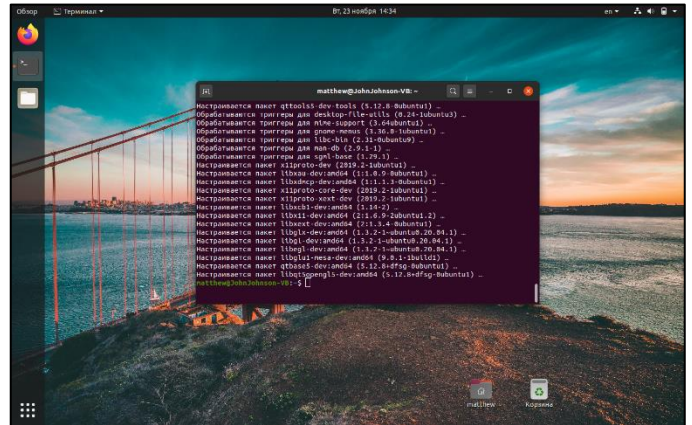
## II. Этап 2. Установка нужно ПО через терминал, на платформе Linux Ubuntu. (→)

II.I. Открываем терминал и устанавливаем необходимые для работы Radare2 пакеты, с помощью специальной команды. Устанавливаем ПО Radare2.

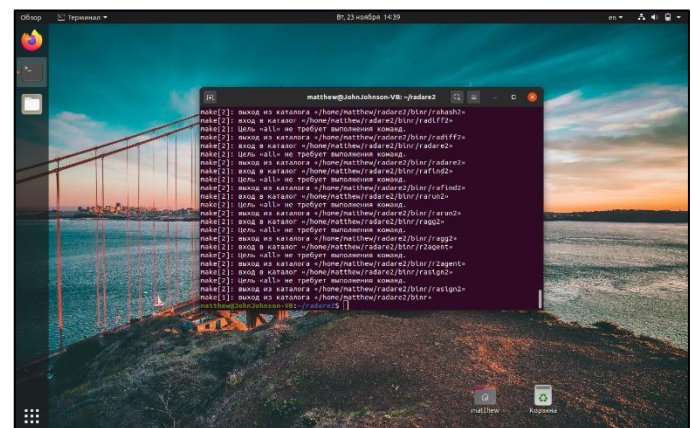
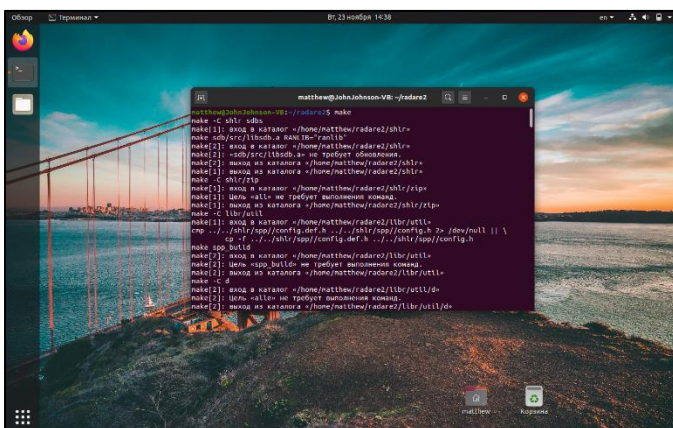
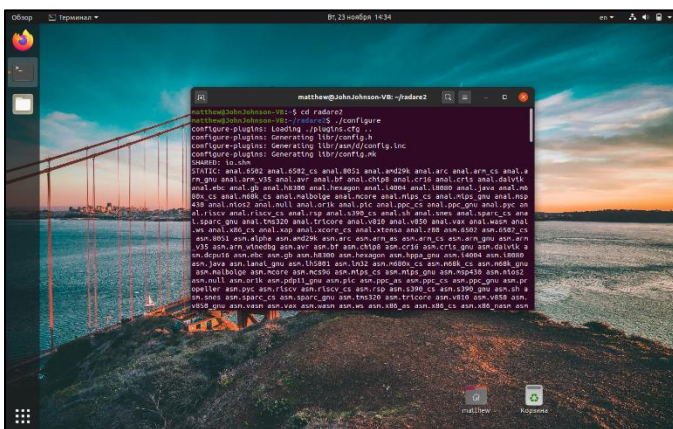




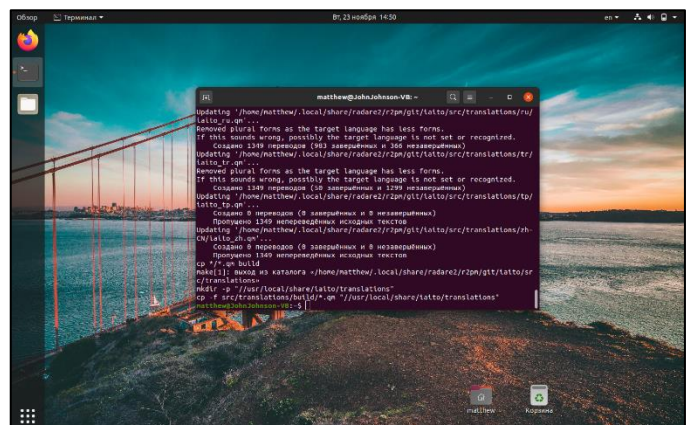
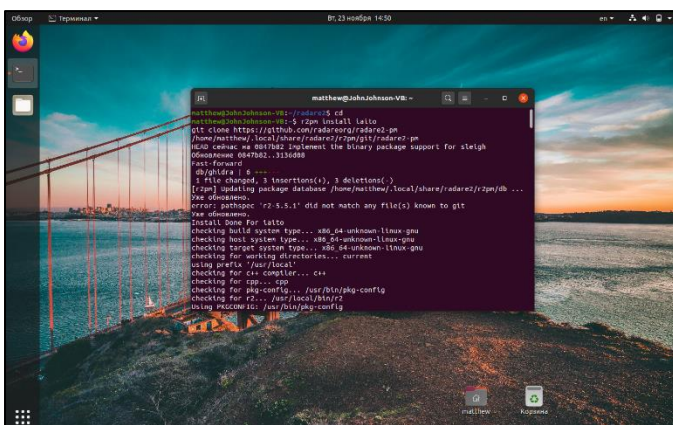
## II.II. С помощью специальной команды устанавливаем необходимые компоненты, для оболочки iaito.



II.III. С помощью специальной команды, открываем конфигурационный файл и производим установку.

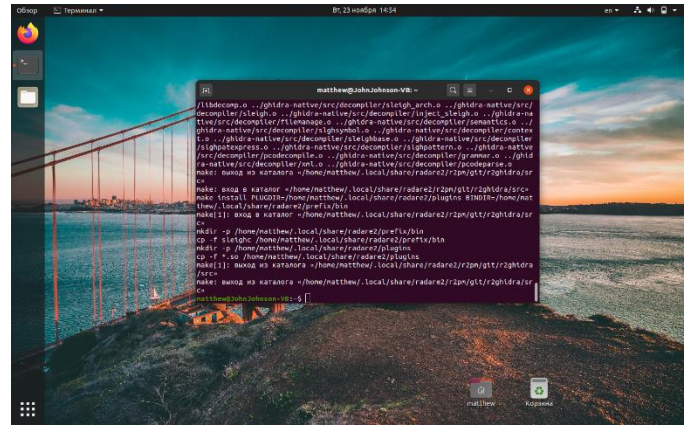


## II.IV. Устанавливаем оболочку `iaito` с помощью специальной команды.

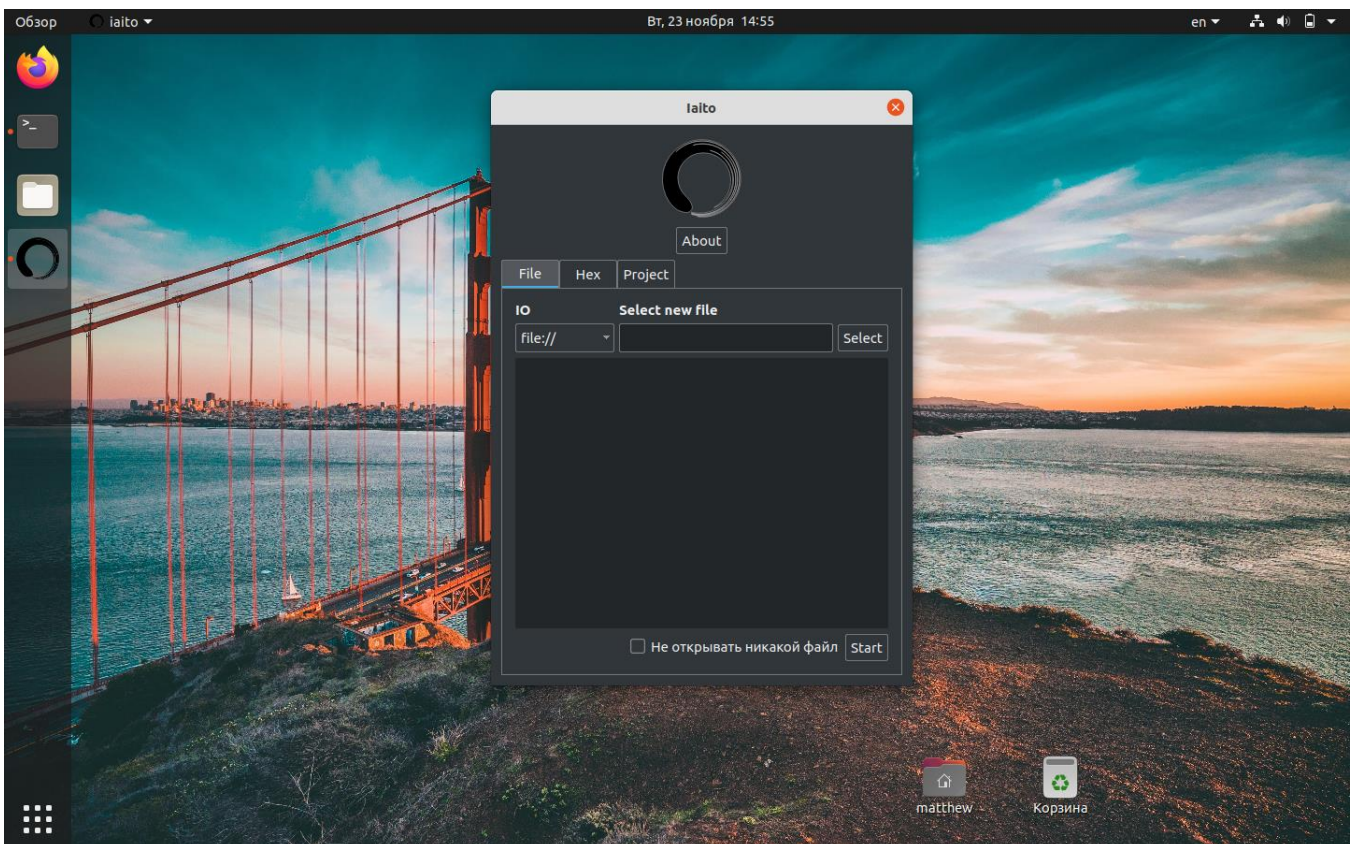
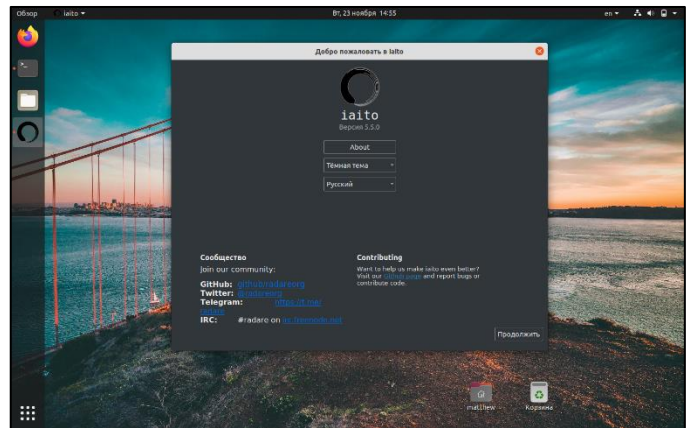
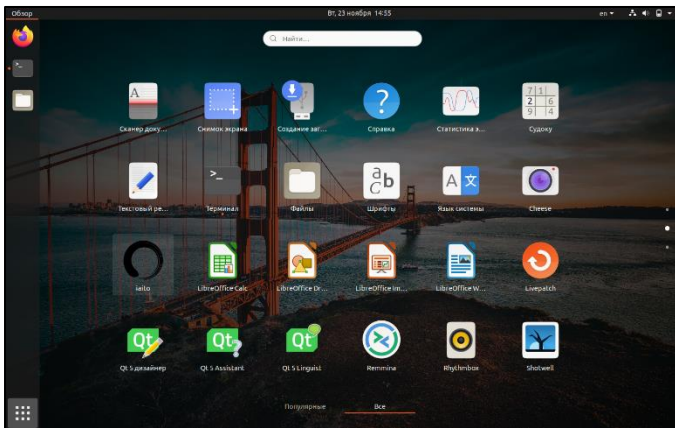




## II.V. Устанавливаем специальный плагин «Ghidra» с помощью специальной команды.



## II.VI. Открываем меню приложений, установленное приложение iaito.

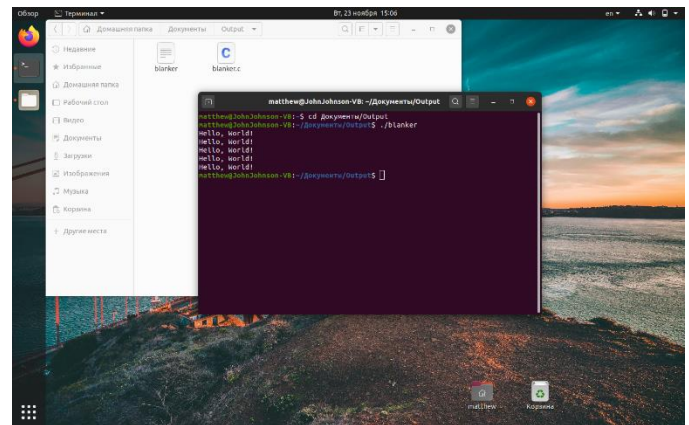
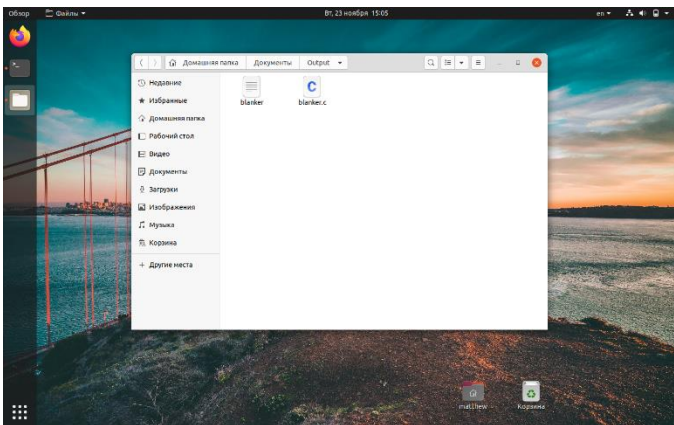
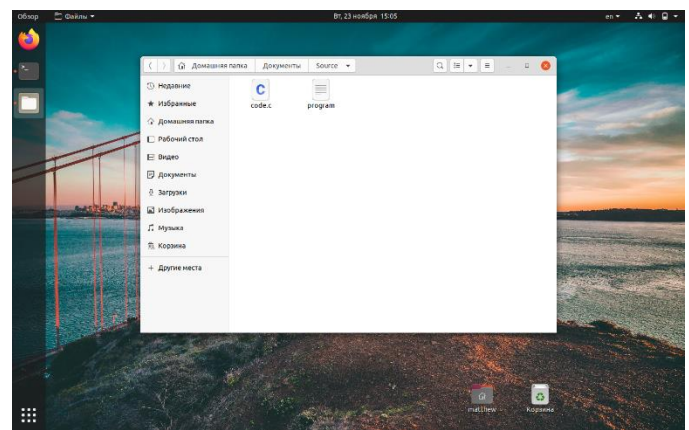
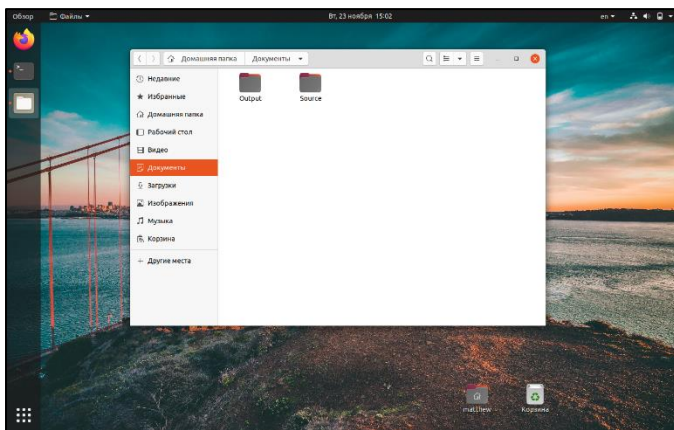


*Итог: Нужное ПО успешно установлено через терминал. Все базовые настройки выставлены верно!*

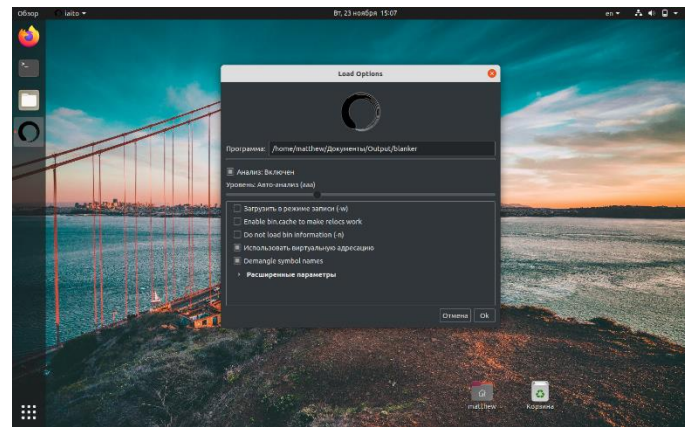
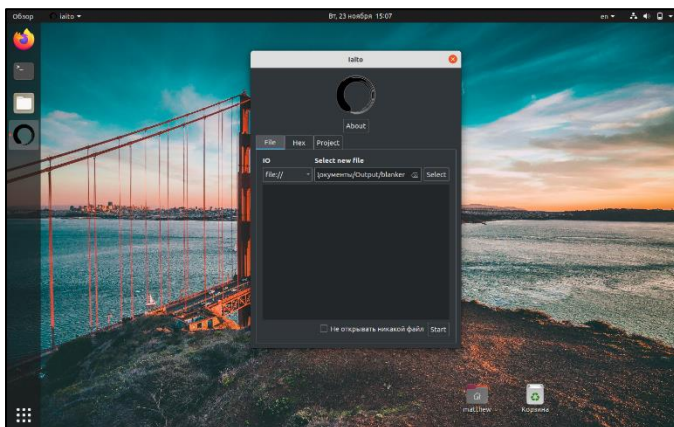


### III. Выполнение задания в ПО Radare2 (iaito), на платформе Linux Ubuntu. (→)

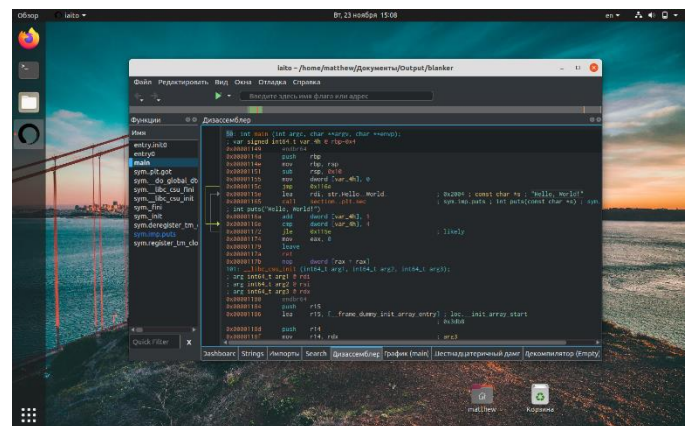
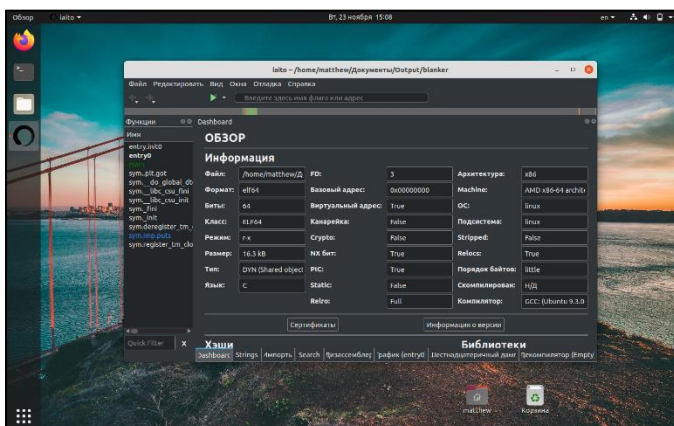
III.I. Прodelываем разветвленную структуру входного и выходного файлов, проверяем компиляцию выходного файла.



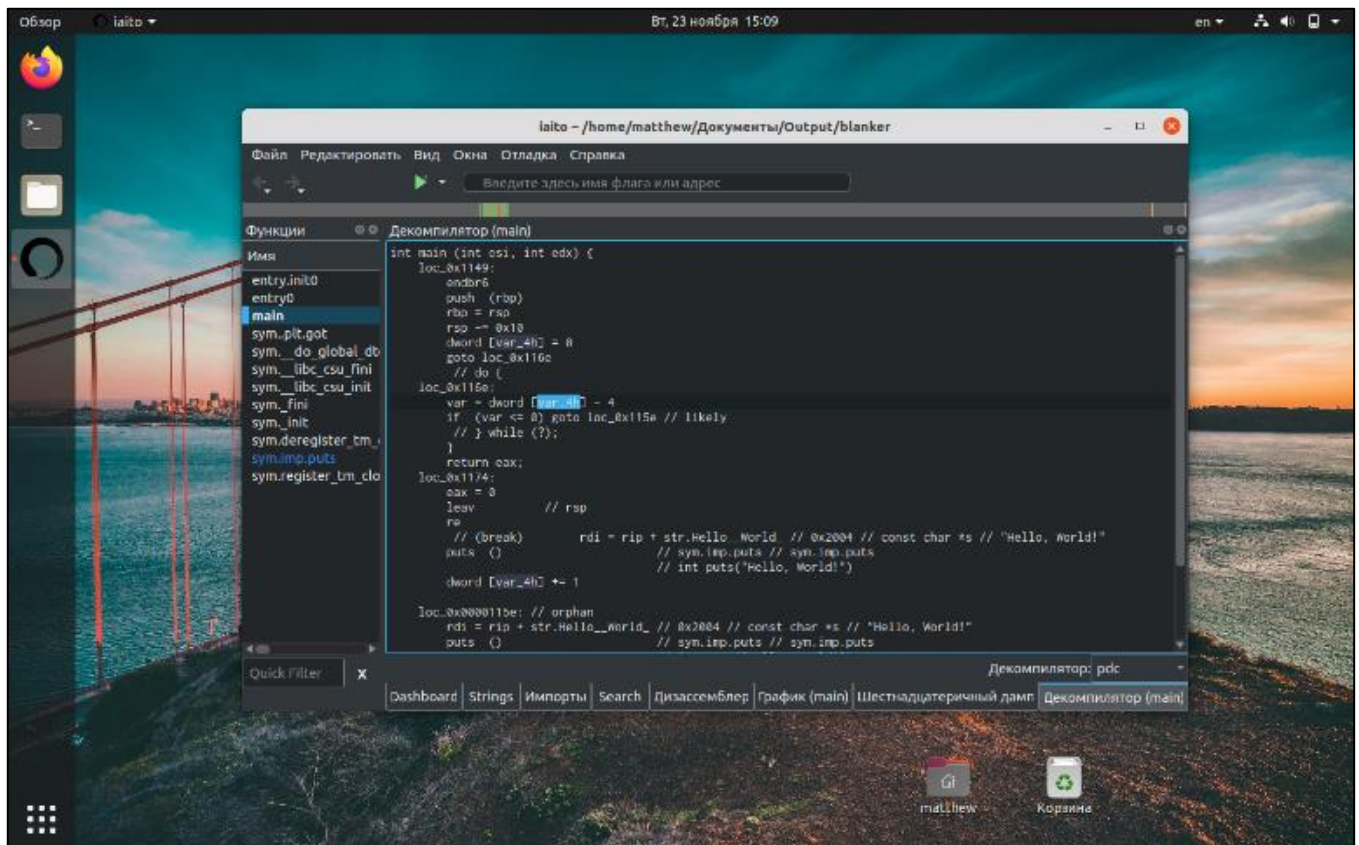
III.II. Открываем выходной файл в iaito. Настройки оставляем стандартные.



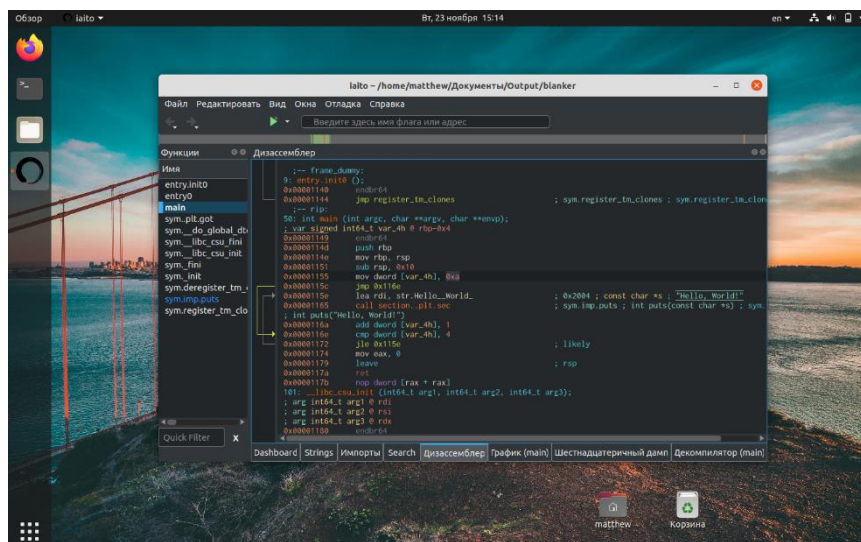
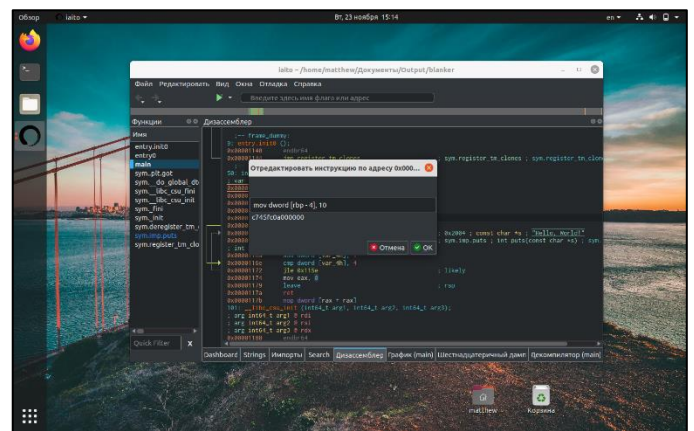
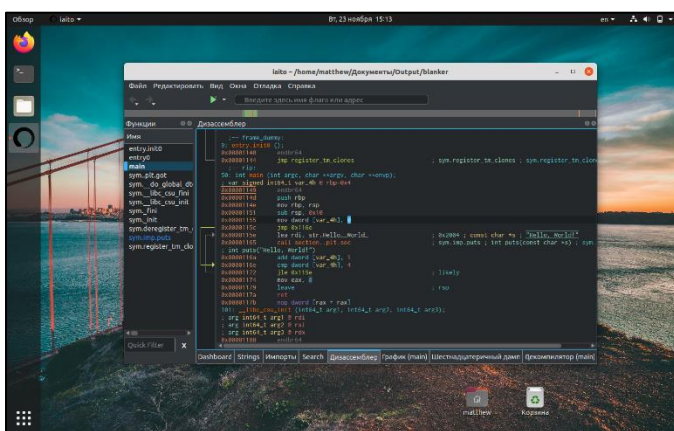
III.III. Открываем выходной файл и заходим в пункт меню «mine». Раздел «Дизассемблер».



### III.IV. Открываем раздел «Декомпилятор» (pdc). Меняем режим работы программы. (на фоне)

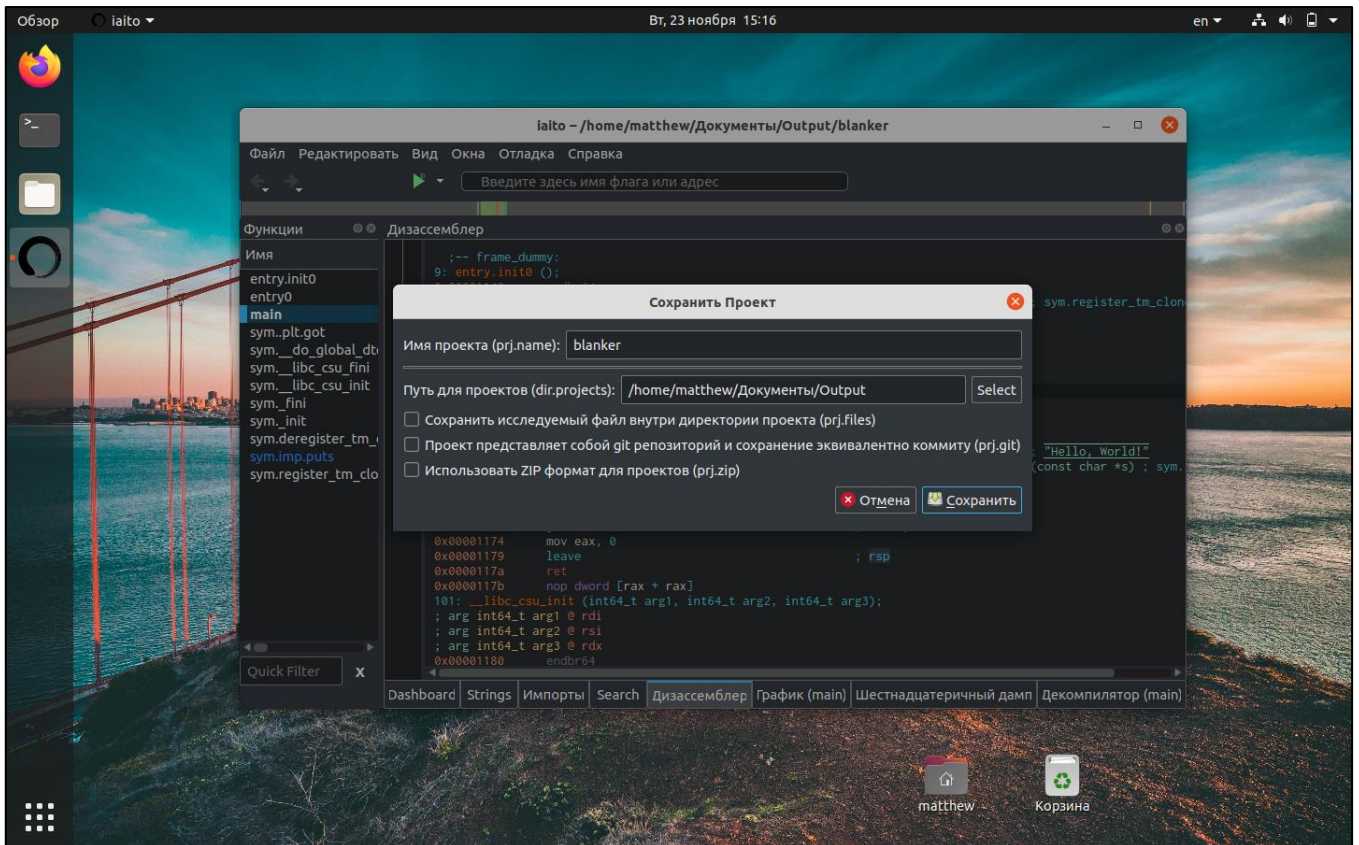


### III.V. Возвращаемся в раздел «Дизассемблер» Выделяем число «0» на строке с количеством проводимых операций (1155) и подтверждаем операцию.

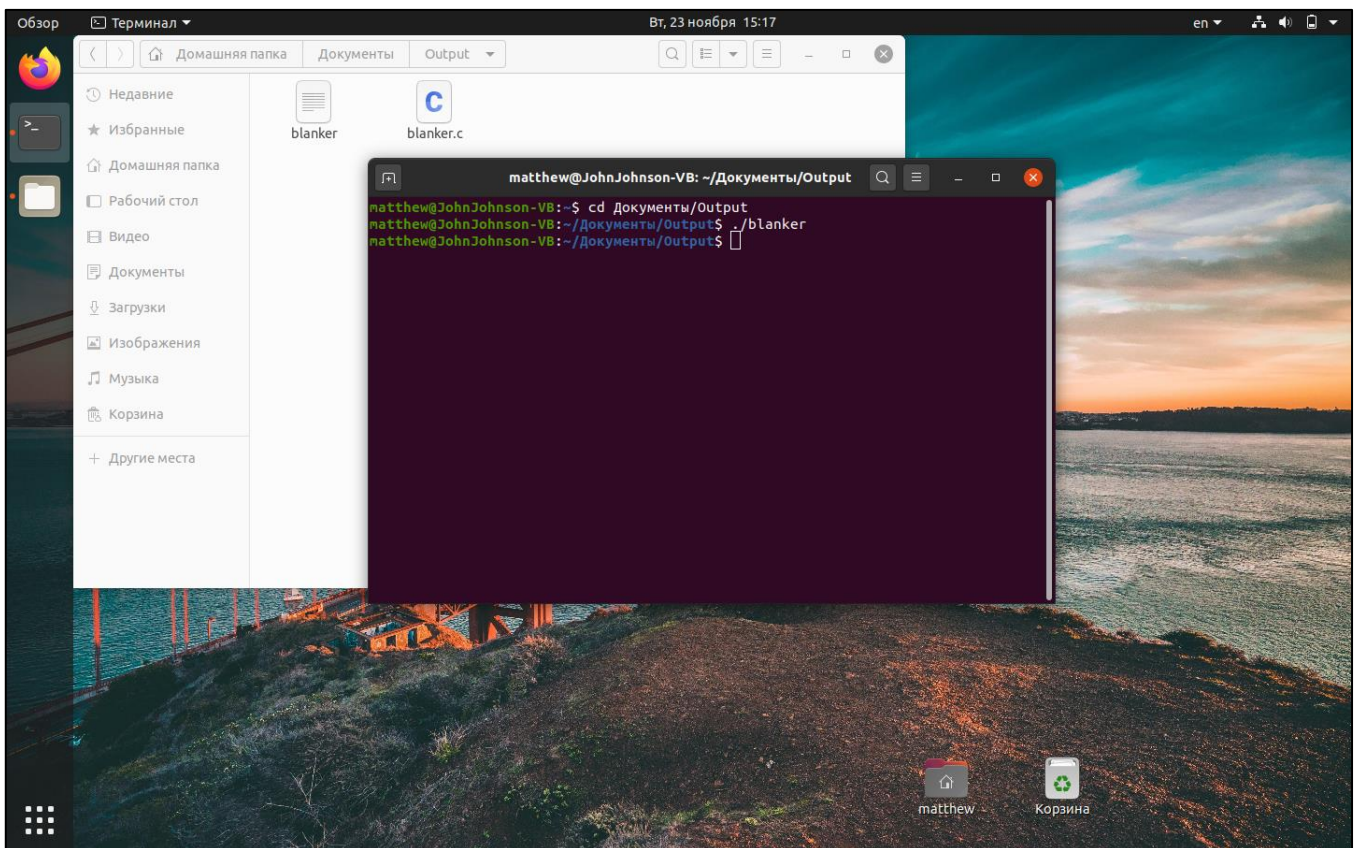




III.VI. Сохраняем сделанные нами изменения кликом по специальной кнопке «Commit Changes». Сохраняем выходной файл в папку «output».



III.VII. Проверяем сделанные нами изменения с помощью открытия выходного файла через команду терминала.



*Итог: Выходной файл получен. Работа завершена.*