**Отчет по ИДЗ №2**

**Морина Ильи БПИ 215**

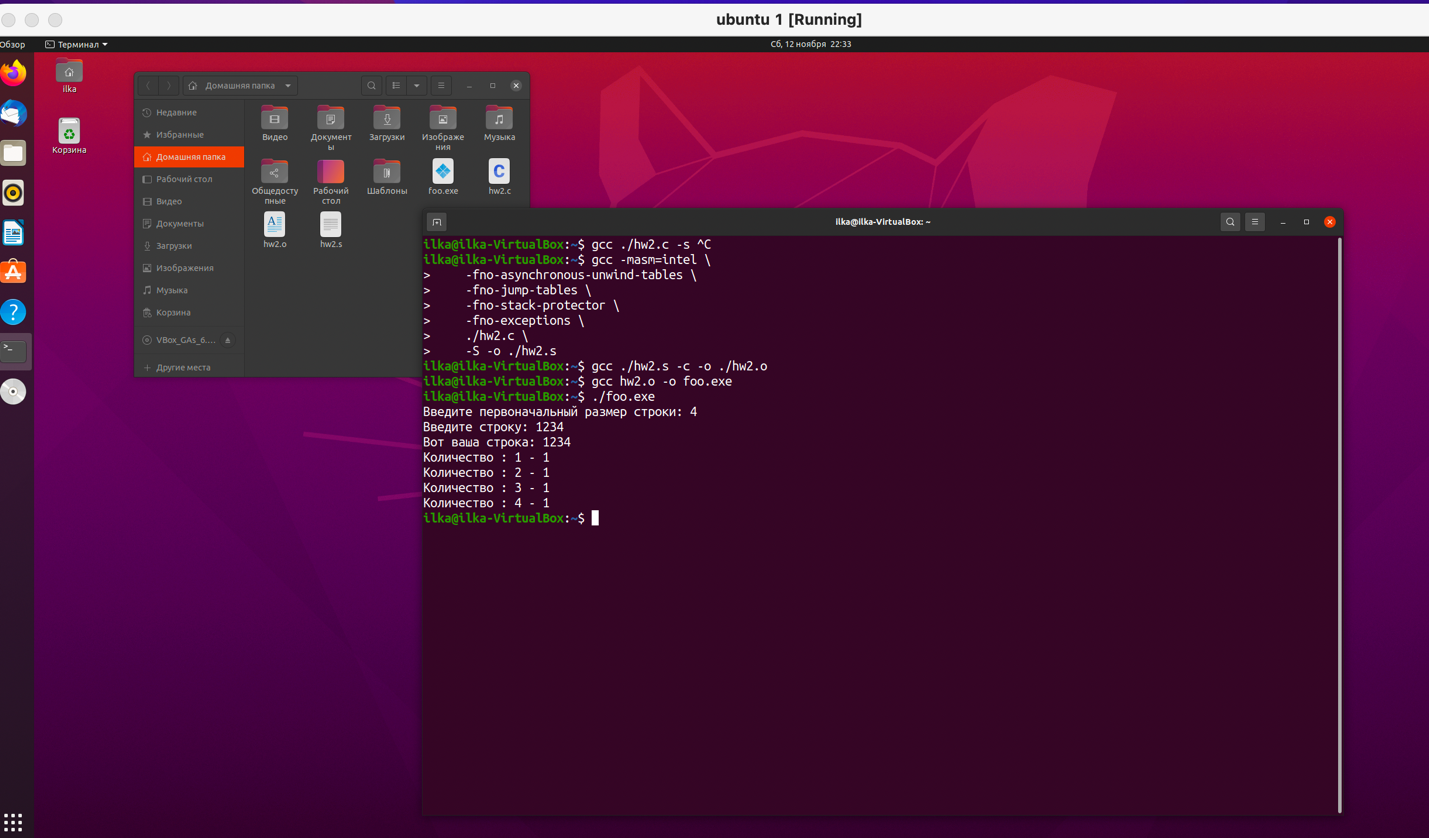
Вариант 20

**Часть 1 – создание программы.**

Первым делом написали код программы на C и разбили его на подпрограммы (функции).

Суть задачи заключается в создании программы, которая принимает строку и выводит число вхождений различных цифр.

Сассемблировали программу, создали объектный файл, создали exe-шник и запустили программу. Видим, что все работает корректно, переходим к комментирование и редактированию ассемблерного кода.



Данные программы:

Первоначальный вес (без оптимизации)

hw2.c – ﻿﻿1,4 kB (1 365 байт)

hw2.o – ﻿3,5 kB (3 464 байта)

hw2.s – 5,9 kB (5 859 байт)

Вес с оптимизацией asm кода.

hw2.c – ﻿﻿1,4 kB (1 365 байт)

hw2.o – ﻿﻿3,0 kB (3 024 байта)

hw2.s – ﻿ ﻿7,2 kB (7 195 байт) (без комментариев)

17,3 kB (17 314 байт) (с комментариями)

**Часть 2 – редактирование и оптимизация ассемблерного кода.**

**Редактирование**

1. Убираем первую строку “file “hw2.c””
2. Удаляем мусор созданный компилятором.
3. Убираем ненужные cdqe.

До редактирования было 315 строк asm кода - cтало 287.

**Оптимизация**

1. Вместо -4[rbp] используем r12d

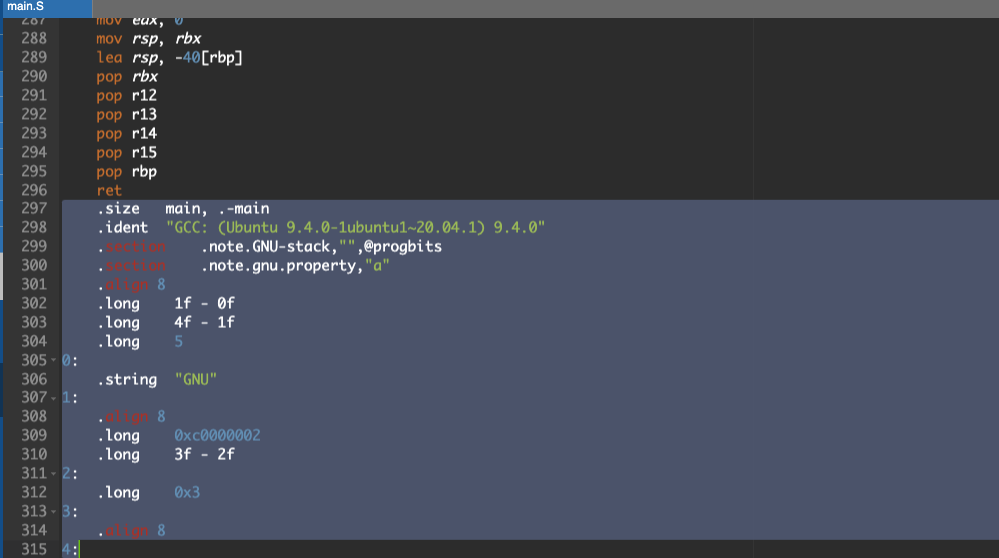
2. Превращение подобных пар

**mov eax, DWORD PTR [rdx+rax]**

**mov esi, eax**

в **mov esi, DWORD PTR [rdx+rax] (без промежуточных регистров)**

-12[rbp] в программе не нашлось, поэтому оптимизация с помощью r13d не получилась.



**Часть 3 – Пишем комментарии.**

Все комментарии находятся в файле с ассемблерным кодом (.S). Попытался привести подробное описаний (из сайтов) некоторых инструкций, те инструкции, которые встречались очень часто (mov, movsx, movzx, pop и тд) комментировал не везде, так как привел подробное их описание выше.

**Часть 4 – написание и прогон тестов.**

Пишем тесты

Тест 1 – строка из 10 символов без цифр.

Тест 2 – строка из 10 символов с 5ью разными цифрами и 5 буквами.

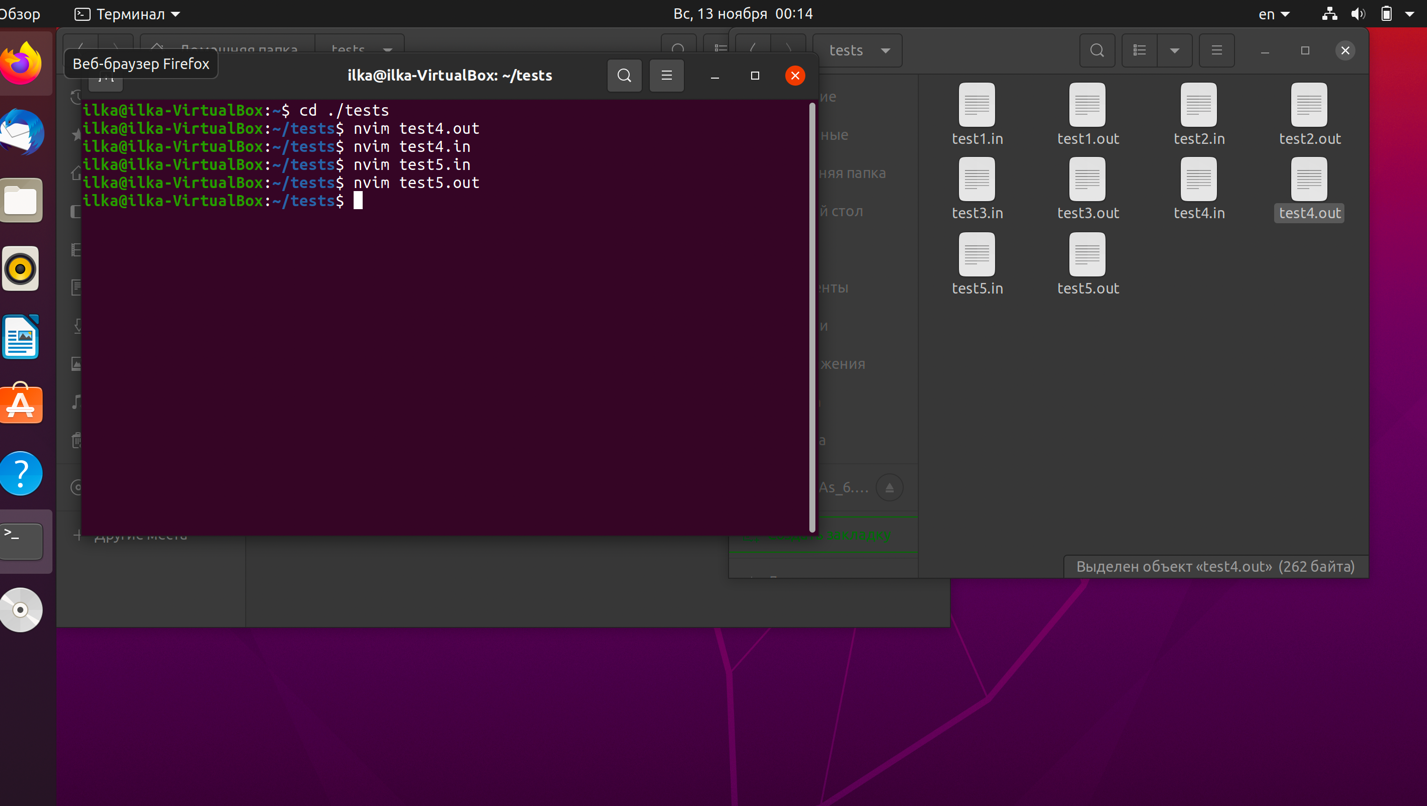
Тест 3 – строка из 10 разных цифр.

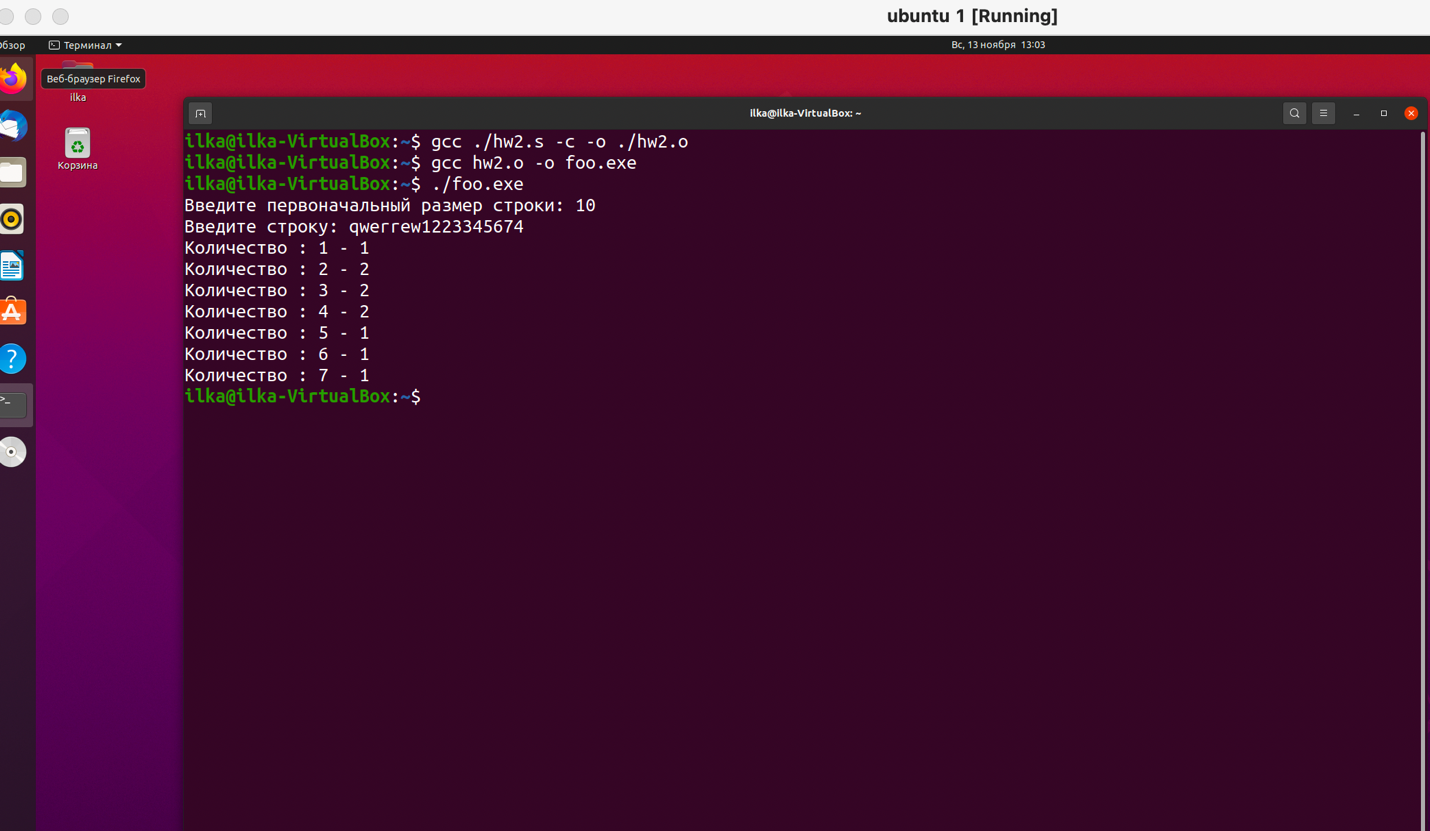
Тест 4 – строка из 10 символов есть цифры, буквы, некоторые цифры повторяются.

Тест 5 – строка из 1 цифры.

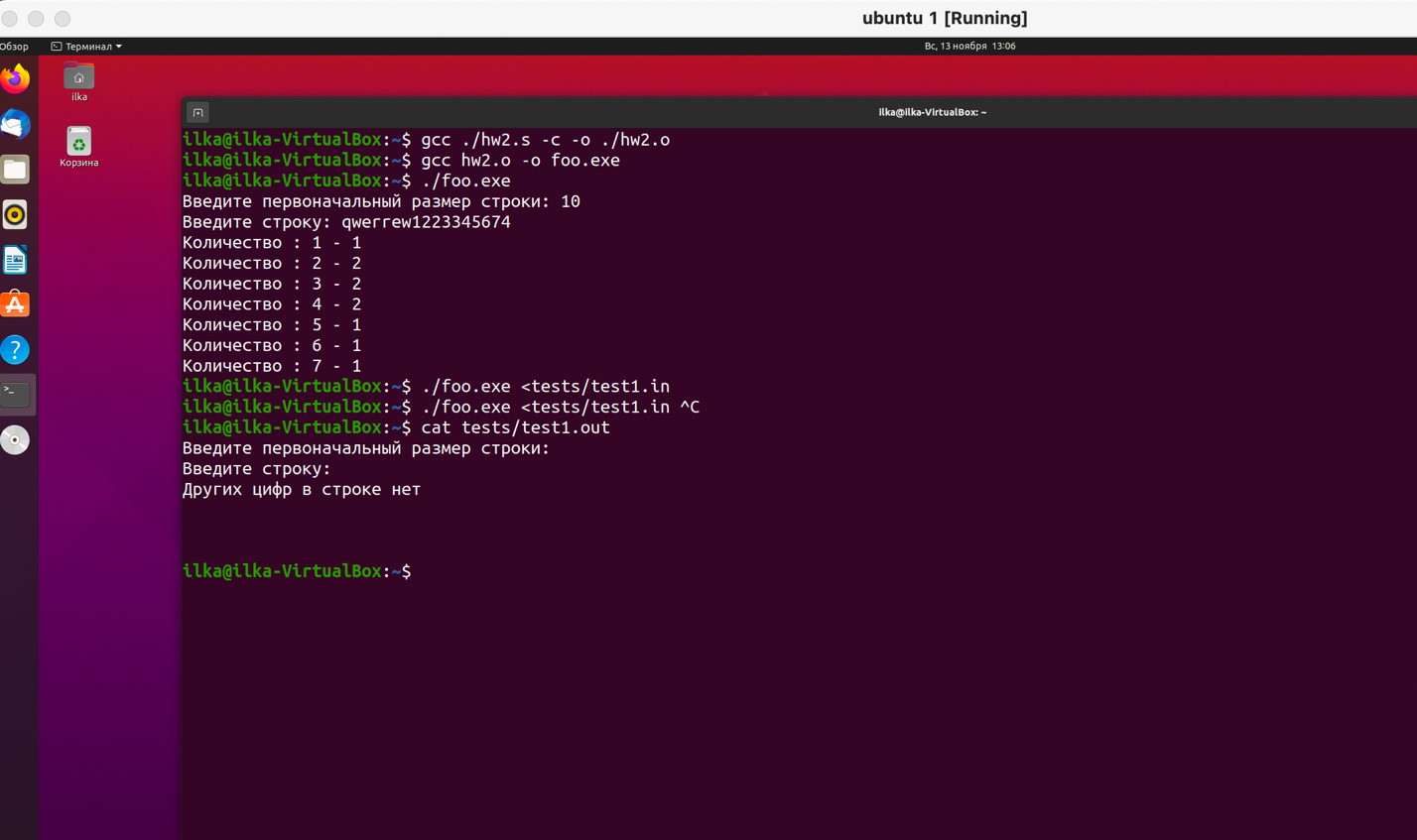
Сначала тестируем программу на C затем на ассемблерe, так как считаем программу на C корректной + она не изменяется в дальнейшем (в отличие от asm файла).

В файлы с правильными результатами тестов (.out) записываем результаты программы на C.



После редактирование и оптимизации asm кода, собрали проект с отредактированным .s файлом и проверили корректность работы.

Далее переходим к прогону тестов на новом файле.



Так делаем с каждым из 5 тестов, видим, что все работает корректно.

На всякий случай прикладываю C- код программы и asm – код.

