Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики» (СибГУТИ)

# Лабораторная работа №3

**«ОТЛАДКА ПРОГРАММ С ПОМОЩЬЮ GDB»**

Выполнил: студент группы ИП-813

Бурдуковский И.А.

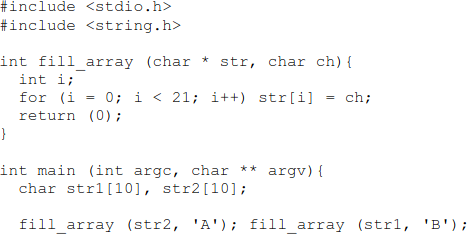
Проверил: профессор кафедры ВС

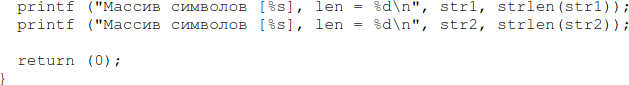
Мамойленко Сергей Николаевич

Новосибирск, 2022 г.

# Задание на лабораторную работу

1. Подготовьте файл с исходным кодом программы, который представлен ниже:

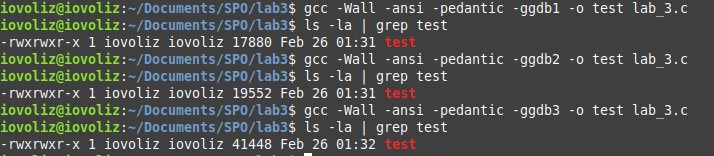




1. Используя компилятор GCC скомпилируйте подготовленную программу с включением отладочной информации (всех уровней). На сколько изменяется размер исполняемого файла в зависимости от уровня детализации отладочной информации?
2. Выделите отладочную информацию в отдельный файл с включением ссылки на него в исполняемый файл. Проведите отладку программы в GDB.
3. Выделите отладочную информацию в отдельный файл без включения ссылки на него в исполняемый файл. Проведите отладку программы в GDB.
4. Используя отладчик GDB, объясните результат выполнения программы. Почему программа выдает в обоих случаях длину строки, отличную от длины массива? Почему длины строк различные? Что произойдет если в функции fill\_array условие окончания цикла изменить на i<25?
5. Используя программу, которую Вы разрабатывали в рамках лабораторной работы «Многонитиевые программы» продемонстрируйте процесс отладки подобных программ с применением GDB.

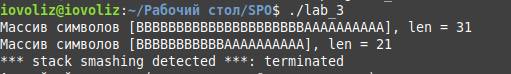
# Результаты работы

1. Между уровнем детализации отладочной информации и размером файла существует прямая зависимость. Чем выше уровень – тем “тяжелее” файл.

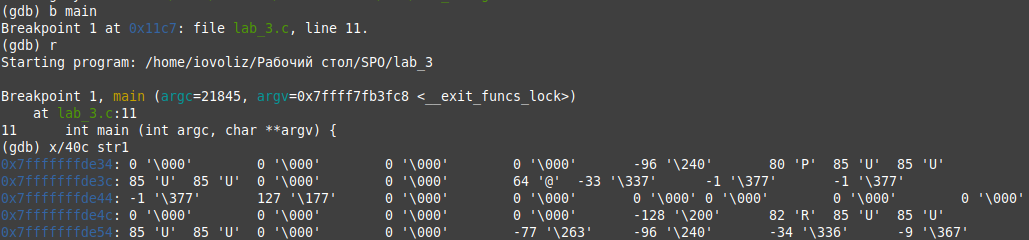


1. Для выделения отладочной информации в файл используем:
2. Для включения ссылки на файл с отладочной информацией в исполняемый файл:

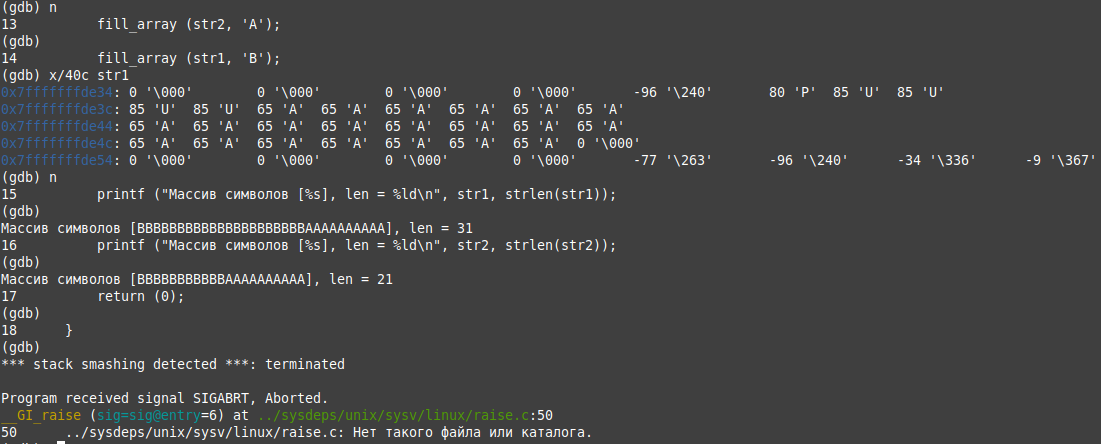


1. Результат выполнения программы:

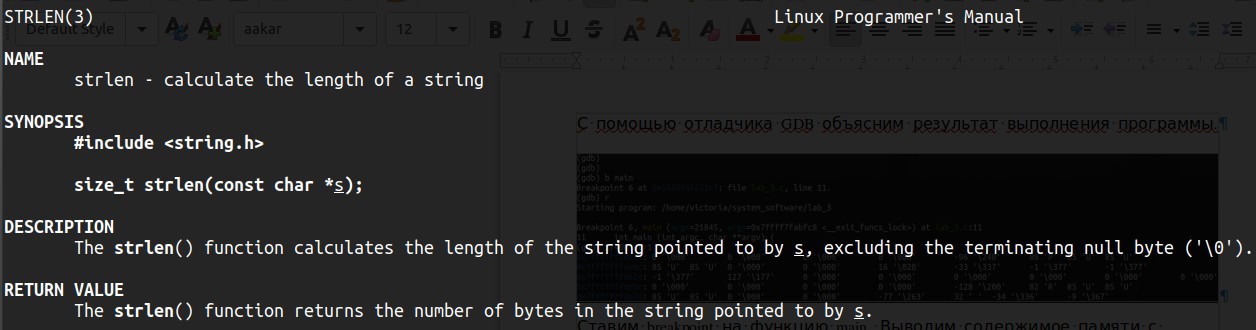
С помощью отладчика GDB объясним полученный результат.

Ставим breakpoint на функцию main. Выводим содержимое памяти с адреса str1 до заполнения строк.

Теперь смотрим содержимое после заполнения строк. Видно, что при заполнении str2 программа вышла за допустимые границы и str2 частично перезаписалась при заполнении str1.



Обратимся к мануалу strlen:



То есть strlen считает количество байт, пока не встретит нулевой (‘\0’). Так как мы вышли за границы строки при заполнении, нужный нулевой символ перезаписался. Поэтому strlen возвращает некорректную длину строки.

Если условие окончания цикла изменить на i < 25, программа перезапишет на 4 байта больше. В текущей ситуации, strlen встретит ‘/0’ сразу после этих 4-х байт. Поэтому при выводе длина str1 будет равна 35, а str2 25.

1. Отладка многопоточного приложения:

