Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Высшая школа интеллектуальных систем и суперкомпьютерных технологий

**Лабораторная работа № 4**

**Дисциплина**: Низкоуровневое программирование

**Тема:** Раздельная компиляция

Вариант №3

Выполнила

студентка гр. 3530901/10003 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.А. Коняшова

(подпись)

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д.А. Коренев

(подпись)

“ ” 2022 г.

Санкт-Петербург

2022

Оглавление

[**Техническое задание** 3](#_Toc121661667)

[**1. Программа на языке C** 4](#_Toc121661668)

[**2. Сборка программы «по шагам»** 6](#_Toc121661669)

[Препроцессирование 6](#_Toc121661671)

[Компиляция 8](#_Toc121661672)

[Ассемблирование 11](#_Toc121661673)

[Компоновка 19](#_Toc121661674)

[**3. Создание статической библиотеки и make-файлов** 24](#_Toc121661675)

[**4. Вывод** 29](#_Toc121661676)

**Техническое задание**

1. Изучить методические материалы, опубликованные на сайте курса.

2. Установить пакет средств разработки “SiFive GNU Embedded Toolchain” для RISC-V.

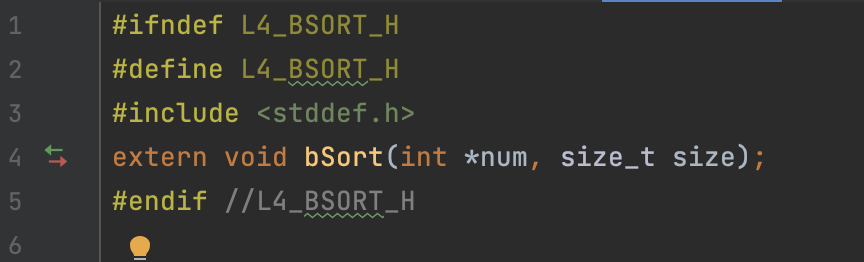
3. На языке C разработать функцию сортировки «пузырьком». Поместить определение функции в отдельный исходный файл, оформить заголовочный файл. Разработать тестовую программу на языке C.

4. Собрать программу «по шагам». Проанализировать выход препроцессора и компилятора. Проанализировать состав и содержимое секций, таблицы символов, таблицы перемещений и отладочную информацию, содержащуюся в объектных файлах и исполнимом файле.

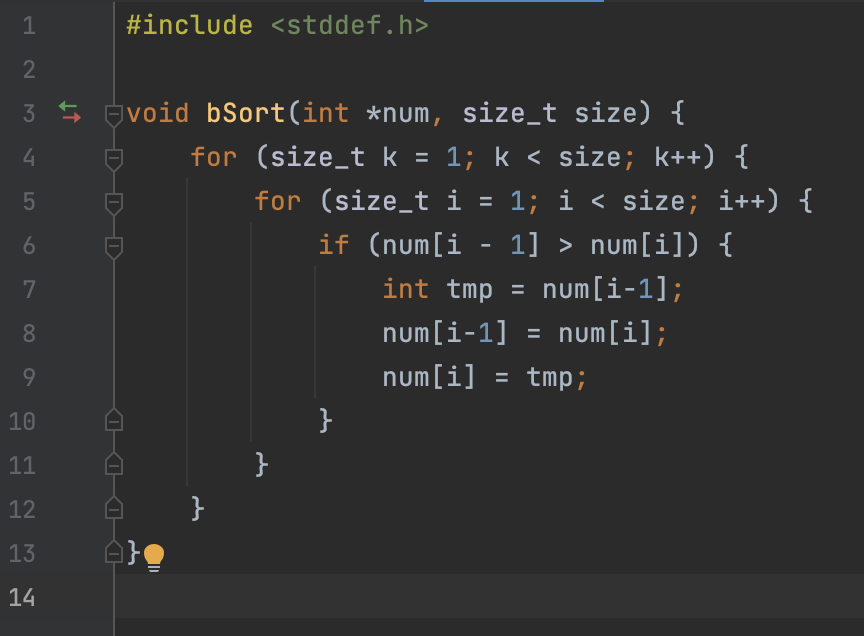
5. Выделить разработанную функцию в статическую библиотеку. Разработать make-файлы для сборки библиотеки и использующей ее тестовой программы. Проанализировать ход сборки библиотеки и программы, созданные файлы зависимостей.

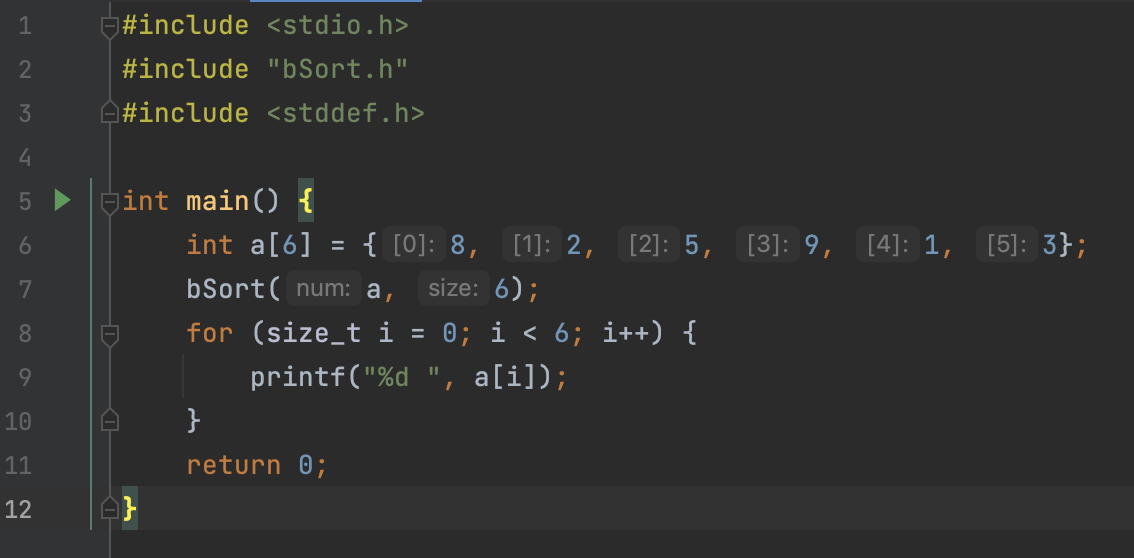
**1. Программа на языке C**

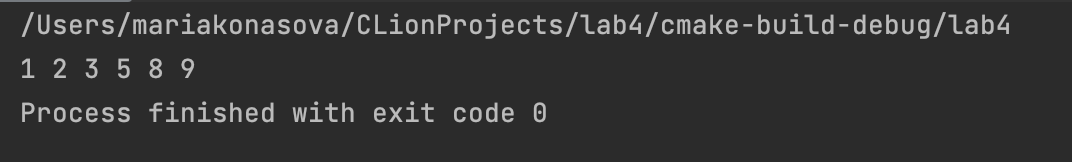
Листинг 1.1. Заголовочный файл bSort.h



Листинг 1.2. Основной файл bSort.c



Листинг 1.3. Тестовая программа main.c



**2. Сборка программы «по шагам»**

Для осуществления раздельной компиляции под архитектуру riscv32 соберем пакет «riscv-gnu-toolchain» со стандартной библиотекой языка Си «Newlib», который возьмём с официального гитхаб-репозитория («github.com/riscv-collab/riscv-gnu-toolchain»).

Препроцессирование

Сначала произведём препроцессирование файлов «main.c» и «bSort.c» посредством следующих команд:

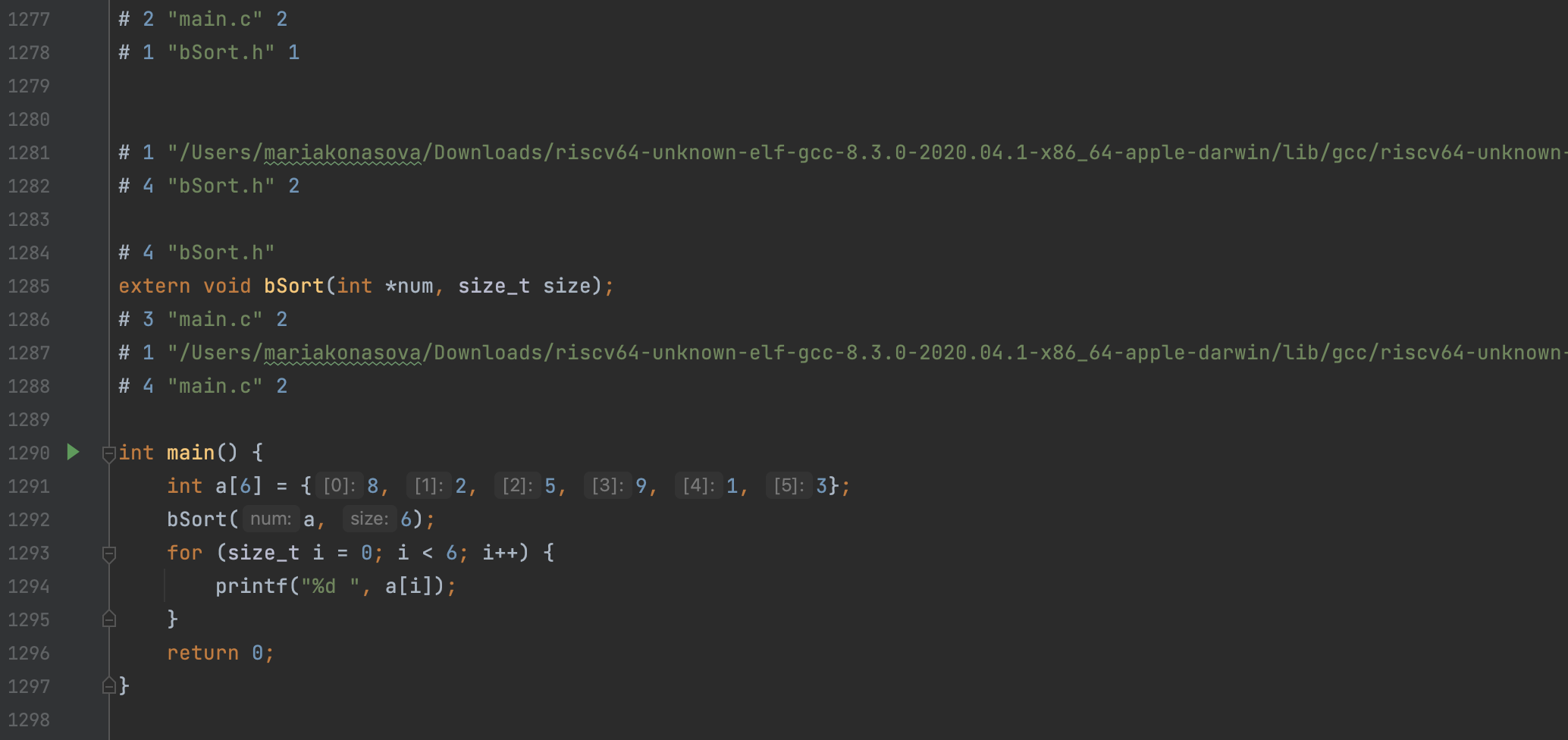
/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-gcc-8.3.0 -march=rv32imac -mabi=ilp32 -O1 -E main.c -o main.i

/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-gcc-8.3.0 -march=rv32imac -mabi=ilp32 -O1 -E bSort.c -o bSort.i

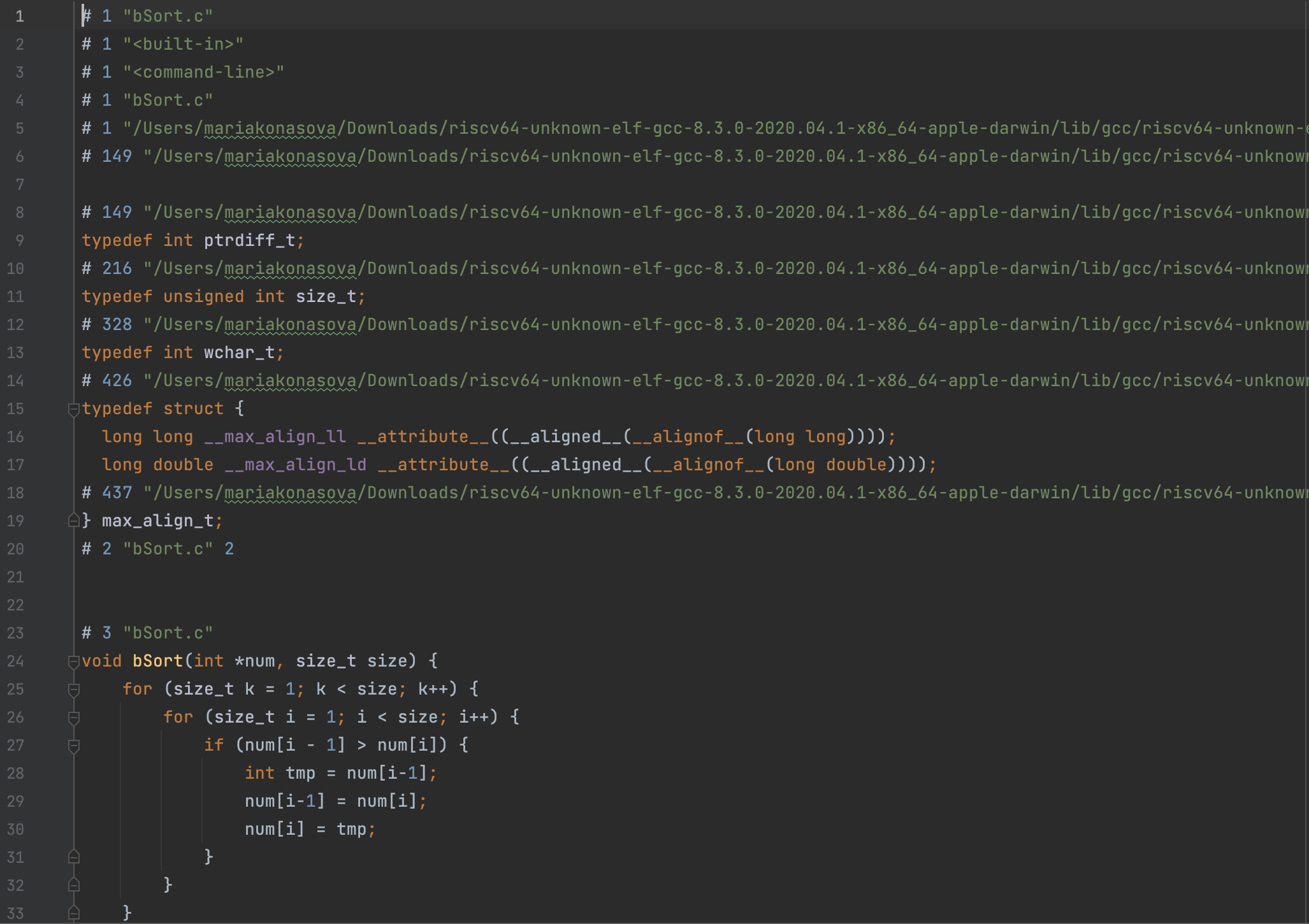
Результат препроцессирования содержится в файлах main.i и bSort.i*.* По причине того, что main.c содержит заголовочный файл стандартной библиотеки языка С stdio.h, результат препроцессирования этого файла имеет достаточно много добавочных строк.

Листинг 2.1. Файл main.i (фрагменты)





Листинг 2.2. Файл bSort.i



Компиляция

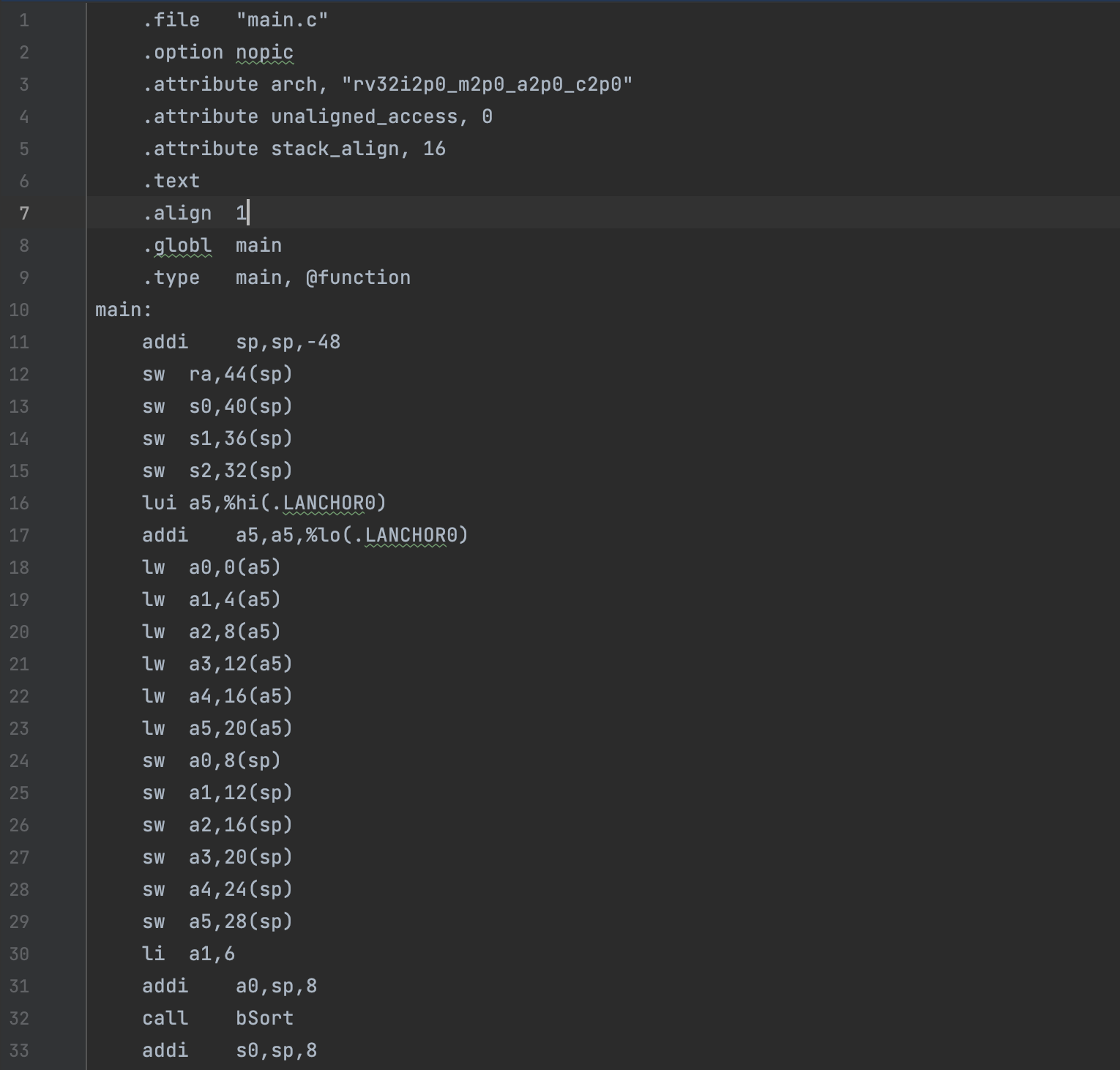
Далее произведем компиляцию файлов «main.i» и «bSort.i», полученных в результате препроцессирования, следующими командами:

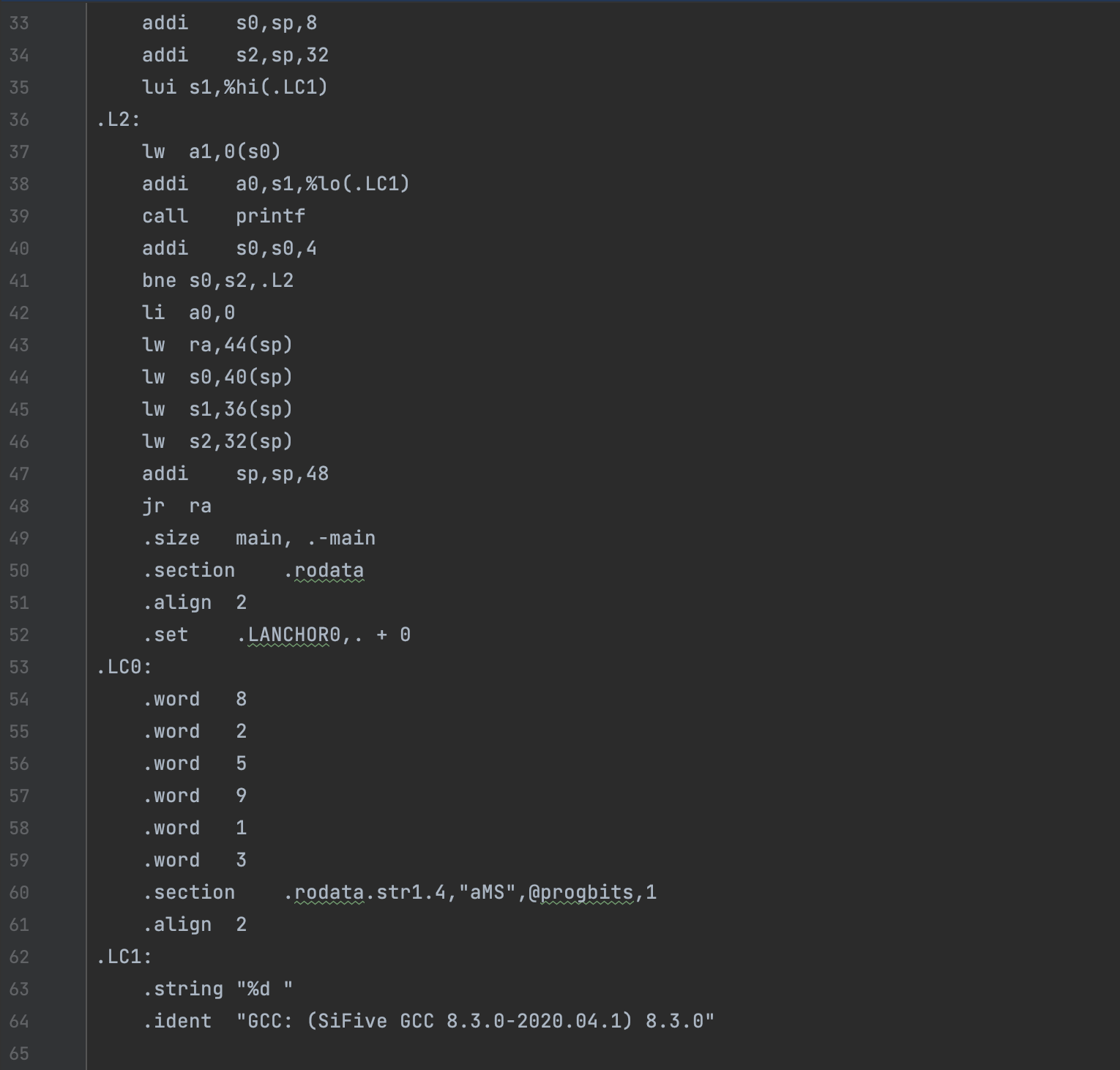
/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-gcc-8.3.0 -march=rv32imac -mabi=ilp32 -O1 -S main.i -o main.s

/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-gcc-8.3.0 -march=rv32imac -mabi=ilp32 -O1 -S bSort.i -o bSort.s

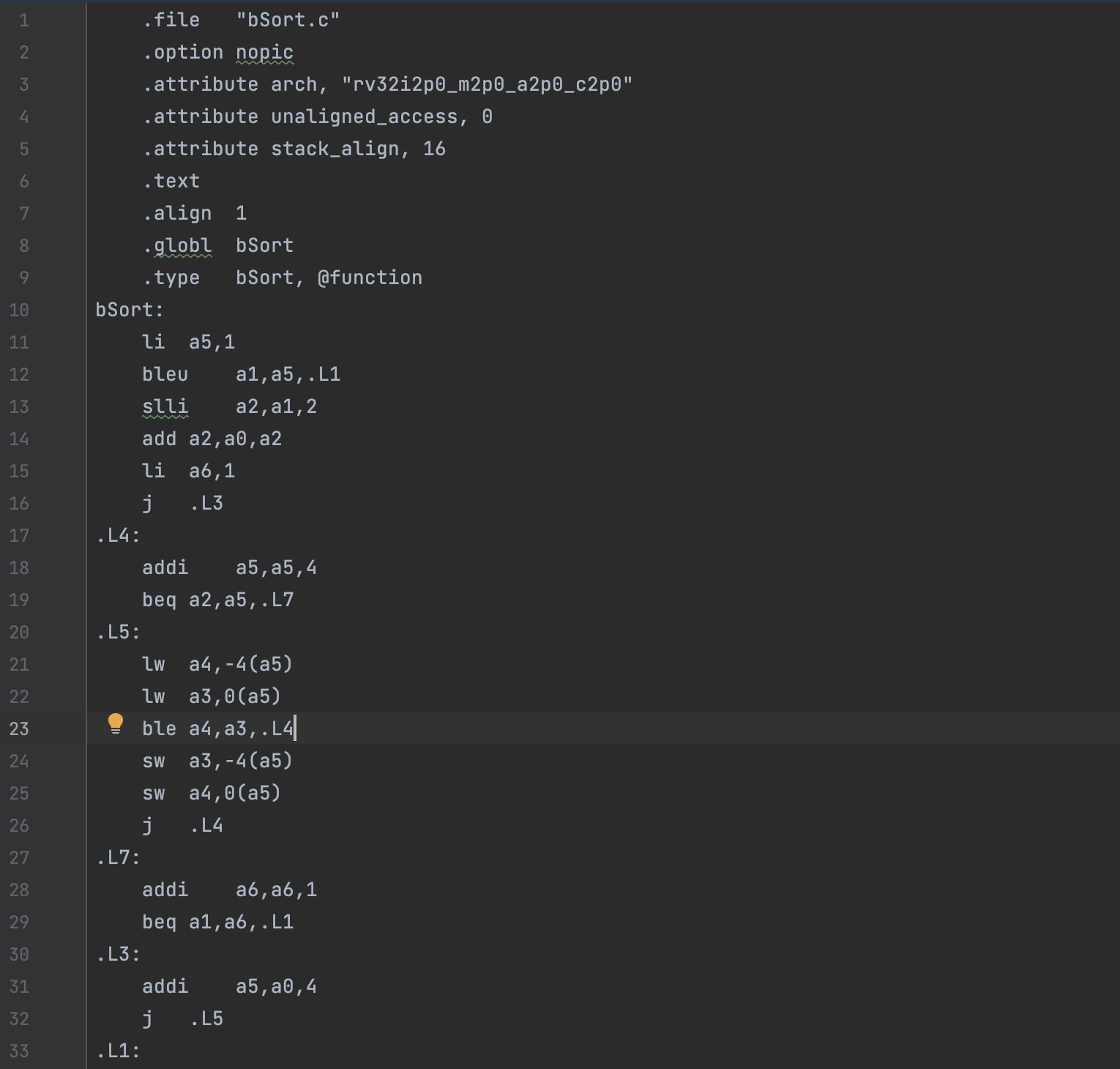
Наибольший интерес представляет файл main.s, так как в нем можно заметить обращение к подпрограмме bSort(значение регистра *ra*, содержащее адрес возврата из main, сохраняется на время вызова в стеке).

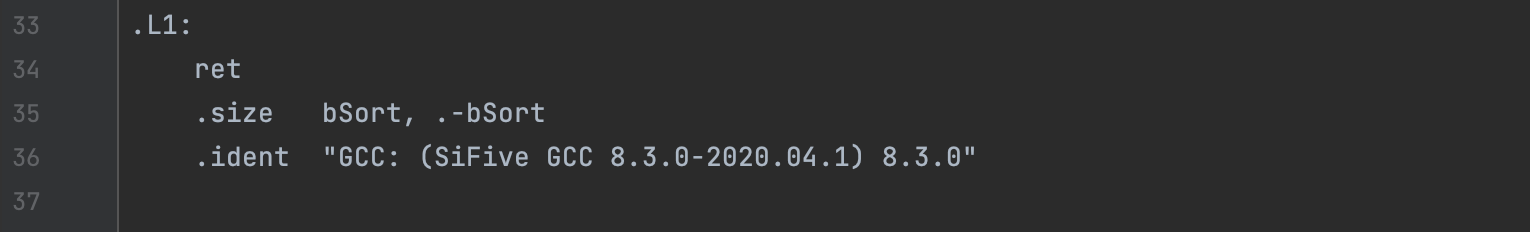
Листинг 2.3. Файл main.s





Листинг 2.4. Файл bSort.s





Ассемблирование

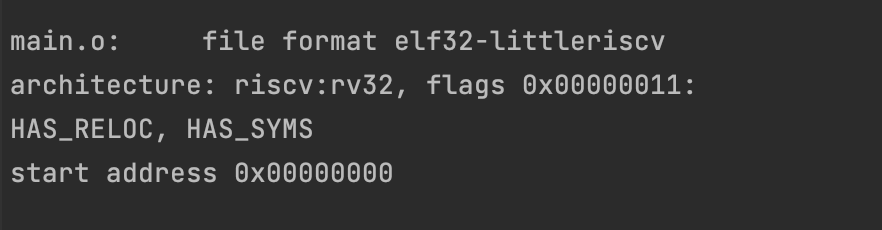
Произведём ассемблирование файлов «main.s» и «bSort.s» следующими командами:

/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-gcc-8.3.0 -march=rv32imac -mabi=ilp32 -v -c main.s -o main.o

/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-gcc-8.3.0 -march=rv32imac -mabi=ilp32 -v -c bSort.s -o bSort.o

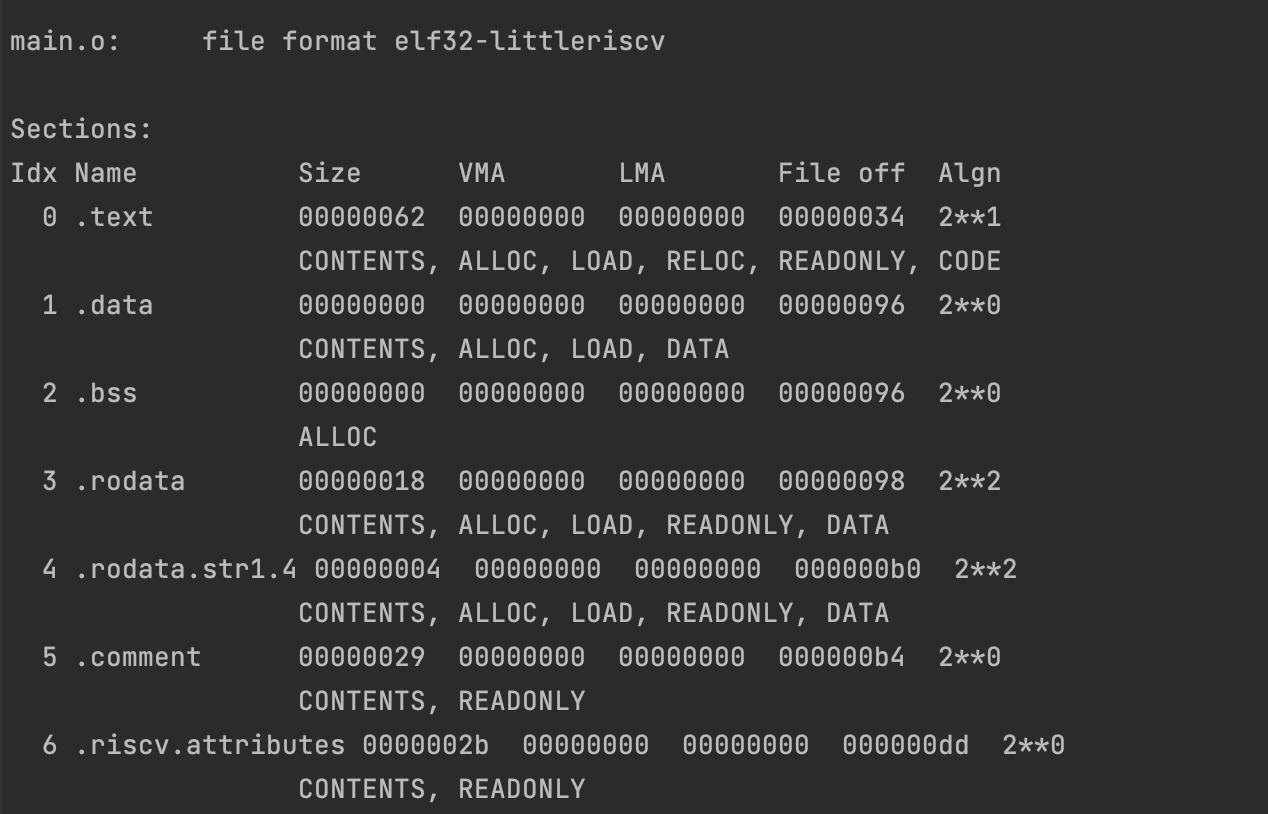
Листинг 2.5. Информация о main.o

/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf -objdump -f main.o



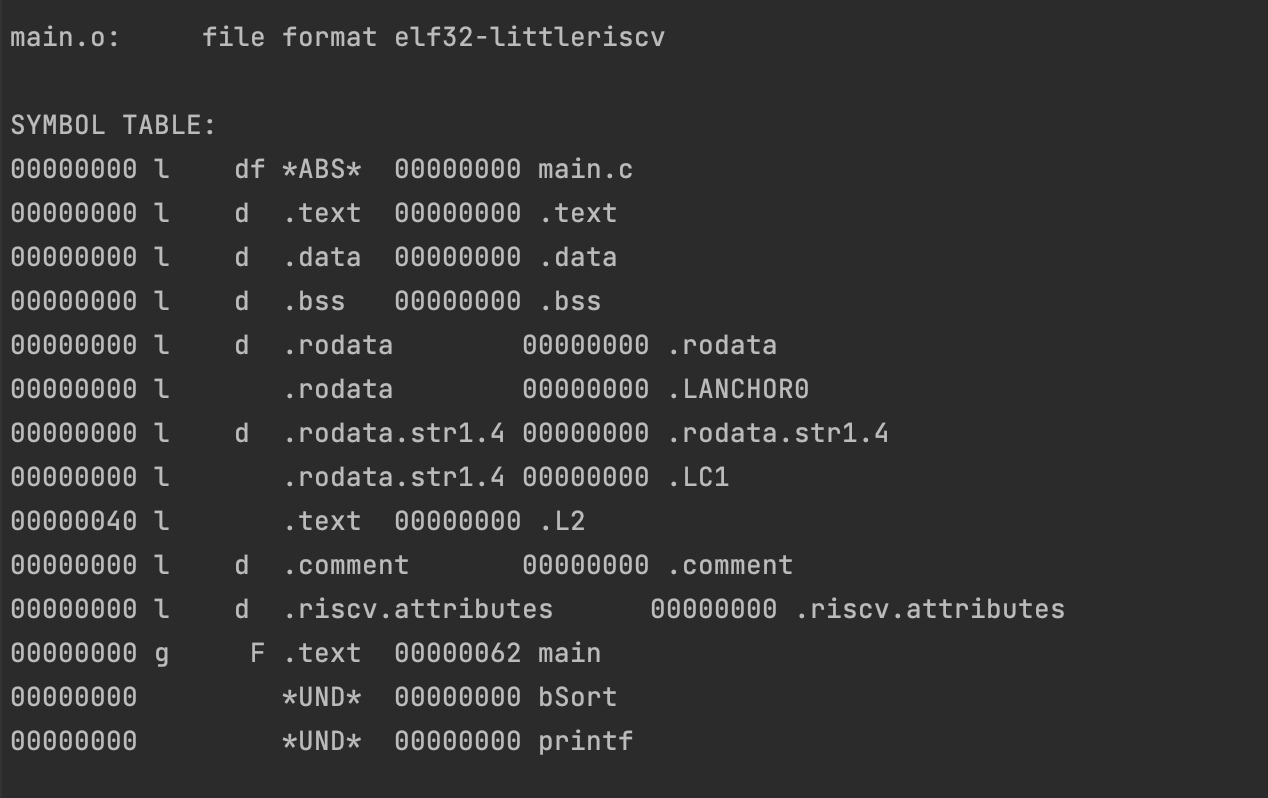
Листинг 2.6. Заголовки секций main.o

/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-objdump –h main.o



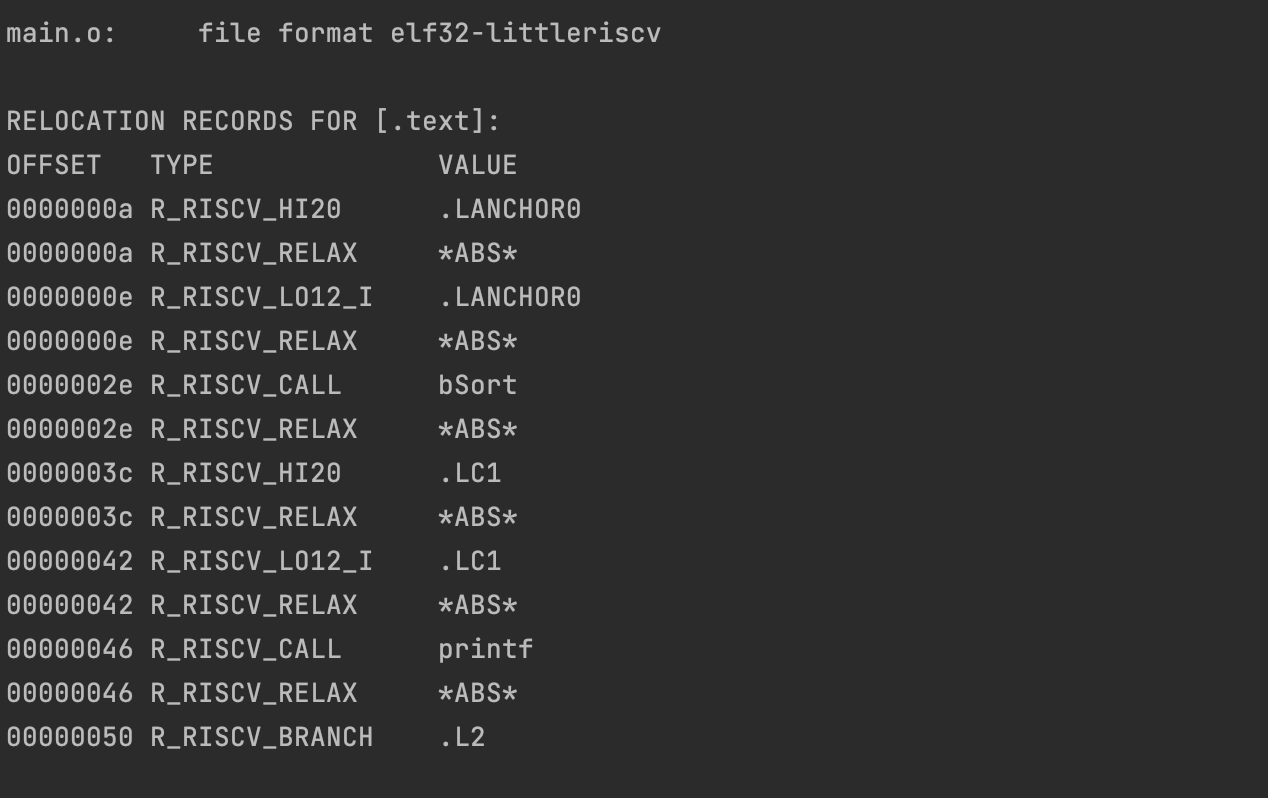
Листинг 2.7. Таблица символов файла main.o

/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-objdump –t main.o



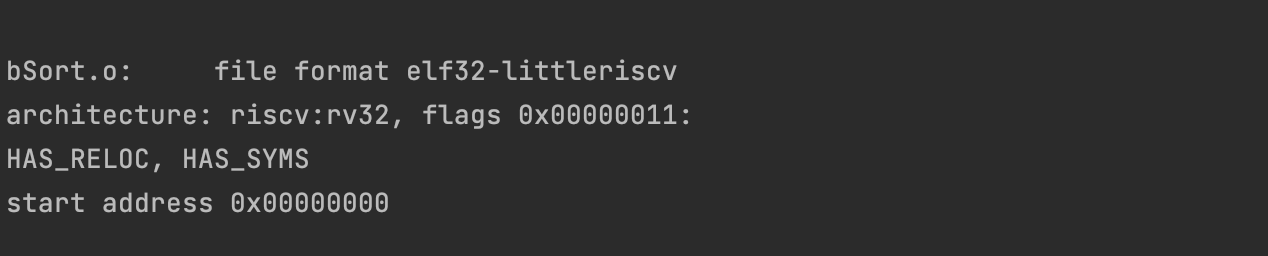
Листинг 2.8. Таблица перемещений файла main.o

/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-objdump -r main.o



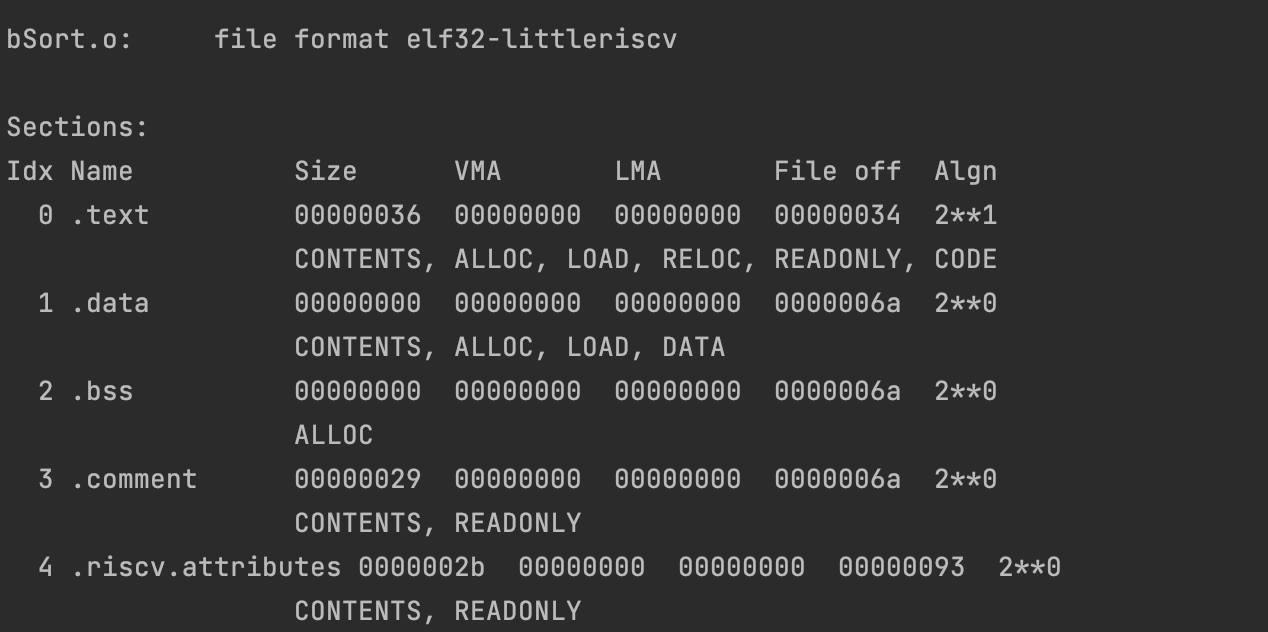
Листинг 2.9. Информация о bSort.o

/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf -objdump –f bSort.o



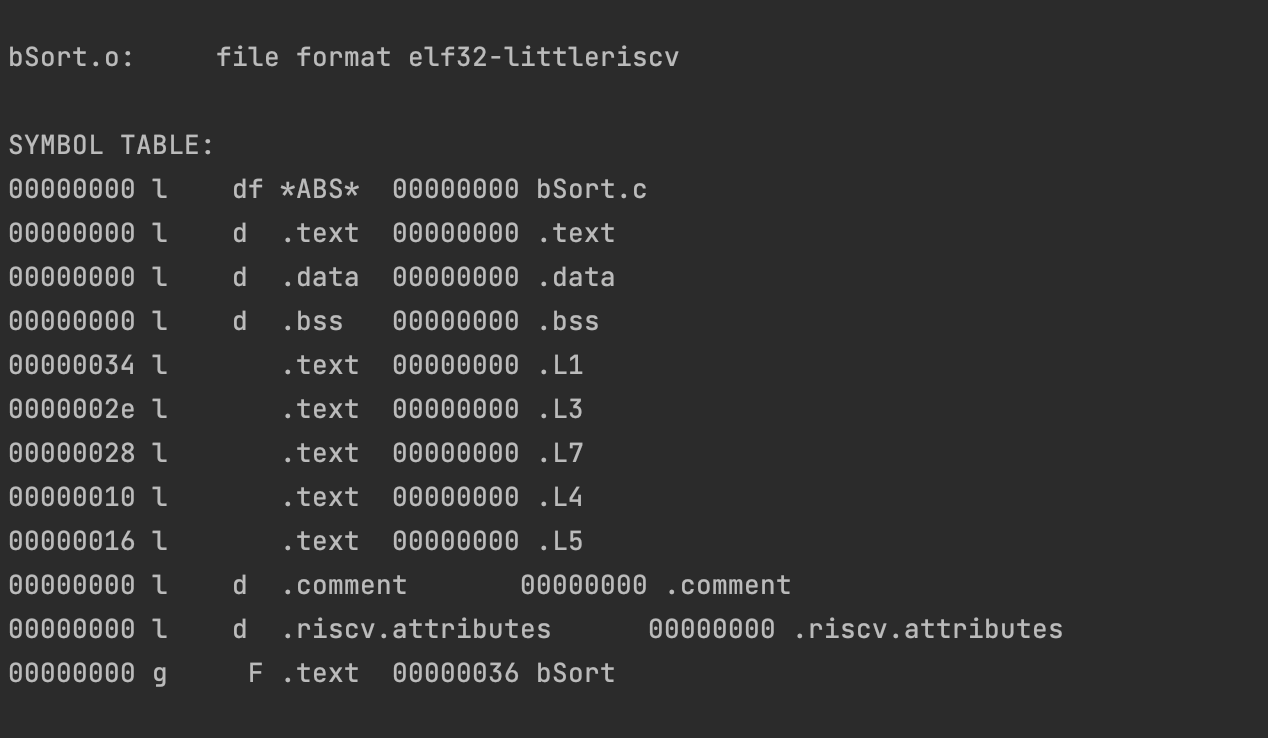
Листинг 2.10. Заголовки секций bSort.o

/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-objdump –h bSort.o



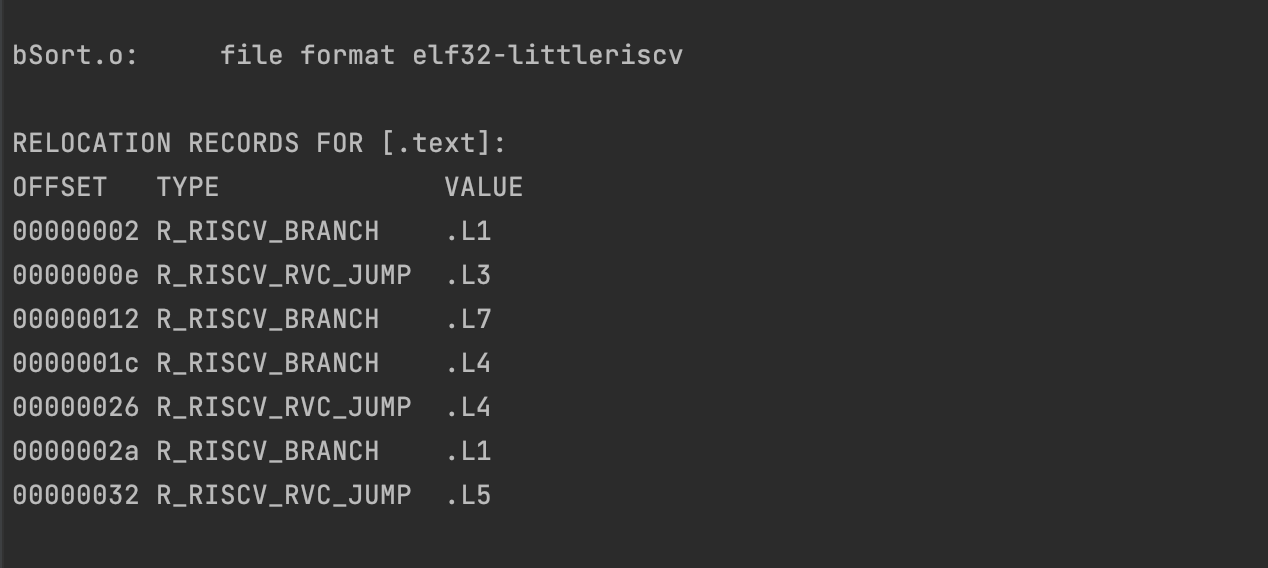
Листинг 2.11. Таблица символов файла bSort.o

/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-objdump –t bSort.o



Листинг 2.12. Таблица перемещений файла bSort.o

/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-objdump -r bSort.o

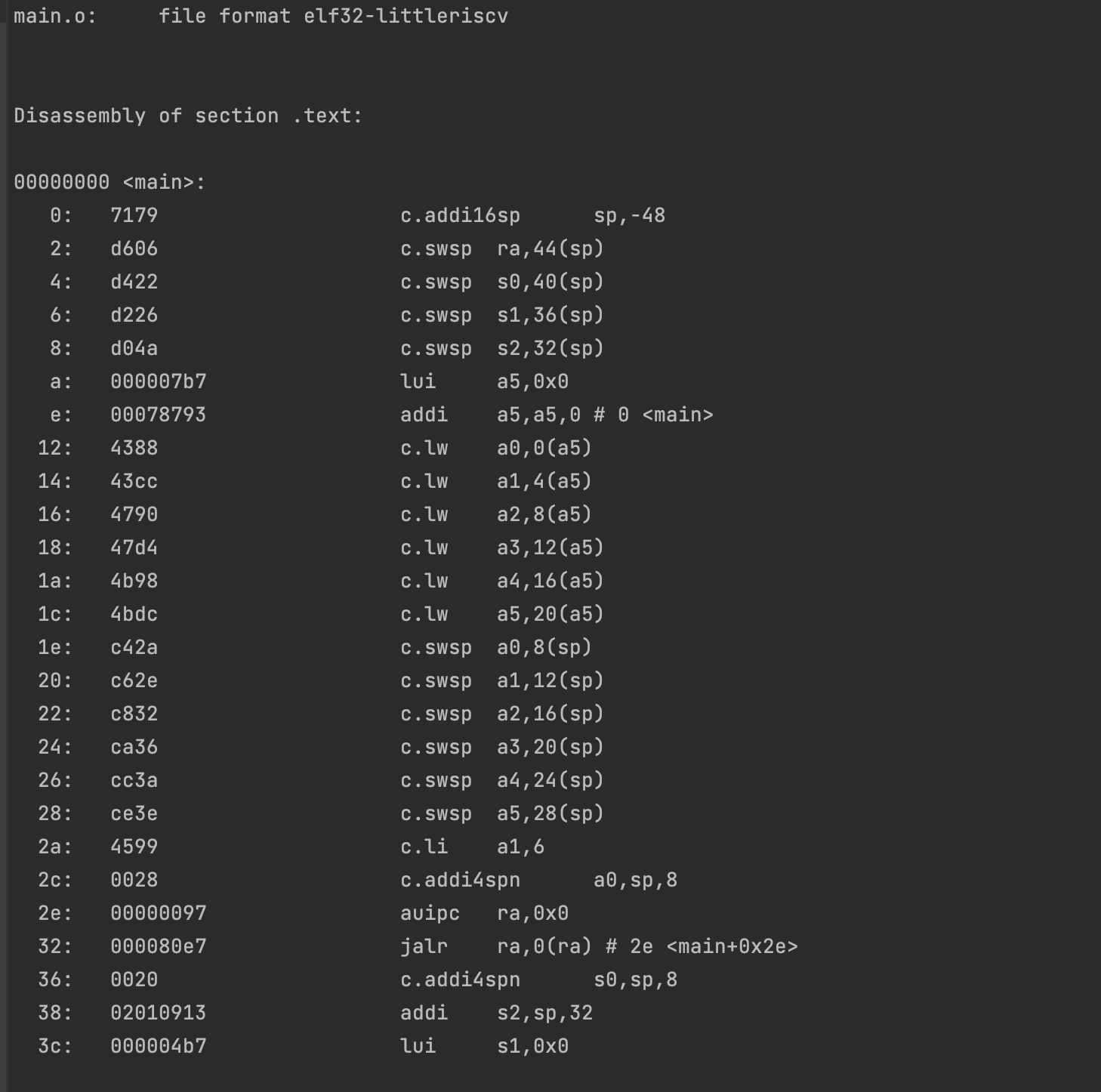


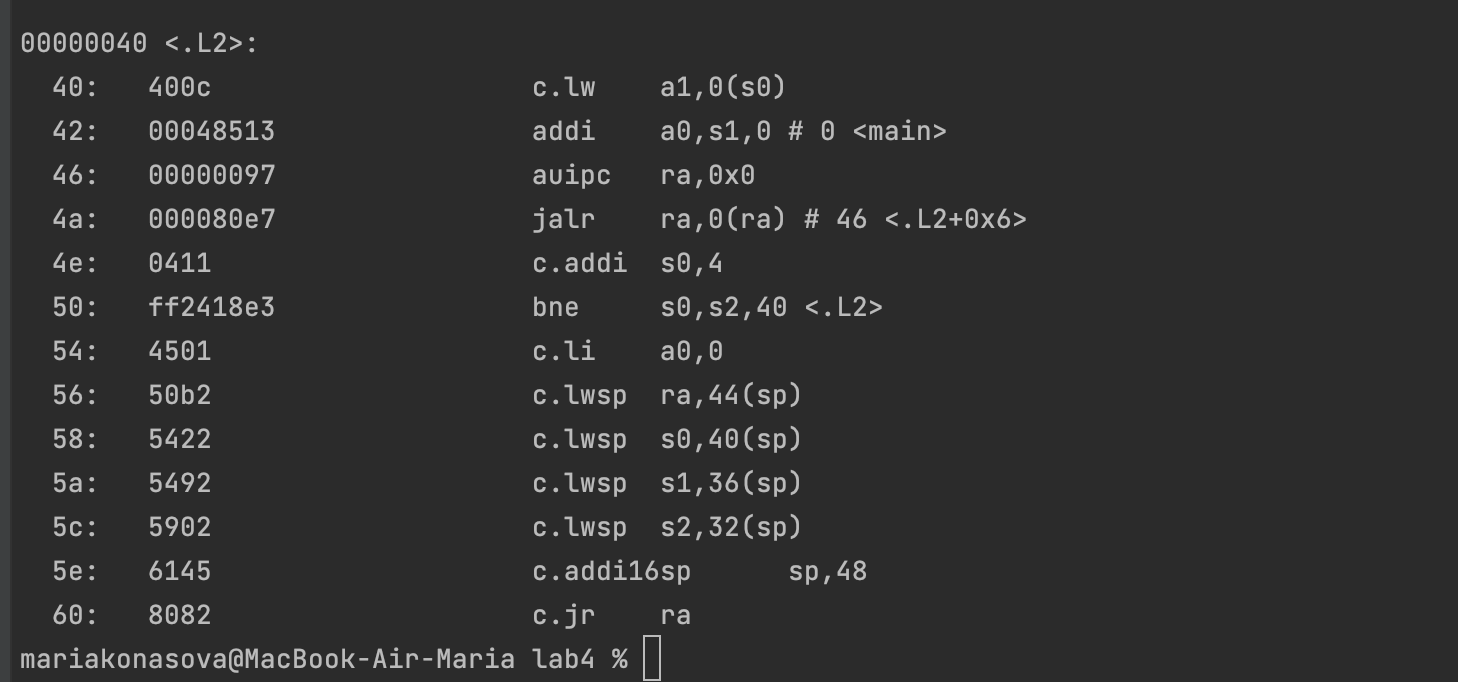
В таблице символов main.o имеется запись: символ “bSort” типа \*UND\*. Эта запись означает, что символ использовался в ассемблерном коде, из которого был получен данный объектный файл, но не был определен; ассемблер сделал вывод о том, что символ должен быть определен где-то еще (адреса вызовов этих функций будут подставлены во время компоновки при помощи таблицы перемещений) и отразил это в таблице символов. То же самое относится и к символу “printf”.

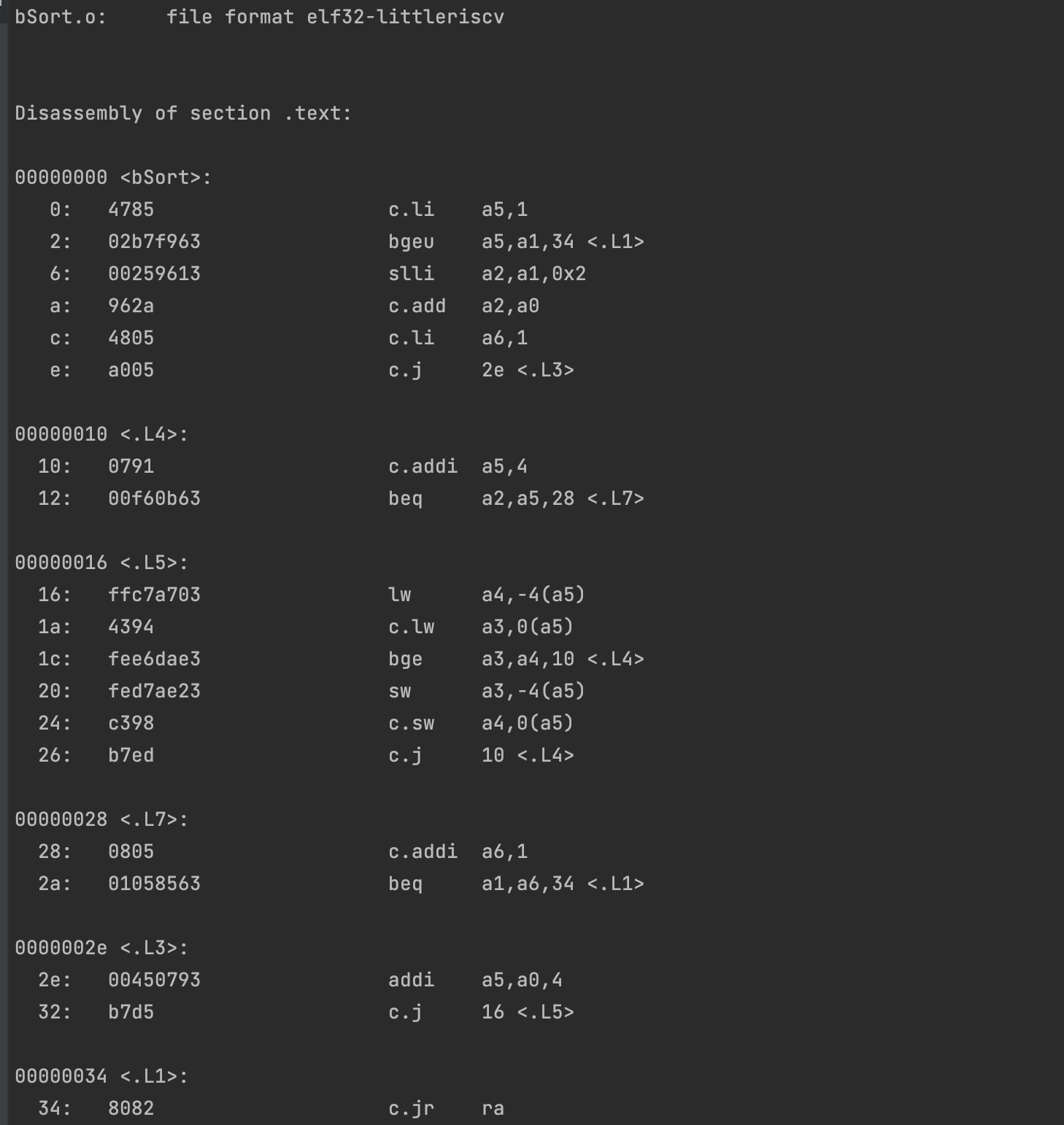
В каждой таблице только один глобальный (флаг “g”) символ типа «функция» (“F”) – “bSort” и “main” соответственно.

Проанализируем секции .text объектных файлов – секций кода, в которых содержатся коды инструкций:

Листинг 2.13. Секции .text







Компоновка

Компоновка осуществляется следующей командой:

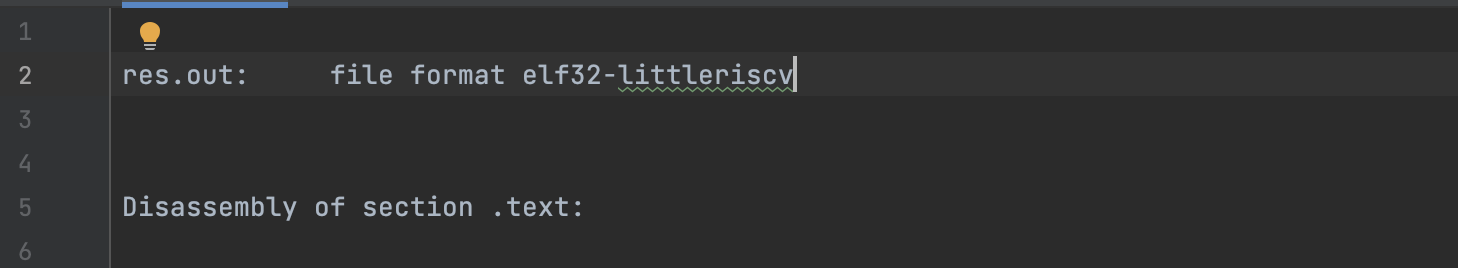
/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-gcc-8.3.0 -march=rv32imac -mabi=ilp32 main.o bSort.o -o res.out

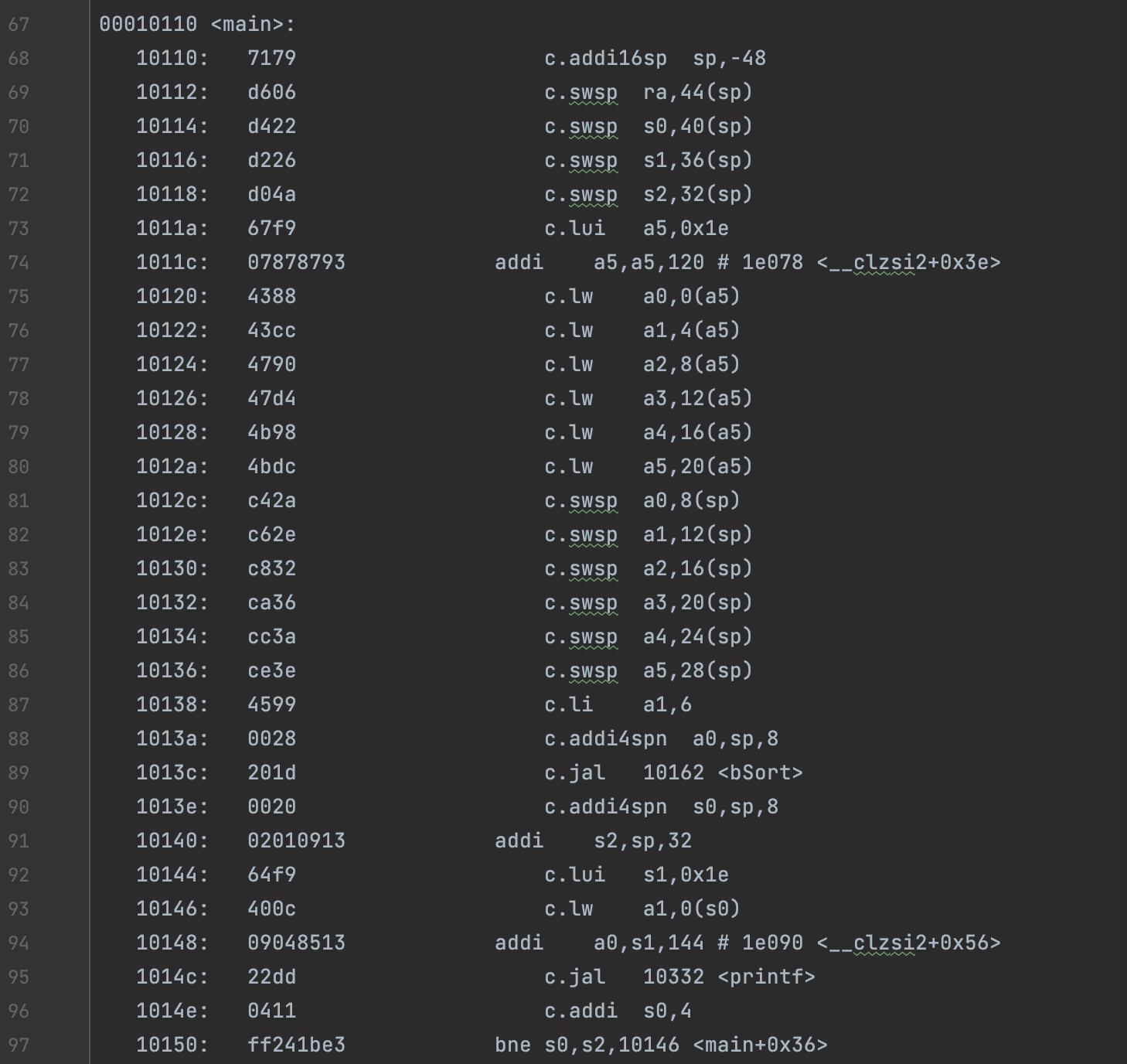
В результате получаем исполняемый файл main, скомпилированный под архитектуру risc-v. Произведём дизассемблирование полученного бинарного файла и обратим внимание на метки, соответствующие функциям.

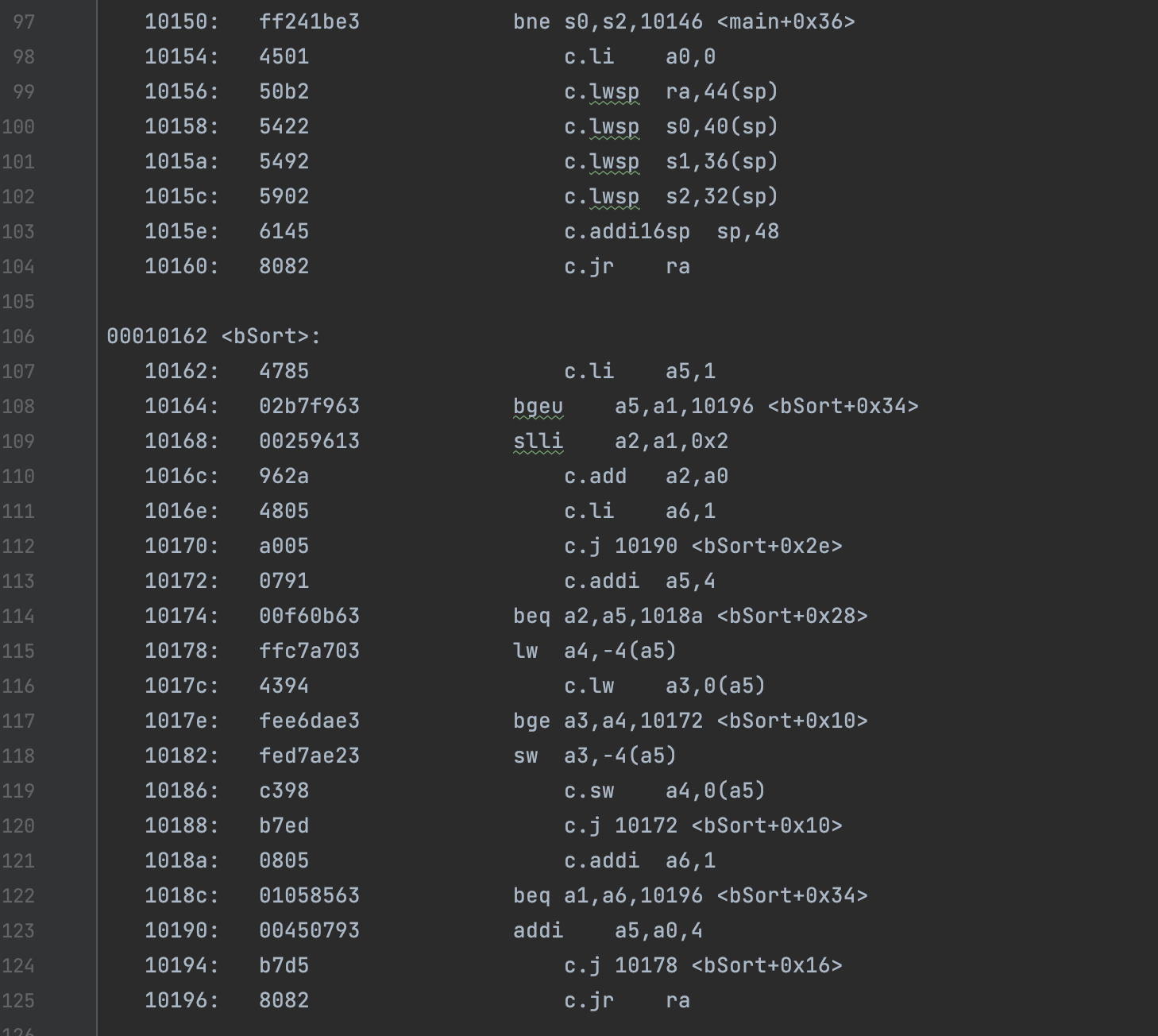
Команда, используемая для дизассемблирования:

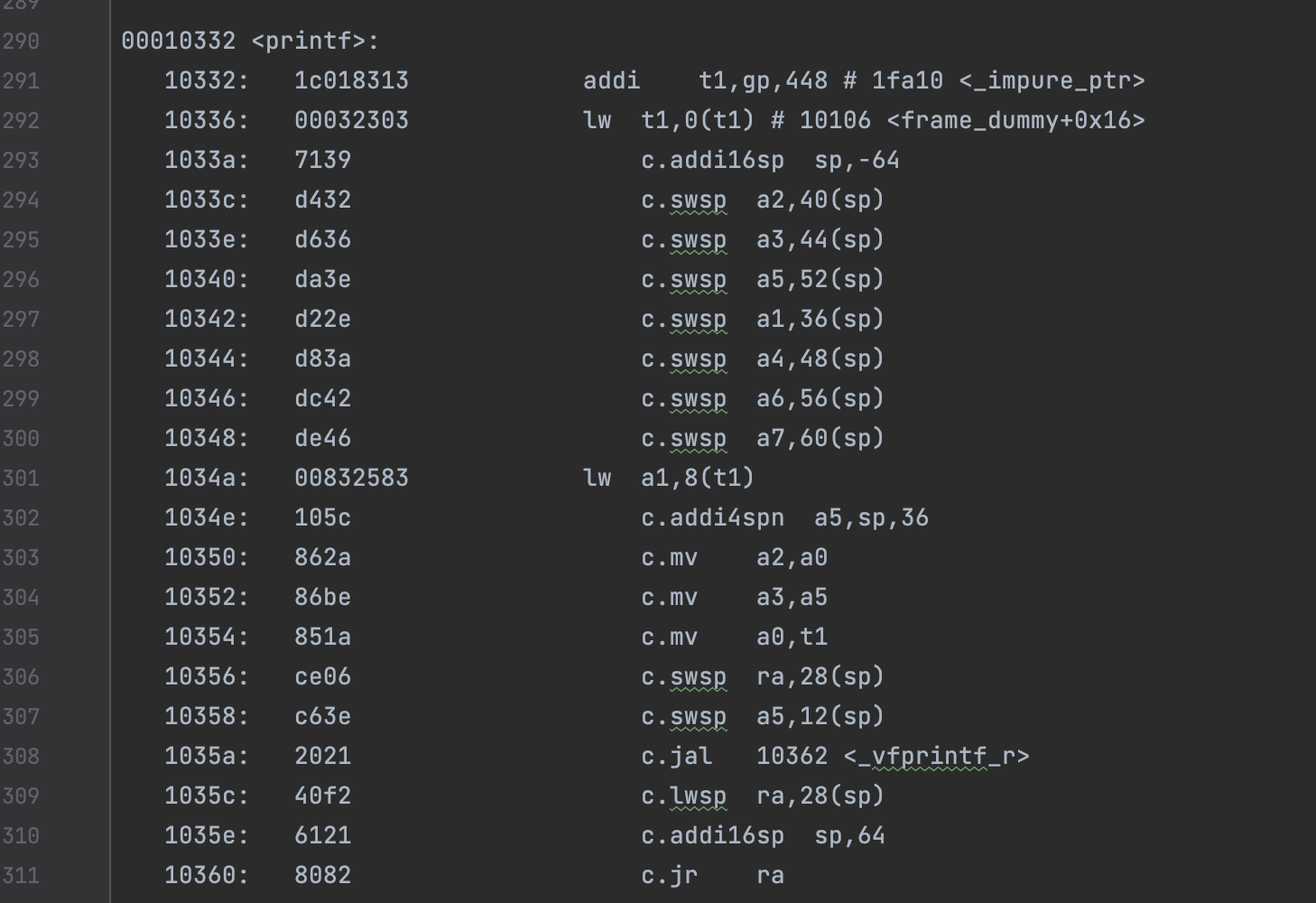
/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-objdump -j .text -d -M no-aliases res.out >res.ds

Листинг 2.14. файл res.ds









В дизассемблированном файле видим, что при компоновке была произведена «склейка» всего исполняемого кода из всех компонуемых объектных файлов, а также были просчитаны и расставлены все адреса.

Листинг 2.15. Информация о res.out

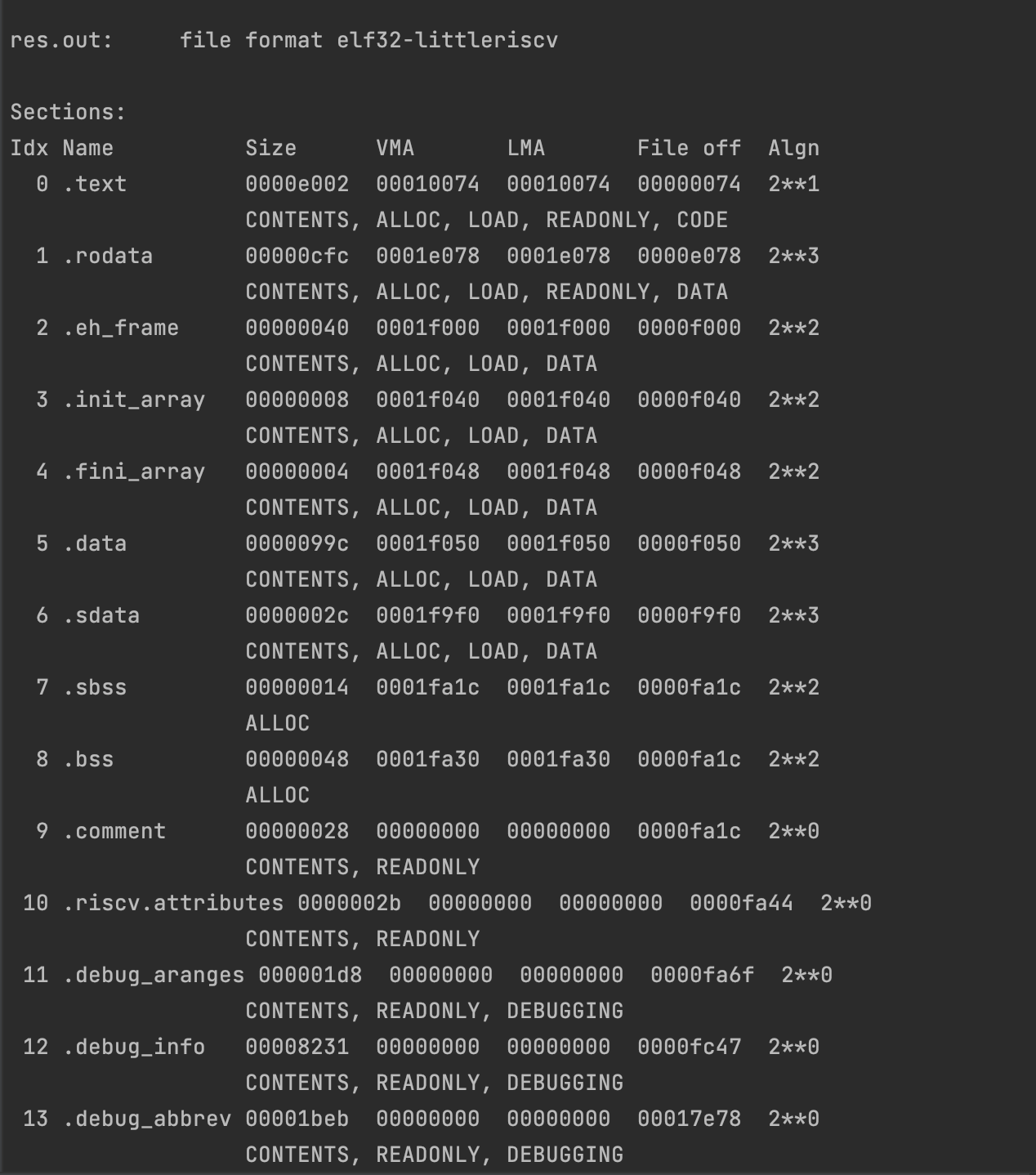
/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf -objdump –f res.out

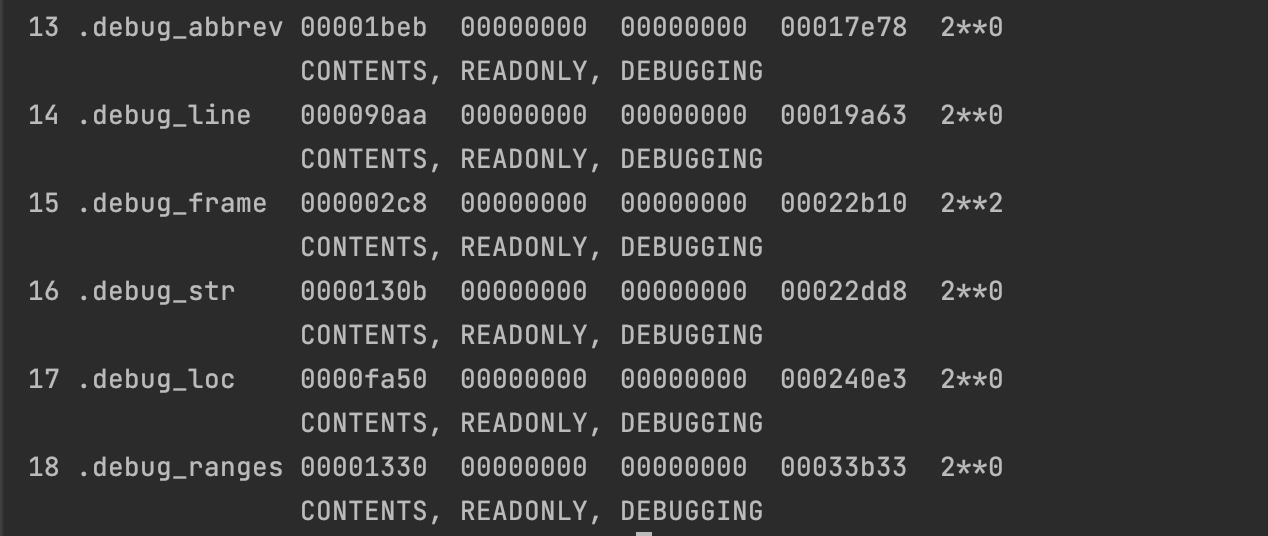


Флаг EXEC\_P указывает, что файл действительно является исполняемым, после загрузки его выполнение должно начаться с адреса 0x0001008a.

Листинг 2.16. Заголовки секций res.out

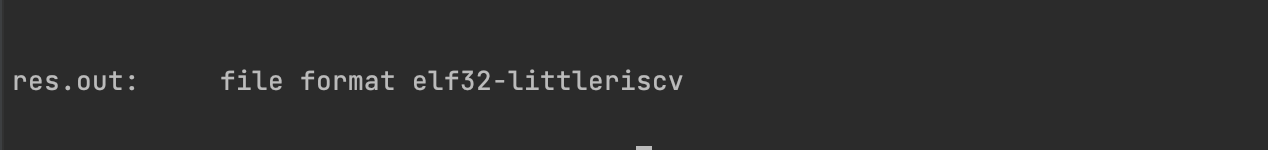
/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-objdump –h res.out



В исполняемом файле действительно производится слияние содержания секций обоих объектных файлов, а также значительное расширение списка секций новыми блоками.

Листинг 2.17. Таблица перемещений файла res.out

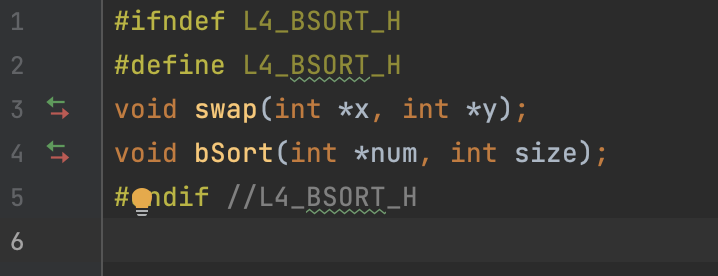
/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-objdump -r res.out

Таблица перемещений оказывается пуста, все необходимые релокации, оптимизации и замены инструкций были успешно проведены компоновщиком.

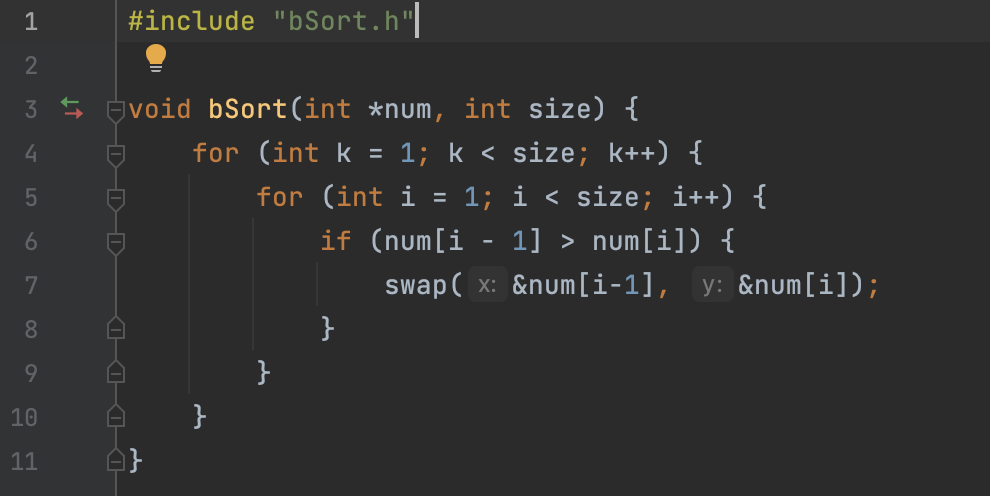
**3. Создание статической библиотеки и make-файлов**

Выделим из программы bSort.c функцию смены местами в отдельную программу swap.c. Объединим bSort.c и swap.c в статическую библиотеку bSortlib, тестовую программу main.c оставим без изменений.

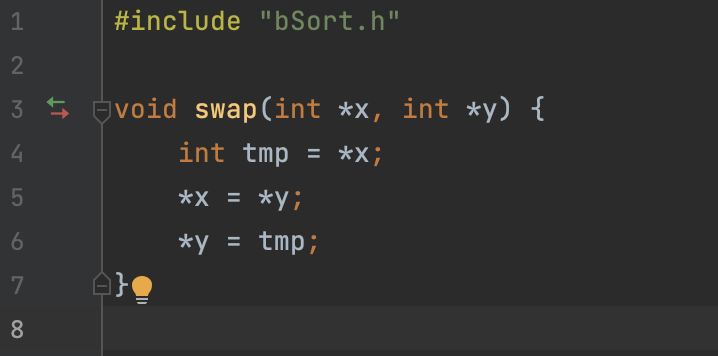
Листинг 3.1. Заголовочный файл bSort.h



Листинг 3.2. Основной файл bSort.c



Листинг 3.3. Функция swap.c



Для создания статической библиотеки получим объектные файлы всех используемых программ: bSort.o и swap.o.

/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/ riscv64-unknown-elf-gcc-8.3.0 -O1 -c bSort.c -o bSort.o

/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/ riscv64-unknown-elf-gcc-8.3.0 -O1 -c swap.c -o swap.o

Объединим получившиеся файлы в одну библиотеку следующей командой:

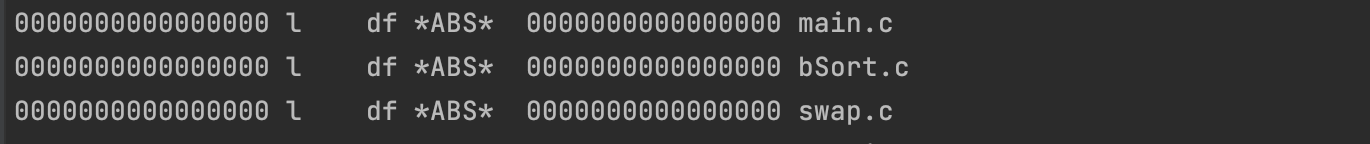
/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-ar -rsc bSortlib.a bSort.o swap.o

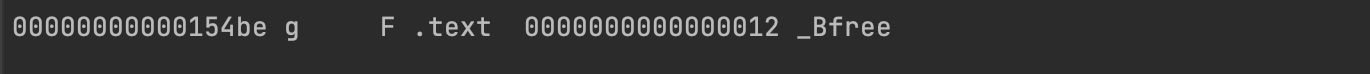
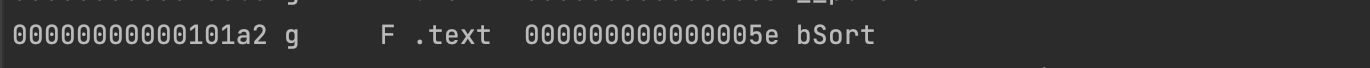
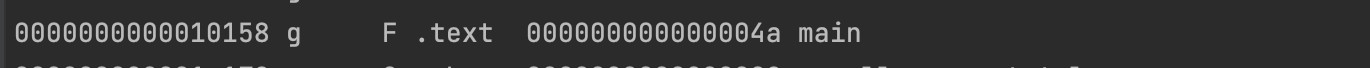
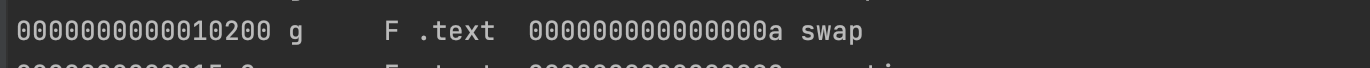
Используя получившуюся библиотеку, соберем исполняемый файл программы следующей командой:

/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/ riscv64-unknown-elf-gcc-8.3.0 -O1 --save-temps main.c bSortlib.a

Листинг 3.4. Таблица символов исполняемого файла (фрагмент)

/Users/mariakonasova/Downloads/riscv64-unknown-elf-gcc-8.3.0-2020.04.1-x86\_64-apple-darwin/bin/riscv64-unknown-elf-objdump -t a.out

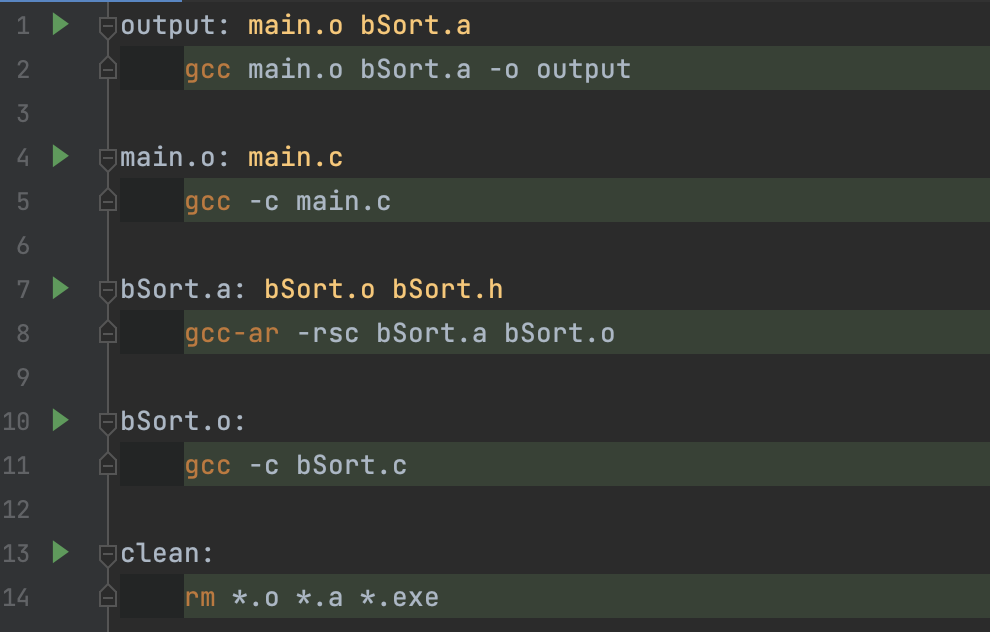




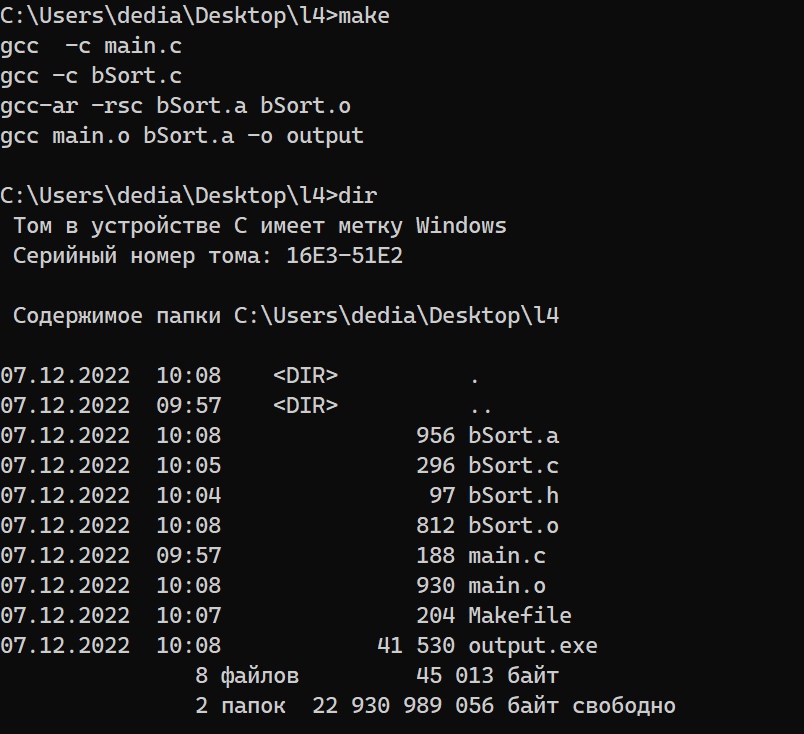
Можно заметить, что в состав программы вошло содержимое объектных файлов bSort.o и swap.o.

Процесс выполнения команд выше можно заменить make-файлами, которые произведут создание библиотеки и сборку программы.

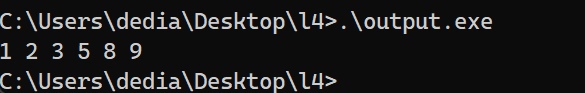
Листинг 3.5. Makefile



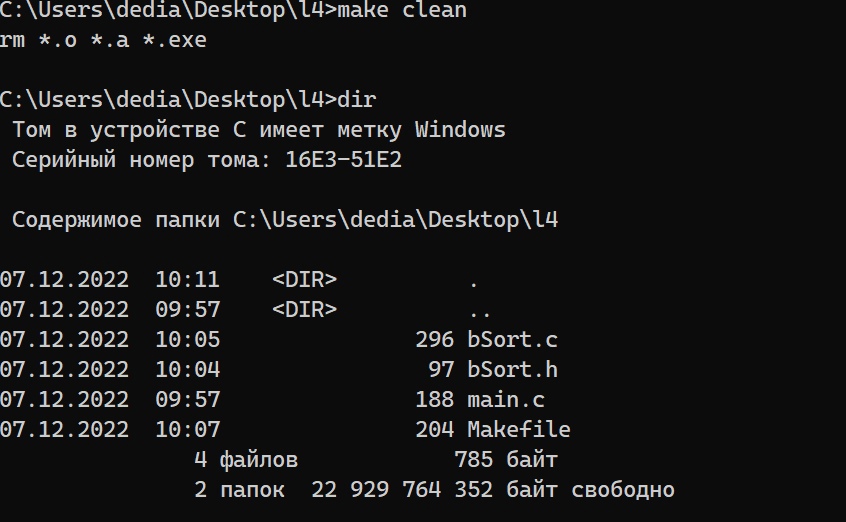
Листинг 3.6. Запуск и полученная директория



Листинг 3.7. Результат работы exeфайла



Листинг 3.8. Директория после очистки



**4. Вывод**

В ходе выполнения лабораторной работы были закреплены знания языка C, ассемблера RISC-V, получены навыки работы спрепроцессором, компилятором, ассемблером и компоновщиком пакета GCC и драйвером компилятора riscv64-unknown-elf-gcc. Были изучены особенности каждого этапа пошаговой сборки набора программ, а также инструменты, позволяющие выделить разработанные программы в статическую библиотеку и автоматизировать сборку этой библиотеки.

Была реализована поставленная задача – «сортировка пузырьком», а затем проведена проверка правильности перевода программы решения этой задачи в набор инструкций, выполняемых процессором.