|  |  |
| --- | --- |
|  | *Структура проекта* |

WINDOWS’ COMMAND LINE:

**Компиляция файлов и исходным кодом .java в файлы с байт-кодом .class:**

PS D:\\_GEEK Brains\JAVA\java\_core> javac

-encoding utf-8

-sourcepath ./seminar1/src/main/java

-d ./seminar1/out

./seminar1/src/main/java/pkgMain/Calc.java

, где

|  |  |
| --- | --- |
| -encoding utf-8 | Указание кодировки, в которой закодированы файлы с исходным кодом, для того, чтобы русский текст компилировался адекватно |
| -sourcepath ./seminar1/src/main/java | В случае, если проект включает много файлов и пакетов указывается папка, в которой расположены созданные нами файлы и пакеты (!!! Структура проекта !!!) |
| -d ./seminar1/out | Указание на папку, куда будут сохраняться скомпилированные .class файлы |
| ./seminar1/src/main/java/pkgMain/Calc.java | Путь к главному классу программы, содержащему точку входа **public static void main**. Путь указывается полностью относительно того места, откуда запускается **javac**. В нашем случае – из корневой папки проекта **java\_core.** |

**Генерация документации для проекта:**

PS D:\\_GEEK Brains\JAVA\java\_core> javadoc

-locale ru\_RU -encoding utf-8 -docencoding cp1251

-d ./seminar1/docs

-sourcepath ./seminar1/src/main/java/

-cp ./seminar1/out

-subpackages

pkgFunctions pkgMain

, где

|  |  |
| --- | --- |
| -locale ru\_RU  -encoding utf-8  -docencoding cp1251 | Указание кодировки, в которой закодированы файлы с исходным кодом (кодировка файлов .java - UTF-8), и кодировки, с которой будет сохраняться документация (кодировка windows - CP1251), для того, чтобы русский текст представлялся адекватно. |
| -d ./seminar1/docs | =destination - указание на папку, куда будут сохраняться файлы сгенерированной документации. |
| -sourcepath ./seminar1/src/main/java | В случае, если проект включает много файлов и пакетов указывается папка, в которой расположены созданные нами файлы и пакеты (!!! Структура проекта !!!) |
| -cp ./seminar1/out | = class path – указание на папку, где располагаются скомпилированные .class файлы. Необходимо дл того, чтобы JAVADOC понимал, какие классы скомпилированы, чтобы взять из них информацию для документации |
| -subpackages | Флаг, указывающий на необходимость рекурсивного обхода всех подпапок |
| ./seminar1/src/main/java/pkgMain/Calc.java | Путь к главному классу программы, содержащему точку входа **public static void main**. Путь указывается полностью относительно того места, откуда запускается **javac**. В нашем случае – из корневой папки проекта **java\_core.** |

ЗАДАЧА

Создать два Docker-образа. Один должен компилировать Java-проект обратно в папку на компьютере пользователя, а второй забирать скомпилированные классы и исполнять их.

Для решения задачи необходимо создать общую с хостовой системой папку. Это можно реализовать двумя путями:

ВАРИАНТ 1.

Создаём докер файл

#syntax=docker/dockerfile:1  
  
FROM bellsoft/liberica-openjdk-alpine:latest

COPY ./seminar1/src/main/java ./src

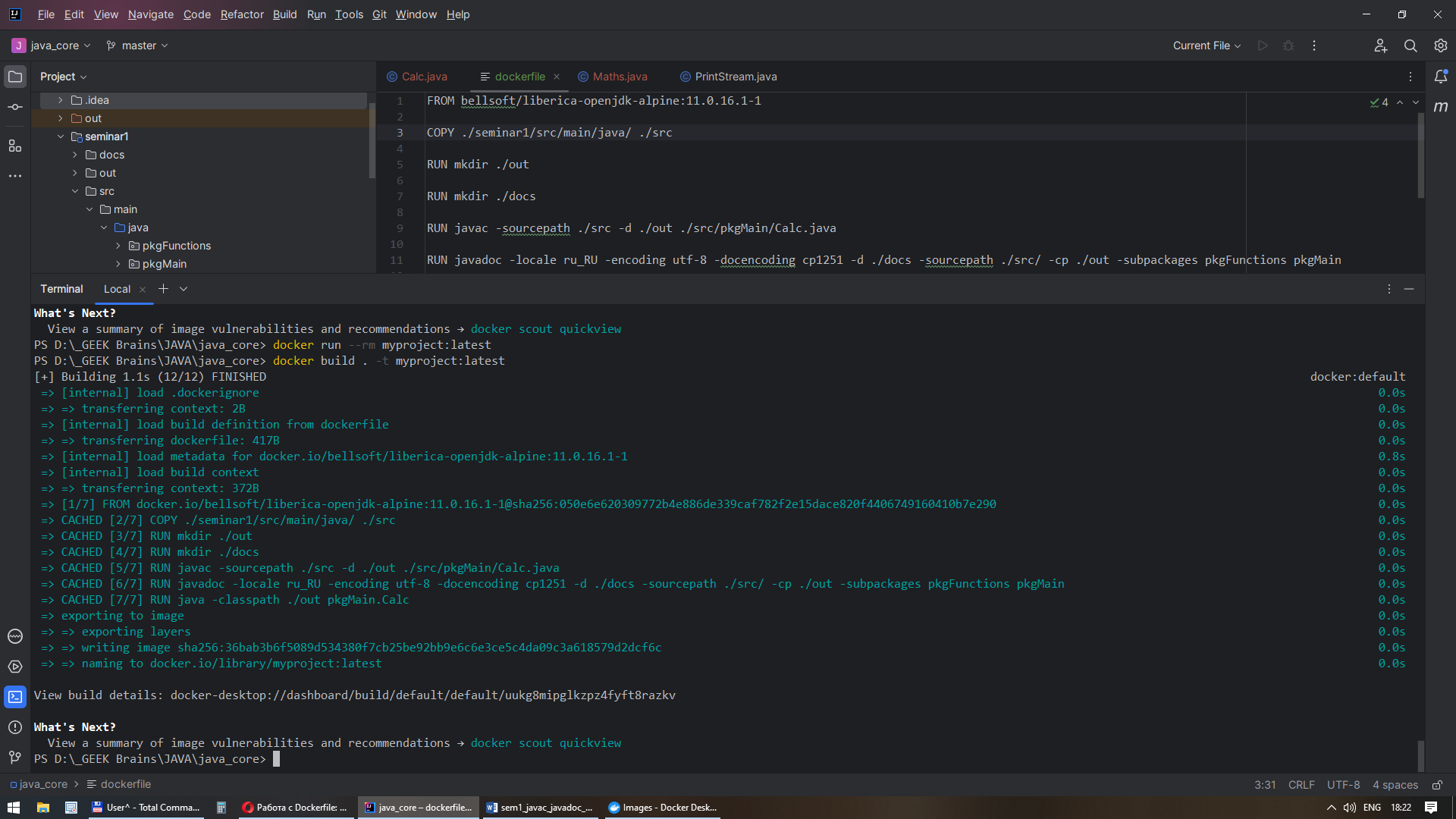
RUN mkdir ./out  
  
RUN mkdir ./docs  
  
RUN javac -sourcepath ./src -d ./out ./src/pkgMain/Calc.java  
  
CMD java -classpath ./out pkgMain.Calc ; javadoc -locale ru\_RU -encoding utf-8 -docencoding cp1251 -d ./docs -sourcepath ./src -cp ./out -subpackages pkgFunctions pkgMain

В последней строчке указываем две команды – выполнение программы JAVA и генерация документации по программе JAVADOC.

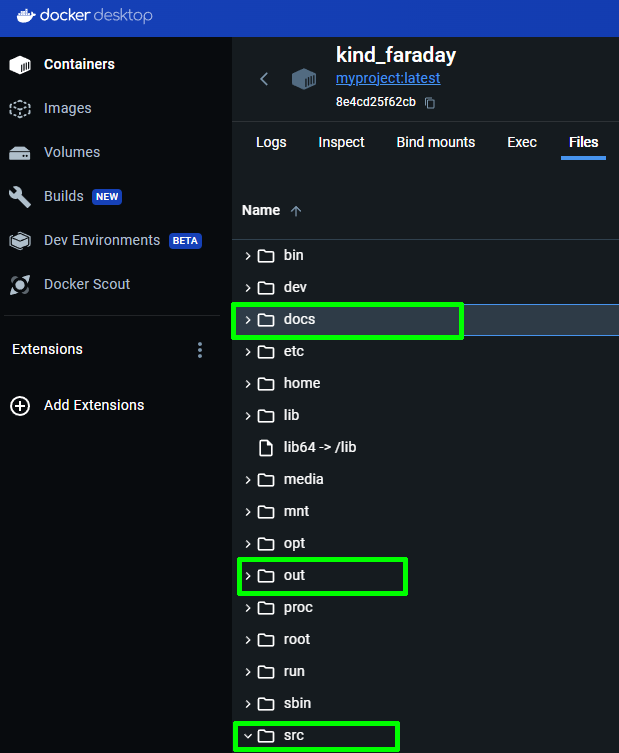
Собираем ОБРАЗ по докер файлу

Docker build . –t myproject:latest

Наблюдаем за ходом сборки образа



ДЛЯ СПРАВКИ: в результате вышеуказанных манипуляций папки /doc, /src и /out находятся в корневом каталоге контейнера.

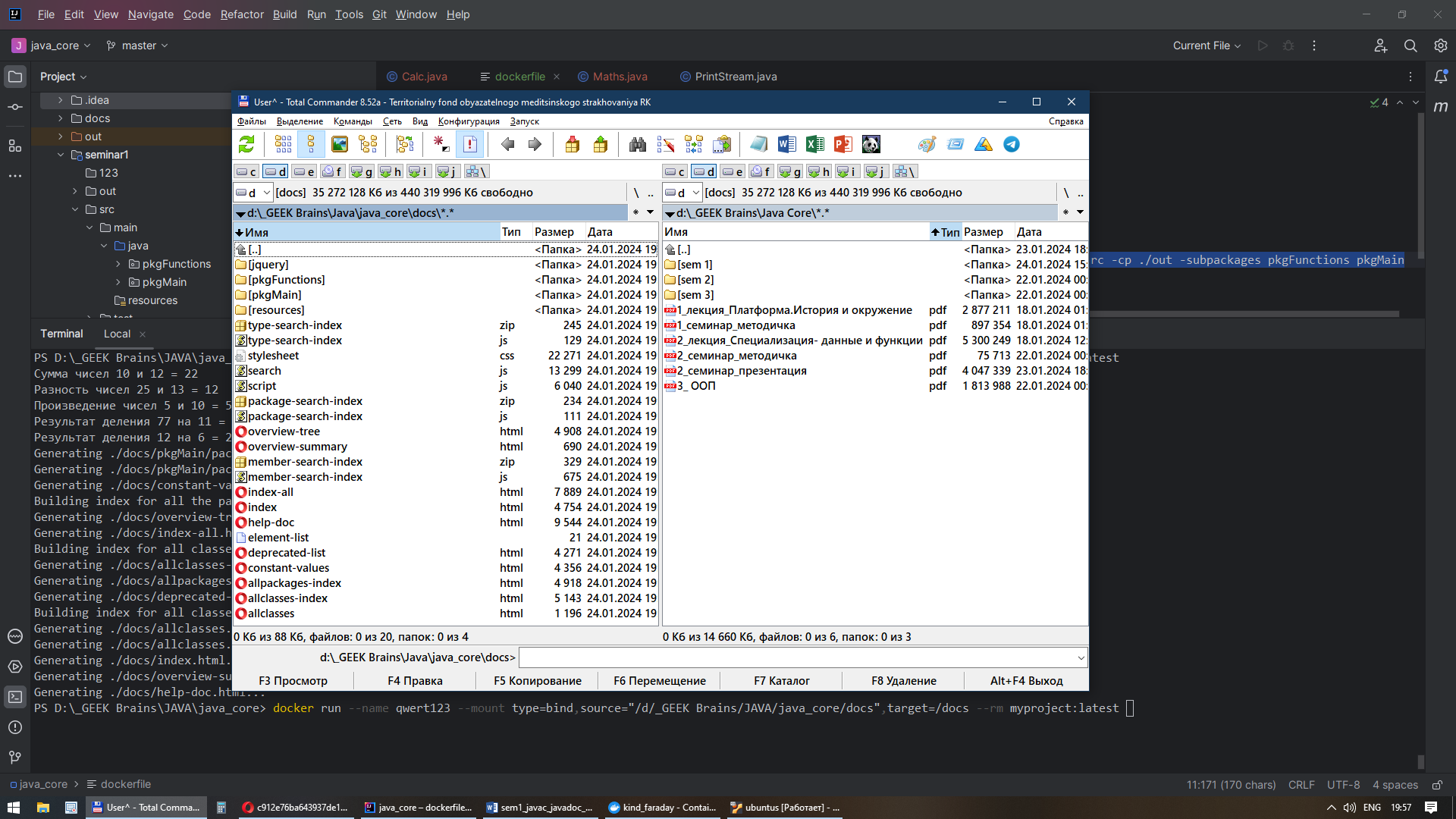
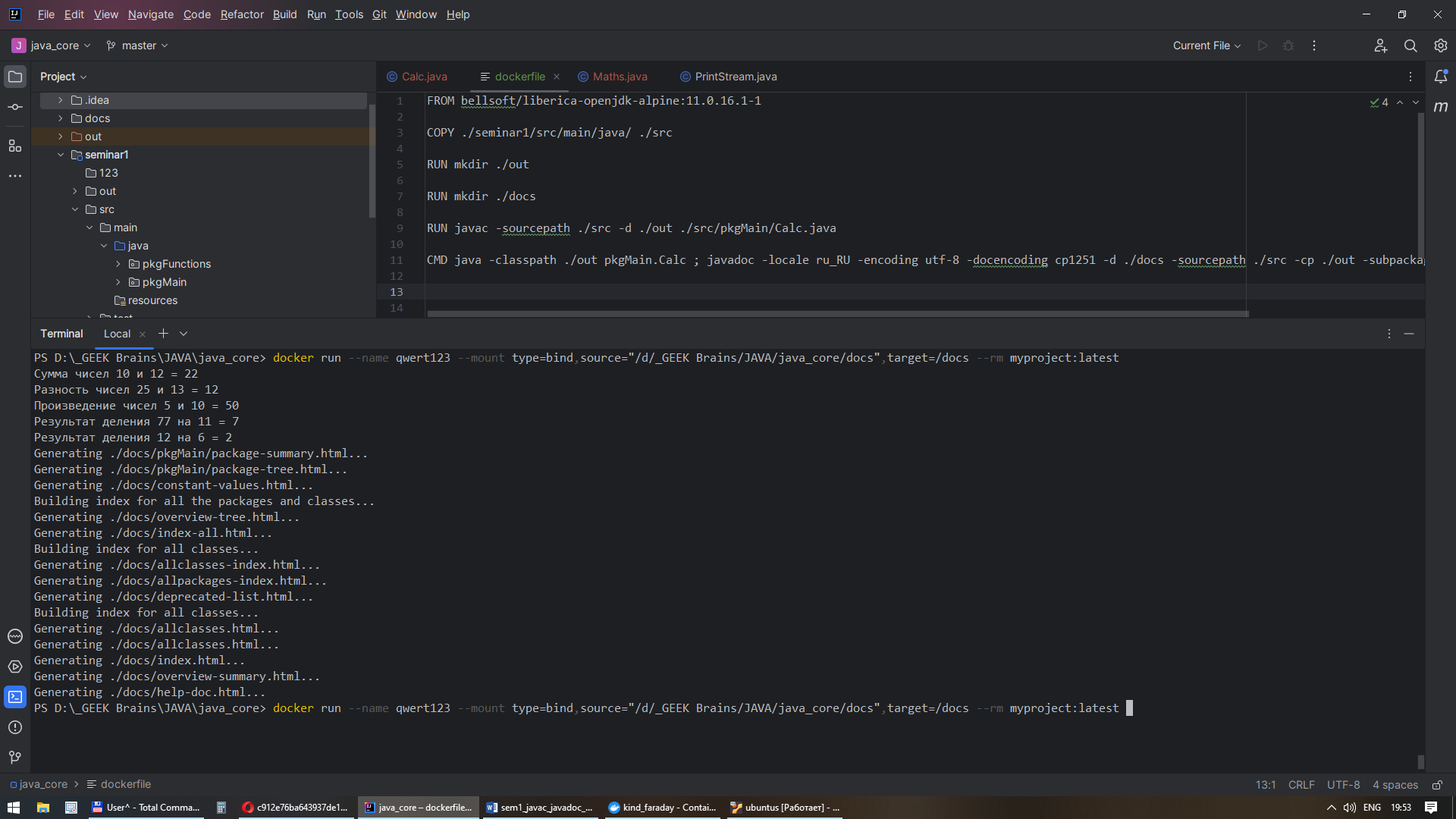


И далее – создаём связь между контейнером и хостовой системой при запуске контейнера через команду docker run с флагом –mount type=bind (используя BIND MOUNT – монтирование папки, а не виртуального диска). При этом в секции source указывается папка хостовой системы, а в секции target путь к папке в контейнере.

При желании таким же способом добавляем --mount для папки с компилированными файлами .class.

ВАЖНО: так как мы пользуемся DOCKER DESKTOP для WINDOWS, указание адреса папки хостовой системы необходимо делать в стиле windows или в стиле UNIX – как показано ниже.

docker run --name qwert123 --mount type=bind,source="/d/\_GEEK Brains/JAVA/java\_core/docs",target=/docs --rm myproject:latest



ВАРИАНТ 2

Используем docker-compose - в YAML файле создаём раздел VOLUMES и прописываем общую папку

Оба эти файла запускаются из корня папки проекта командами

docker compose -f docker-compose-class.yml up

docker compose -f docker-compose-exec.yml up

docker-compose-class.yml :

services:

app:

image: bellsoft/liberica-openjdk-alpine:11.0.16.1-1

command: javac -sourcepath /app/src -d /app/out /app/src/ru/gb/jcore/ sample/Main.java

volumes:

- ./seminar1/out:/app/out

- ./seminar1/src/main/java:/app/src

docker-compose-exec.yml

services:

app:

image: bellsoft/liberica-openjdk-alpine:11.0.16.1-1

command: java -classpath /app/out ru.gb.jcore.sample.Main

volumes:

- ./bin:/app/out

ПОЛЕЗНЫЕ ССЫЛКИ ПО ТЕМЕ

Как подключить каталог хоста к контейнеру Docker

<https://cpab.ru/kak-podkljuchit-katalog-hosta-k-kontejneru-docker-cloudsavvy-it/>

Управление данными в Docker

<https://doka.guide/tools/docker-data-management/>