VE281 — Data Structures and Algorithms

Programming Assignment 5

Instructor: Hongyi Xin
— UM-SJTU-JI (Summer 2022)

Notes

- Due Date: Lecture time on July.25, 27, 29
- Submission: None

1 Introduction

In this project, you are asked to implement an android takeaway platform. You will learn about front end and back end programming of an app. Don't worry too much! We have finished most of the parts and what you need to do is just to copy-paste your code from former projects. Once you setup the files correctly, the app is able to work. A bonus will be awarded to those who have optimized the user interface.

2 Android Studio Setup

2.1 Download

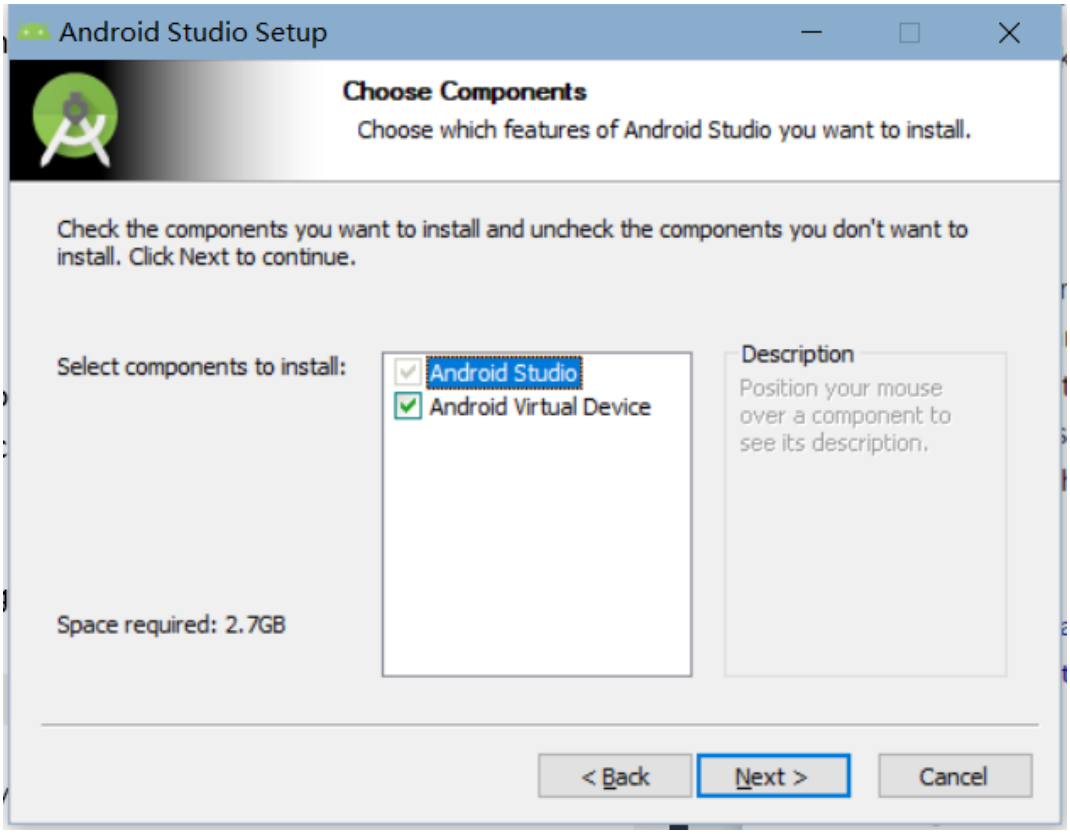
Please refer to link https://developer.android.com/studio#downloads to download the latest version of Android Studio Package for your OS.

Android Studio downloads

Platform	Android Studio package	Size	SHA-256 checksum
Windows (64-bit)	android-studio-2021.2.1.15-windows.exe Recommended	929 MiB	d99d2b24e232ac869d9c9e64cd19cd2572cfd4512e7df7941b812c4cd39d7b45
	android-studio-2021.2.1.15-windows.zip No .exe installer	940 MiB	a992449e546660fc5f8026f72b0f0e804f8d9c82a5e628d502ab60d1ab2b0f8c
Mac (64-bit)	android-studio-2021.2.1.15-mac.dmg	1017 MiB	fcaf413951119324ab22a0a4f2634bade90692991e9381cbf4a274e3cd60243a
Mac (64-bit, ARM)	android-studio-2021.2.1.15-mac_arm.dmg	1014 MiB	ce1fb8ba48c93e65fca450667786b4b98aa000ff274ca890e3de36efdabcc22b
Linux (64-bit)	android-studio-2021.2.1.15-linux.tar.gz	964 MiB	0018e0dfc0dd2921700516f7a2c443377c557788da7fb0a45243ecb4300745be
Chrome OS	android-studio-2021.2.1.15-cros.deb	817 MiB	1300f2e48734ad57b9598f1f620ab2c72436b7fa60ced2d96266dadaf3078489

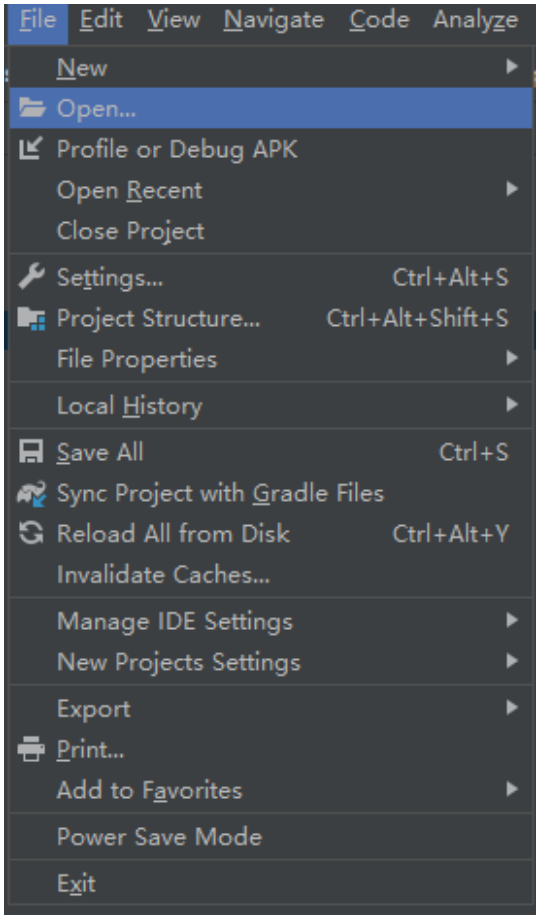
2.2 Install

During the installation process, make sure you select both **Android Studio** and **Android Virtual Device** for choosing components. For the remaining steps, just click **Next>**.



2.3 Open the Project

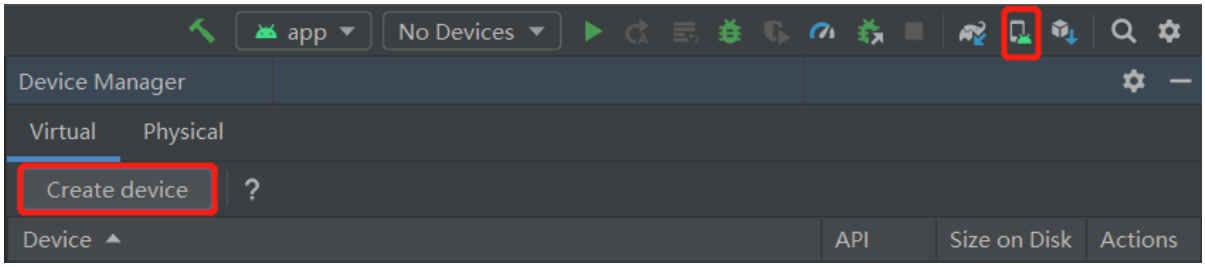
Please first unzip the handout zip file in a certain file folder. You can see a file folder named **Test**. Open the project in your **Android Studio** by clicking **File** $\longrightarrow$ **Open**, then choose **Test** in the file folder.



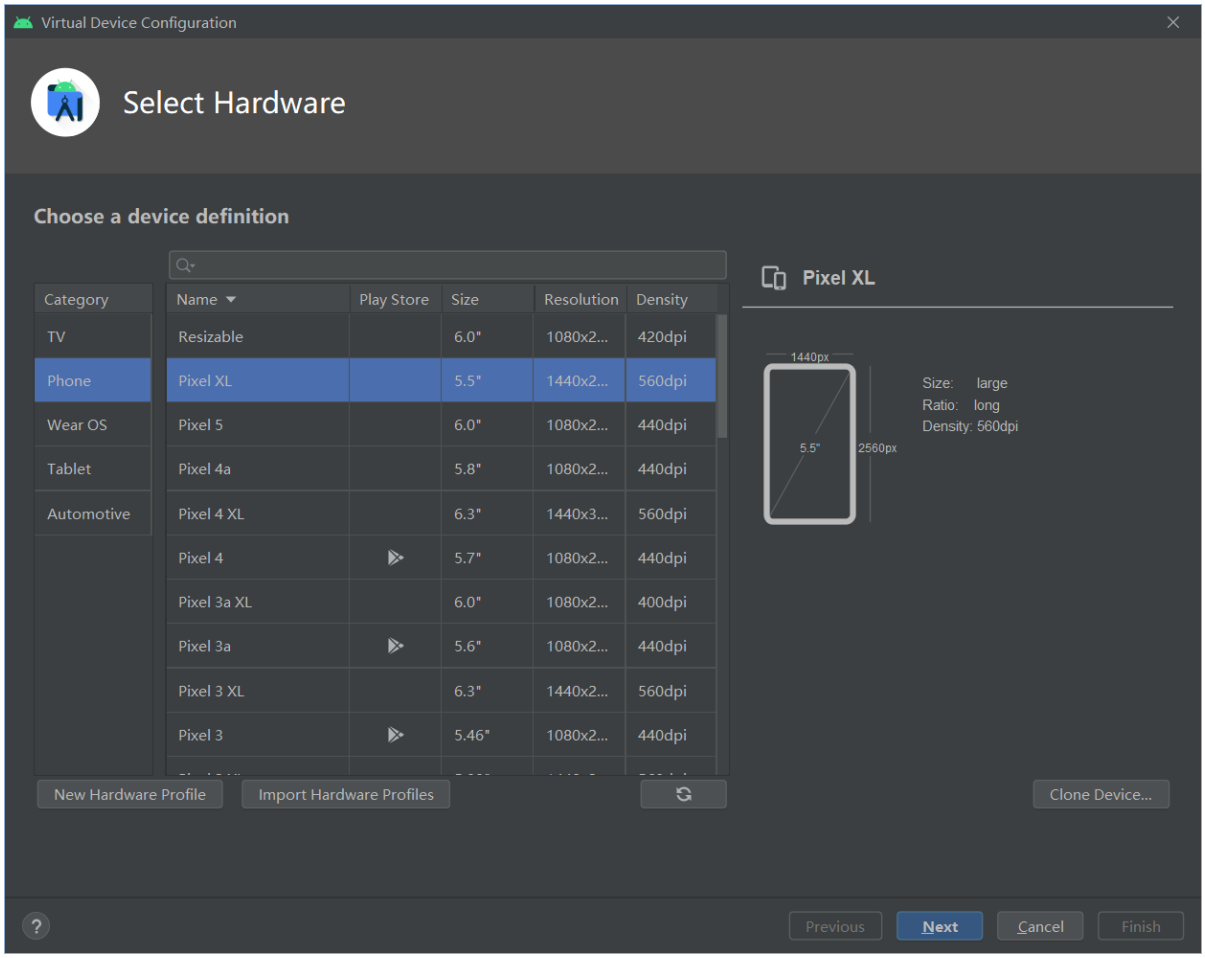
You should be able to see different coding files now, we will introduce them to you in detail later.

2.4 Device Manager

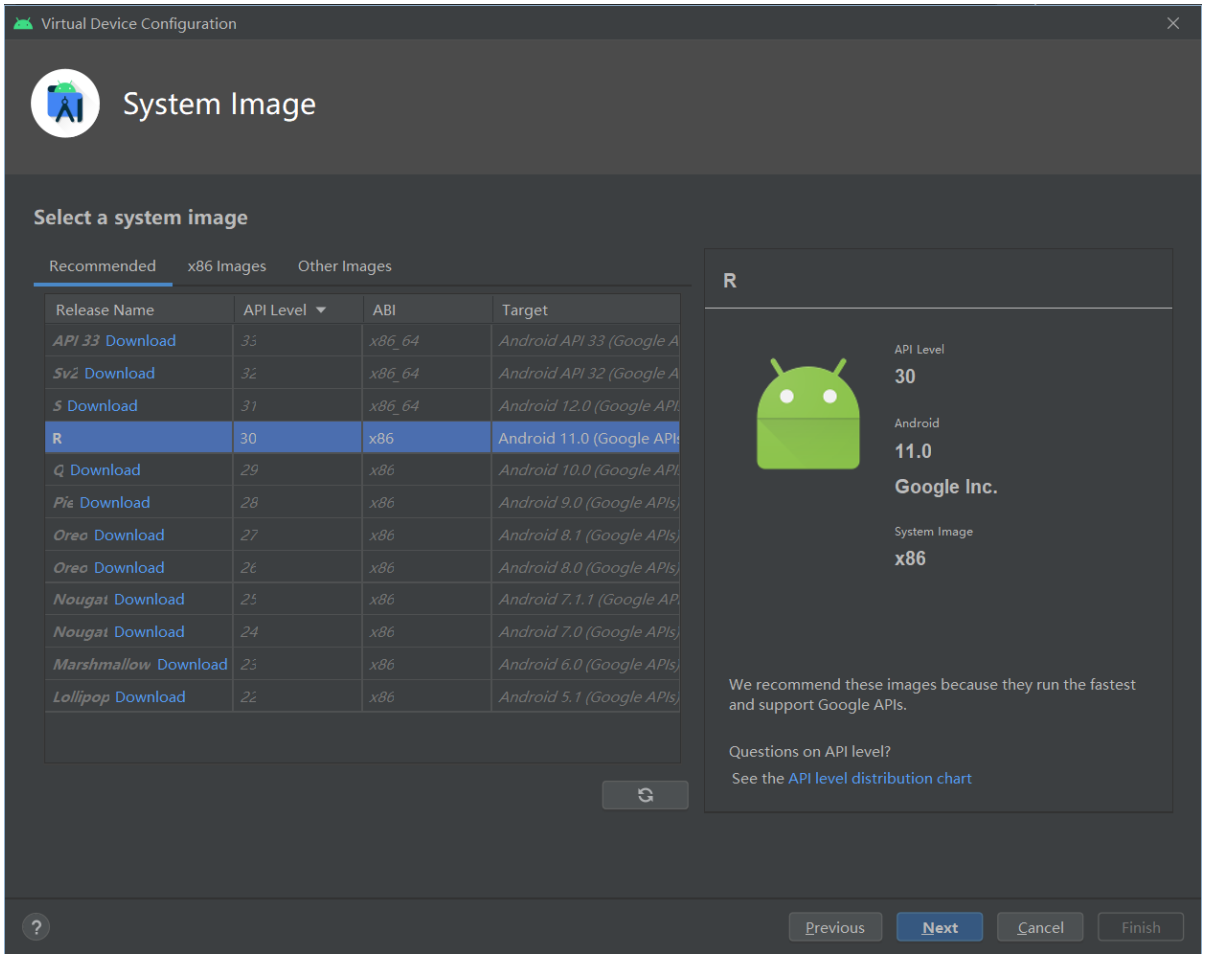
Here is where you add virtual android devices as simulator, or physical devices if you have one as carrier of your program. You are able to find **Device Manager** on the top right corner of the interface. Since a virtual implementation is enough for this project, we then click **Create device** under **Virtual** column.



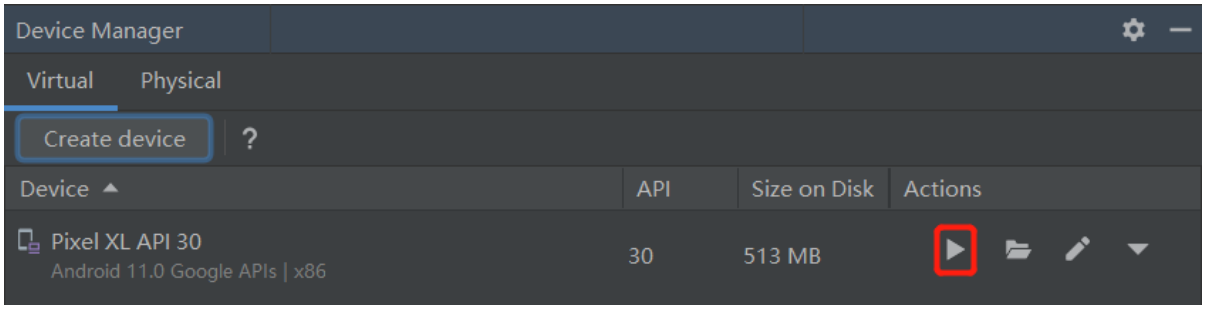
Here we recommend you to pick **Pixel XL** under **Phone** category.



For system image, please choose **R**, as shown in the figure. You may need to download for the first time. If any connection issue is encountered, please try using VPN.



Click **finish** to finalize virtual device selection. You should find Pixel XL ready on the right side. Click **Run** to connect to the emulator. Congratulations!!! You now have a new android phone!!!



3 Functions of App

3.1 Data

As a delivery APP, we have a map as the background. Each point on the map can be uniquely represented by its x-coordinate and y-coordinate. For simplicity, the coordinates are integers. The range of both x-coordinate and y-coordinate is [0,500]. There are several crossroads on our map, each of them is considered as a point. A point is a crossroad if and only if both of its x-coordinate and y-coordinate are multiples of 100.

There is an edge between the adjacent crossroads, with an unique integer to represent its length. To go from a

crossroad to another on the map, we must travel on the edges.
We also have shops in our delivery APP. It will initialize 25 takeaway shops randomly when you start the simulator. Each shop has the following several factors:

i) Location: each shop has its unique x-coordinate and y-coordinate to represent its location on the map.

ii) Average Cost: each shop has own average cost. The average cost of different shops may not be unique.

iii) Rating: each shop has its own rating to represent its reputation. The rating of different shops may not be unique.

iv) Name: each shop has its unique name. In our APP, the name of a certain shop with x-coordinate x_0 and y-coordinate y_0 is "testx_0, y_0".

3.2 Sorting

In this part, you're required to use the code you have in project 1. This part of function will be shown on the **LIST** page.
Originally, you have a list of the shops representing their factors in a random order. And there are two buttons, **ORDER BY RATING** and **ORDER BY PRICE**. Once the user press the button, the shops will be rearranged and represented in a different order. If the user press **ORDER BY RATING**, the shops should be represented in a descending order based on their rating. If the user press **ORDER BY PRICE**, the shops should be represented in a ascending order based on their average cost.

3.3 Search

In this part, you're required to use the code you have in project 2. This part of function will be shown on the **FIND** page.
There are a text box and a result box on this page. Users can type a specific shop name in the text box to search whether it is one of our shops. If the shop name perfectly matched with one of the shops we have, print its name, average cost and rating in the result box. Otherwise, print **None** in the result box.

3.4 Map

In this part, you're required to use the codes you have in project 3 and project 4. However, the codes should be further modified by yourself, the requirement will be introduced later. This part of function will be shown on the **MAP** page.
This part has two sub functions. Originally you will have a map, on which you can see each shop, crossroad and edge. Firstly, users can pick a certain point on the map with coordinates x_0 and y_0, and give a certain distance d. The APP will only show the shops whose x-coordinate is in the range [x_0 - d, x_0 + d] and y-coordinate is in the range [y_0 - d, y_0 + d]. Other shops will be hidden in the map. Users can press the button **RESET MAP** to reset the map and check all the shops.
Secondly, users can pick a certain point on the map and a certain shop. The APP will first find crossroads near the given point and the given shop correspondingly. Then it will highlight the shortest path between these two crossroads.

4 Files Preparation

4.1 C++ Files

C++ codes is the back end program of this APP. It determines how this APP works. You can find all C++ files needed for this project in the folder **/app/cpp**.
You may be familiar with some of them, as they are the handout files of your project 1-4. However, you may find other files in the folder new, you can refer to the comments given in the code if you are interested in them. They are basically used as connections between c++ and java.

4.2 User Interface Files

This part introduces several files that construct the user interface.
You can find five xml files in the folder **/app/res/layout**. These xml files determines the appearance of user interface (the overall layout, the color, etc.). You can refer to the following notes for the function of them.

i) **activity_main.xml**: determines the background layout of the whole APP, i.e., three buttons on the top and the main interface below.

ii) **fragment_shop_list.xml**: determines the layout of the **LIST** page. Corresponds to the function introduced in Section 3.2

iii) **fragment_shop.xml**: determines how the shops are represented in the **LIST** page. Corresponds to the function introduced in Section 3.2

iv) **fragment_find.xml**: determines the layout of the **FIND** page. Corresponds to the function introduced in Section 3.3

v) **fragment_map.xml**: determines the layout of the **MAP** page. Corresponds to the function introduced in Section 3.4

You can also find five java files in the folder **/app/java/com.example.myapplication/ui.main**. These java files interacts with the back end program and determines the functionality of the user interface, for example, what happens when you click the button. Please refer to the comments in the file for detail information.
Unlike the files mentioned in Section 4.1, you should understand the code in order to finish your task. You may want to learn **xml** coding and **android studio components** yourself. After having basic knowledge about them, you may find it easy to read the code.

5 Task

Your task for this project consists of two parts: finish the APP and optimize it.

5.1 Finish the APP

Your first tasks is to implement the function in **sort.hpp**, **hashtable.hpp**, **kdtree.hpp** and **shortestP2P.hpp**. All of them are in the folder **/app/cpp**. For the first two files, you can directly copy-paste your code from project 1 and 2 and finish them. But for **kdtree.hpp** and **shortestP2P.hpp**, please refer to the following two notes:

(1) For project 3 codes (in kdtree.hpp), you are required to implement an additional method **Range_Search** which is introduced in the lecture. We have already written the starter code of it, please finish it directly.

(2) For project 4 codes (in shortestP2P.hpp), it is not the same as your project 4 codes and the requirement of the function is different from it. Please refer to the explanation we give you in the file. Don't mix it up with project 4!

Since **Android Studio** is not a good place for debugging, please finish your code in your own IDE first.
After finishing the codes in these files, your APP should function well. Please set up **device manager** and **simulate** your code to check! You can refer to the demo video to see whether your APP functions well.

5.2 Optimize User Interface

Your second task is to optimize the user interface, to make it clearer and more pleasing! You are free to modify the files mentioned in Section 4.2, including java files and xml files. We don't have requirement for you in this part. Please use your imagination and design your unique APP!

6 Grading

Your APP will be evaluated during the lectures in the last week. It will be graded in two parts:

6.1 Functional Correctness

Once you setup the files right and your APP can function well with the required method, you will get full points for this part.

6.2 User Interface Optimization

We will organize students to evaluates each other's user interface. If you have done proper optimization of your user interface, you will get bonus for this part.

7 Acknowledgement

This project is jointly designed by Hongyi Xin, Shiyuan Shan and Chengyu Wu.