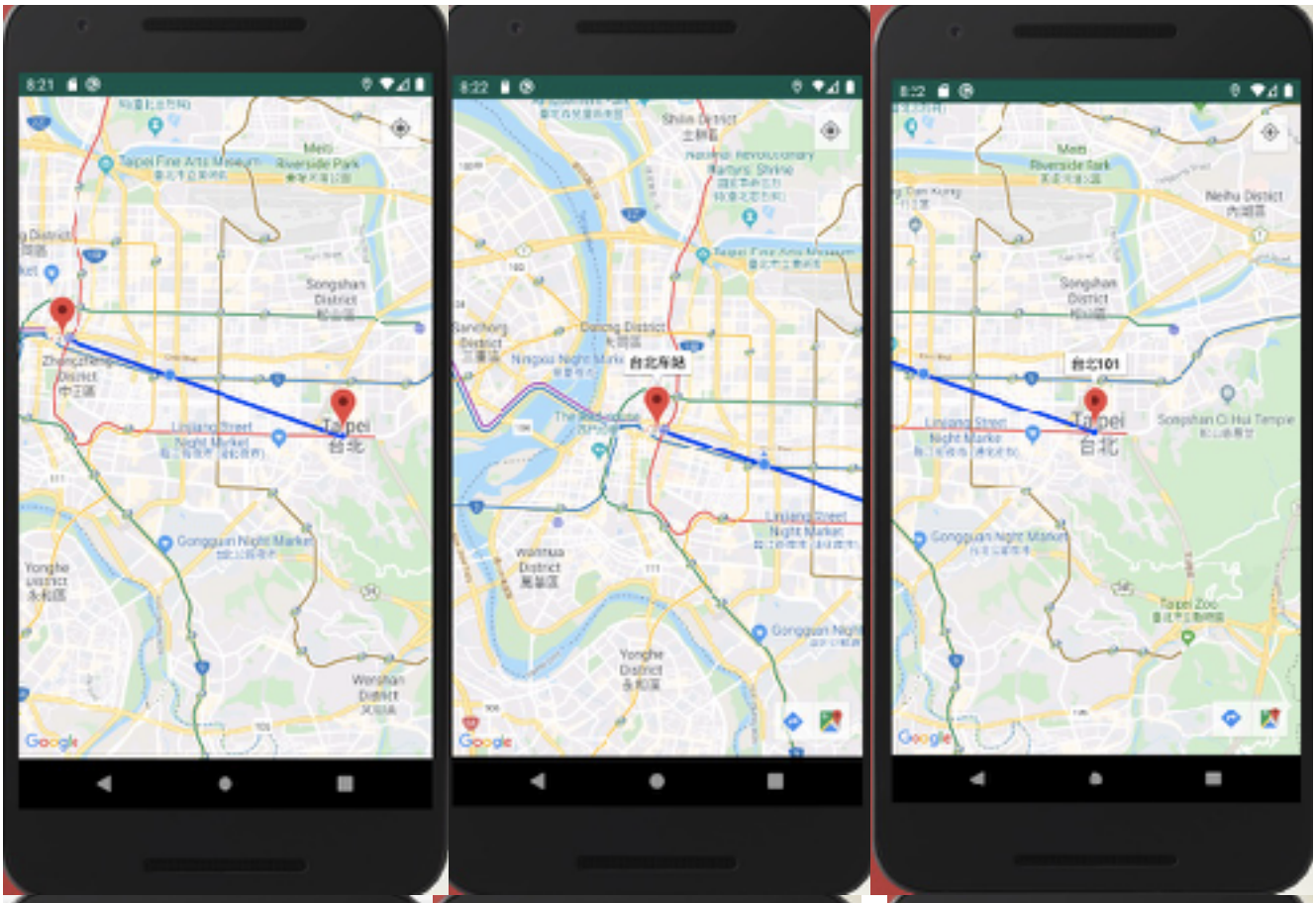

Taipei
台北
台北車站
台北101


lab11
書名:
價格:
查詢
新增
修改
刪除
書名:shzi 價格:580
書名:xiaomi 價格:777777777


lab11
書名:
價格:
查詢
新增
修改
刪除
書名:shzi 價格:500
書名:xiaomi 價格:777777777
書名:sanmao 價格:400


lab11
書名:
價格:
查詢
新增
修改
刪除
書名:shzi 價格:500
書名:xiaomi 價格:9999999999999
書名:sanmao 價格:400

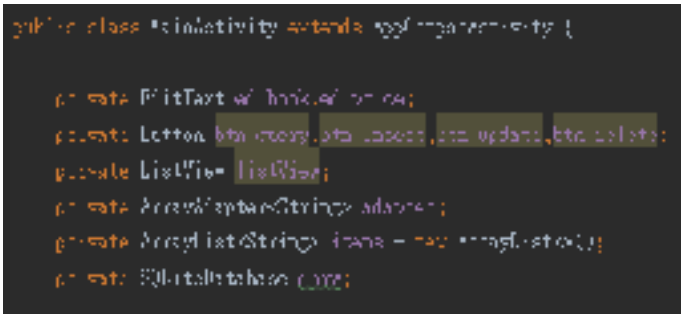
Lab10:

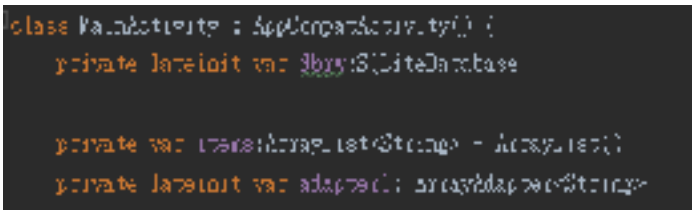
作業內容：

Google Maps 整合基本地圖、3D 建築、室內樓層平面圖、街景和衛星影像，以及自訂標記等功能。Google Maps Android API 可以根據「Google 地圖」的資料，將地圖新增至應用程式中。API 會自動處理對「Google 地圖」伺服器的存取、資料下載、地圖顯示，以及回應地圖手勢。也可以使用 API 呼叫，將標記、線段新增至基本地圖，以及變更使用者觀看的特定地圖區域。這些物件為地圖位置提供其他資訊，並允許使用者與地圖進行互動。

對比兩種語言，我在以下發現了有不同：

1：java需要事先宣告的東西比較的多，基本上有用到的東西都需要事先宣告，而kotlin語言會比較簡潔。





2：對比java語言來講，kotlin語言會比較簡潔。

3：java語言中常常使用@override 這個重寫方法，上網搜尋了其實不打override也是可以的，只是加上了很多的好處，可以當注釋使用，方便閱讀，kotlin語言中會使用override fun方式。

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_maps);
    if (ActivityCompat.checkSelfPermission( context: this,
            android.Manifest.permission.ACCESS_COARSE_LOCATION) !=
```

4：kotlin語言中沒有new關鍵字。

```
map.setMyLocationEnabled(true);
MarkerOptions m1=new MarkerOptions();
m1.position(new LatLng( v: 25.033611, v1: 121.565000));
m1.title("台北101");
m1.draggable(true);
map.addMarker(m1);
```

```
map.isMyLocationEnabled=true
val marker = MarkerOptions()
marker.position(LatLng(25.033611, 121.565000))
marker.title("台北101")
marker.draggable(true)
map.addMarker(marker)
```

Lab11: SQLite是一個由C語言撰寫的小型關聯式資料庫管理系統，與一般資料庫不同在於它不是一個主從關係結構的資料庫，而是被整合在應用程式中的嵌入式資料庫。Android 應用程式可以將資料儲存在手機上SQLite中，作為資料的快取之用，缺點是本地資料庫與伺服器的資料會有不同步的疑慮。

對比兩種語言，我在以下發現了有不同:

1: java語言需要宣告的東西比較多。

```
ed_book=findViewById(R.id.ed_book);
ed_price=findViewById(R.id.ed_price);
btn_query=findViewById(R.id.btn_query);
btn_insert=findViewById(R.id.btn_insert);
btn_update=findViewById(R.id.btn_update);
btn_delete=findViewById(R.id.btn_delete);
listView=findViewById(R.id.listView);
adapter=new ArrayAdapter<>( context: this,android.R.layout.simple_list_item_1,items);
listView.setAdapter(adapter);
dbrw=new MyDBHelper( context: this).getWritableDatabase();
```

2：kotlin語言比較簡潔。

```
    dbrw.execSQL(" DELETE FROM myTable WHERE book LIKE'"+ed_book.getTex
    Toast.makeText( context: MainActivity.this, text: "删除书名"+
            ed_book.getText().toString(),Toast.LENGTH_SHORT).show();
    ed_book.setText("");
    ed_price.setText("");
}catch (Exception e){
    Toast.makeText( context: MainActivity.this, text: "删除失败: "+
            e.toString(),Toast.LENGTH_SHORT).show();
```

```
/
brw.execSQL( sql: "DELETE FROM myTable WHERE book LIKE '${ed_book.tex
past.makeText( context: this, text: "刪除書名${ed_book.text}",Toast.LENGT
/
d_book.setText("")
d_price.setText("")
h (e:Exception){
past.makeText( context: this, text: "刪除失敗:$e",Toast.LENGTH_LONG).show
```

3：override的方式不一樣。

```
        });

    }
    @Override
    public void  onDestroy(){
        super.onDestroy();
        dbrw.close();
    }
```

```
    override fun onDestroy() {
        super.onDestroy()
        dbrw.close()
    }

}
```

心得：

通過這兩個作業，我發現lab10作業的內容很實用，明白了定位系統時怎麼而來的。java語言以及kotlin語言的練習更加讓我了解到了兩種語言的不同之處，以及他們之間的優缺點。

而lab11這個作業讓我學會了構建簡單的資料庫。在裡面嘗試新增.修改.查詢.刪除等功能。很適用於大型的查詢系統。