Criterion C: Development

The product is an Android application. The primary function of this application is to facilitate the collection of data, for the IB Geography IA. It saves the data that the student collected as text files into the internal storage of the device.

List of techniques used

- 1 dimensional arrays
- 2 dimensional arrays
- 3 dimensional arrays
- Data validation
- Error handling
- Passing parameters
- ActionListeners
- Reading a text file
- Writing to a text file

As previously mentioned, most of the code in Android Studio is automatically generated; this is a screenshot of the code of the MainActivity.java. By the way, in Android studio, there is no such thing as classes, instead they are called activities. The first half of the class was auto-generated; those lines of code are simply to create the Navigation Drawer (the technical name for a slide-in menu in Android Studio) and also to make it look animated and not just static (pressing buttons without it looking pretty).

```
public class MainActivity extends AppCompatActivity
        implements NavigationView.OnNavigationItemSelectedListener {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);


        DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
        ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(
                this, drawer, toolbar, R.string.navigation_drawer_open, R.string.navigation_drawer_close);
        drawer.setDrawerListener(toggle);
        toggle.syncState();

        NavigationView navigationView = (NavigationView) findViewById(R.id.nav_view);
        navigationView.setNavigationItemSelectedListener(this);
    }

    @Override
    public void onBackPressed() {
        DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
        if (drawer.isDrawerOpen(GravityCompat.START)) {
            drawer.closeDrawer(GravityCompat.START);
        } else {
            super.onBackPressed();
```

```
public boolean onNavigationItemSelected(MenuItem item) {
    // Handle navigation view item clicks here.
    int id = item.getItemId();

    FragmentManager fragmentManager = getFragmentManager();

    if (id == R.id.nav_first_objective) {
        fragmentManager.beginTransaction().replace(R.id.content_frame, new FirstFragment()).commit();
    } else if (id == R.id.nav_second_objective) {
        fragmentManager.beginTransaction().replace(R.id.content_frame, new SecondFragment()).commit();
    } else if (id == R.id.nav_third_objective) {
        fragmentManager.beginTransaction().replace(R.id.content_frame, new ThirdFragment()).commit();
    }


    DrawerLayout drawer = (DrawerLayout) findViewById(R.id.drawer_layout);
    drawer.closeDrawer(GravityCompat.START);
    return true;
}
```

The Boolean method OnNavigationItemSelected(MenuItem item) is in a way like a parser for the menu. It gets an input saying that one out of the four buttons in the slide in menu has been pressed and this method that I've created is just to tell which class to "commit" (is the keyword) so it creates a new instance of that class, this time it's not called an activity but a fragment, because the menu is fragmented into different parts. This works because in the activities, they each have a specific id which is randomly generated, but It's the same as the input sent when a

button is pressed, with that in mind, a link can be made between the two; displaying the layout of the respective fragment.

Bibliography

https://www.youtube.com/watch?v=ju837bQOBfg