# Do not remove any of the comments. These are marked by #

# HW 8

# (1). Please upload R code and report to Moodle

# with filename: HW8\_IS457\_YourClassID.

## Important: Make sure there is no identifying information on your printout, including name, username etc.

## Only include your class ID on there.

### **ClassID: 39**

### For this assignment, you will extract useful information from HTML and use Google Earth for data visualization.

### The LatLon.rda file containing the country geographic coordinate is uploaded to Moodle.

### Look at detail instructions for the assignment in hw8\_Intro.pdf.

### Part I. Create the data frame from HTML file (20 pts)

### Q1. Load LatLon.rda,

### install and load XML and RCurl libraries.

### Your code here

load("LatLon.rda")

library(XML)

library(RCurl)

### Q2. Download the html file from the url below, and parse it to html\_text. (2 pts)

### Open the website and read it before coding.

### We will be working on the data from one of the tables: United Nations 2016-2017 global population data

url = "https://en.m.wikipedia.org/wiki/List\_of\_countries\_by\_population\_(United\_Nations)"

html\_text = readLines(url)

### Your code here

### Q3. Read the tables in html\_text with the readHTMLTable() function, set the 1st row as header.

### Now you should have a list object of 2 data frames, of which one is the table of population data.

### Coerce the table of population data to a data frame named "population". (2 pts)

### Your code here

help("readHTMLTable")

Tables = readHTMLTable(html\_text, header = TRUE, as.data.frame = TRUE)

class(Tables)

Tables[[2]]

population = Tables[[2]]

population

### Q4. Let's simplify the data frame.

### Remove all other columns except country name, 2016 population, and 2017 population.

### Rename the 3 columns "Country", "Population\_2016" and "Population\_2017". (2 pts)

### Your code here

population = population[,c(2, 5, 6)]

population

names(population)[1] = "Country"

names(population)[2] = "Population\_2016"

names(population)[3] = "Population\_2017"

names(population)

### Q5. Recall our regular expression lessons.

### In the population data frame, some country names have annotations at the end.

### Using regular expressions, remove all the annotations in country names. (e.g. change "China[a]" to "China") (4 pts)

### (Hint: Combine apply family functions and string split methods), then convert the country names to uppercase. (1 pt)

### Show the first 5 rows of your new population data frame. (1 pt)

### Your code here

population

population$Country = gsub("\\[.\*]", "", population$Country)

population$Country = toupper(population$Country)

population[1:5,]

### Q6. Now merge LatLon with population by country to create a data frame named AllData. (2 pts)

### It should have 6 columns: Country, Code, Latitude, Longitude, Population\_2016 and Population\_2017.

### Your code here

LatLon

AllData = merge(LatLon, population, by.x = "Country", by.y = "Country")

AllData

### Q7. Finally, convert the population data to numeric values, and

### calculate the 2016-2017 growth rate percentage of population by country,

### and add the growth rate to AllData as a new column named "Growth". (3 pts)

### (Hint: growth rate percentage = (population in 2017 / population in 2016 - 1)\*100)

### Show the last 5 rows of your AllData data frame. (1 pt)

### Your code here

AllData$Population\_2016 = gsub(",", "", AllData$Population\_2016)

AllData$Population\_2017 = gsub(",", "", AllData$Population\_2017)

AllData$Population\_2016 = as.numeric(AllData$Population\_2016)

AllData$Population\_2017 = as.numeric(AllData$Population\_2017)

AllData$Growth = (AllData$Population\_2017 - AllData$Population\_2016)/AllData$Population\_2016 \* 100

tail(AllData, 5)

### Part II. Create a KML document for google earth visualization

### First take a look at the file on moodle: HW8\_Intro.pdf

### It shows the structure of the KML file which we will create next.

### Q8. Let's start with creating the base of the KML document.

### Create a base document named doc1. (1 pt)

### Then create nodes "kml" and "document". (2 pts)

### (Hint: Check arguments "doc" and "parent" to make the nodes connected)

### Your code here

doc1 = newXMLDoc()

root = newXMLNode("kml", doc = doc1)

child1 = newXMLNode("Document", parent = root)

### Q9. According to the KML tree in HW8\_Intro.pdf, you can add many placemark nodes with parent "Document".

### The addPlacemark() function below can be used to add one placemark to your file in each call.

### Explain what each line does. (4 pts)

### Your answer here

addPlacemark = function(lat,lon,country,code,pop16,pop17,growth,parent){ # Define the function with needed input arguments

pm = newXMLNode("Placemark",attrs=c(id=code),parent = parent) # Create the placemark node as the child node of "Document"

newXMLNode("name",country, parent = pm) # Create "placemark"'s child node "name" containing the country names

newXMLNode("description",paste(country,"\n population\_2016: ",pop16,"\n population\_2017: ",pop17,"\n growth: ",growth,sep =""),parent = pm) # Create "placemark"'s child node "description" which contains the detailed information of population in 2016 and 2017 as well as growth rate.

newXMLNode("Point",newXMLNode("coordinates",paste(c(lon,lat,0),collapse=",")),parent = pm) # Create "placemark"'s child node "point" which contains the coordinate information of each country

}

### Q10. Now let's create the KML file.

### Take doc1 as your base, then use addPlacemark() to create a KML file. (5 pts)

### Save it as "Part2.kml" using the saveXML() function. (1 pt)

### (Hint: First find the root of your base, then add placemark nodes to it in a for loop)

### Open the KML document in Google Earth. (You will need to install Google Earth.)

### If you are doing correctly, it should have pushpins for all the countries.

### Your code here

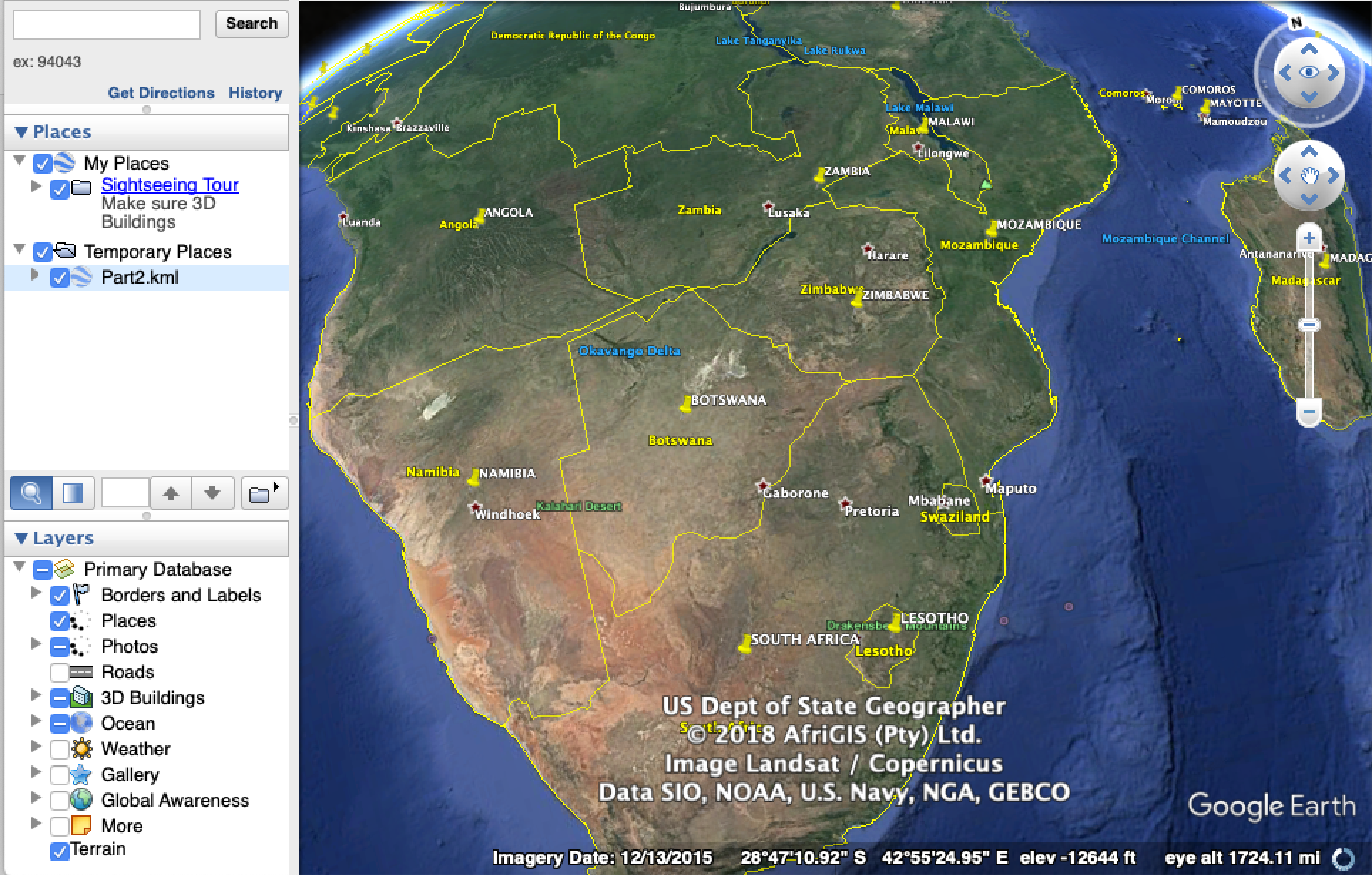
for (i in 1:nrow(AllData)){

addPlacemark(lat = AllData$Latitude[i], lon = AllData$Longitude[i], country = AllData$Country[i], code = AllData$Code[i], pop16 = AllData$Population\_2016[i], pop17 = AllData$Population\_2017[i], growth = AllData$Growth[i], parent = child1)

}

doc1

saveXML(doc1, file = "Part2.kml")



### Part III. Add Style to your KML file (16 pts)

### Now you are going to make the visualization a bit fancier.

### Instead of pushpins, we want different labels for countries with different population sizes in 2017,

### and we will use different colors to represent different levels of population growth rate.

### Code is given to you below to create style elements.

### Here, you just need to figure out what it all does.

### Q11. The following code is an example of how to create cut points for different categories of population in 2016.

pop16Cut = as.numeric(cut(AllData$Population\_2016, breaks=5))

### (But this example contains too many 1's, which is not suitable for visualization)

### So find suitable cut points for 2017 population and growth rate,

### and create your categories named pop17Cut and growCut. (3 pts)

### Hint: take a look at their distribution first. You may want to perform some simple transformations

### before finding the cuts.

### Explain the transformation you chose and why; also explain how you chose your cuts. (3 pt)

### Your code here

help("quantile")

quantile(AllData$Population\_2017, probs = seq(0, 1, 0.2))

pop17Cut = as.numeric(cut(AllData$Population\_2017, breaks = c(0, 284091.8, 3862275, 9844211.4, 31796090.2, 1409517397)))

quantile(AllData$Growth, probs = seq(0, 1, 0.2))

growCut = as.numeric(cut(AllData$Growth, breaks = c(-1.9574225, 0.2341591, 0.8342999, 1.3174967, 2.2173115, 4.7799182)))

### Your answer here

# The cuts chosen for population 2017 and growth rate were based on their quantiled distribution, both of which were

# split up into 5 ranges by 6 break points. Therefore, the classes of pop17 and growth would be able to be evenly

# generated as 5.

### Q12. We modify the addPlacemark() function in Q8, so it can add both placemark and style information.

### It has 3 new arguments: pop17cut, growcut, and style.

### Explain what the new line of code does. (2 pts)

addPlacemark = function(lat,lon,country,code,pop16,pop17,growth,parent,pop17cut,growcut,style=TRUE){

pm = newXMLNode("Placemark",newXMLNode("name",country),attrs=c(id=code),parent=parent)

newXMLNode("description",paste(country,"\n population\_2016: ",pop16,"\n population\_2017: ",pop17,"\n growth: ",growth,sep =""),parent=pm)

newXMLNode("Point",newXMLNode("coordinates",paste(c(lon,lat,0),collapse=",")),parent=pm)

if(style){newXMLNode("styleUrl",paste("#YOR",growcut,"-",pop17cut,sep=''),parent=pm)}

}

### Your answer here

# the new line in the code gives the function the ability to add style information based on new input arguments "pop17cut"

# and "growcut". It creates a child node "styleUrl" for "Placemark", containing the style url which is

# able to link the labels to their corresponding countries.

### Q11. Here is another function addStyle(), by which we can add style information to KML file.

### Figure out what the arguments "scales" and "colors" should be, and

### create two objects scale\_label and color\_label that you can input into this function. (5 pts)

### (Hint: For growth rate from low to high, you want to use this order of color: blue-green-yellow-orange-red)

### (Hint2: make a bigger symbol for country with larger population)

addStyle = function(parent,scales,colors){

for(j in 1:5){

for(k in 1:5){

st = newXMLNode("Style",attrs=c("id"=paste("YOR",j,"-",k,sep="")),parent=parent)

newXMLNode("IconStyle",newXMLNode("Icon",paste("color\_label/label\_",colors[j],".png",sep="")),newXMLNode("scale",scales[k]),parent=st)

}

}

}

### Your code here

scale\_lable = c(1, 2, 4 ,6, 10)

color\_lable = c("blue", "green", "yellow", "orange", "red")

### Q12. Let's build a tree properly, that contains both country and style information. (6 pts)

### You can complete this by following steps:

### 1) Create a base KML document named "doc2", similar to what you did in Q8

### 2) Add style information by addStyle() in nested for loops

### 3) Add placemarks by addPlacemark()

### Your code here

doc2 = newXMLDoc()

root2 = newXMLNode("kml", doc = doc2)

child2 = newXMLNode("Document", parent = root2)

addStyle(parent = child2, scales = scale\_lable, colors = color\_lable)

for (i in 1:nrow(AllData)){

addPlacemark(lat = AllData$Latitude[i], lon = AllData$Longitude[i], country = AllData$Country[i], code = AllData$Code[i], pop16 = AllData$Population\_2016[i], pop17 = AllData$Population\_2017[i], growth = AllData$Growth[i], parent = child2, pop17cut = pop17Cut[i], growcut = growCut[i], style = TRUE)

}

doc2

### Q13. Output your KML document, call it "Part3.kml". (1 pt)

### Open it in Google Earth to verify that it works.

### Explain your findings about the world population from the results you get. (2 pts)

### Your code here

saveXML(doc2, file = "Part3.kml")

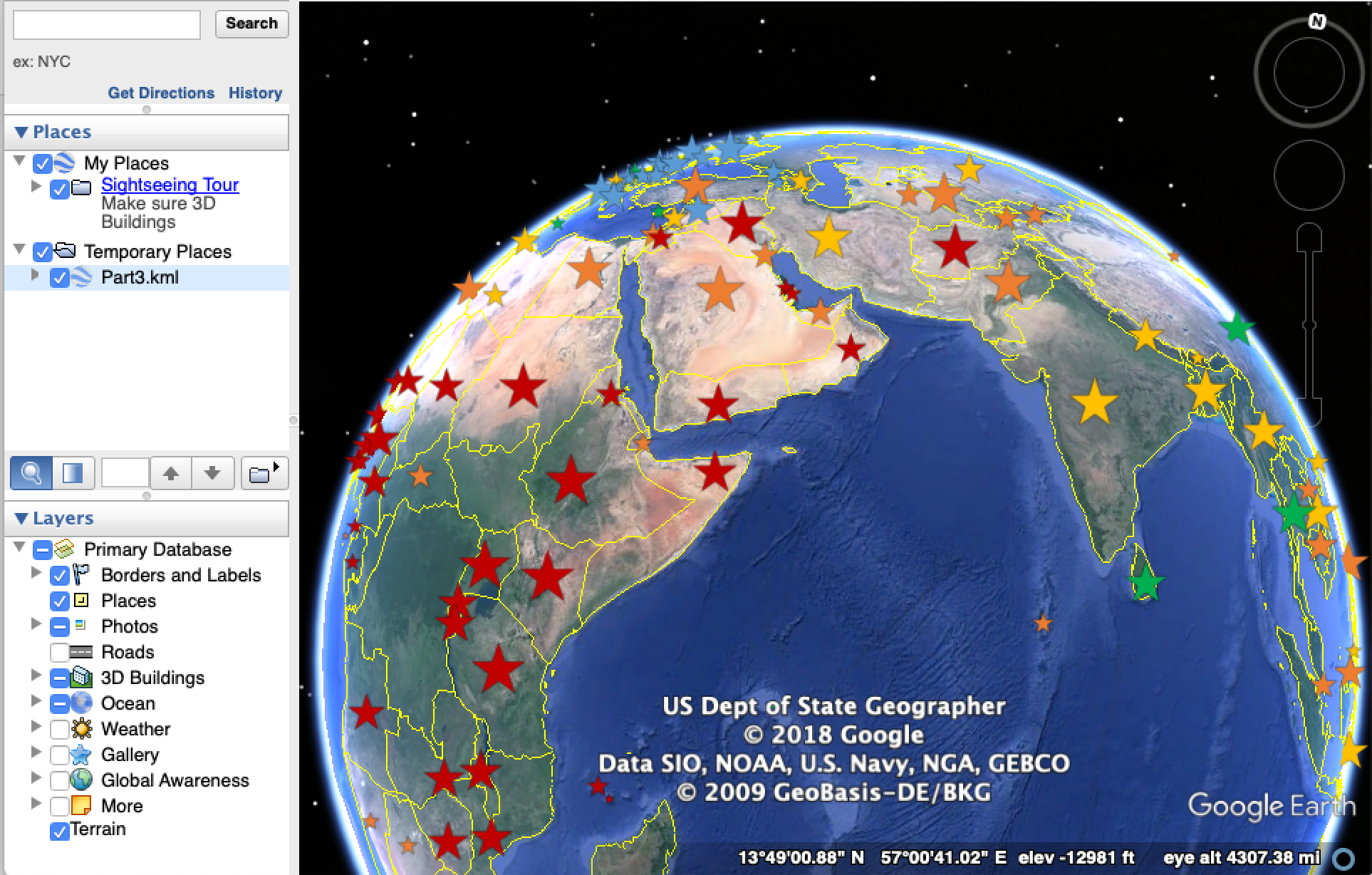
### Your answer here

# Generally speaking, countries with large land size usually have large population. In terms of population growth

# rate, regions with developped countries such as Western Europe, North America and East Asia have very low population growth rate, even some countries with

# negative growth rate. In opposite, regions with lots of developping countries such as Africa, South America and South Asia helds relatively high

# population growth rate.



### For this assignment, you only need to submit your R code and "Part3.kml", the PDF report is not required.