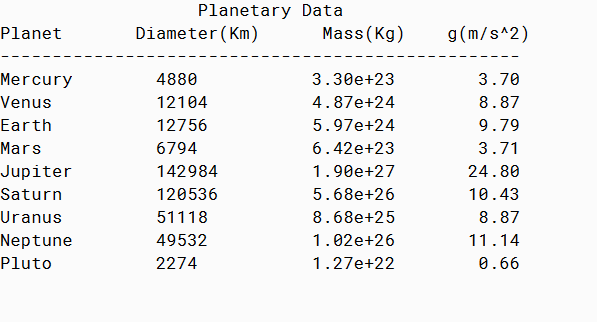
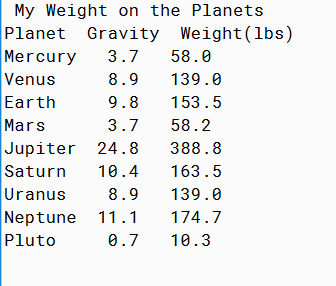
Gravity



Weight



PMR: This program was relatively easy to build. The hardest part was coding the iterations which took a while to get used to. On the weight one it took some work to get the correct format from grams to pounds.

**Gravity**

/\*\*

\* Calculates surface gravity using static methods given planet stats.

\*

\* @author Anika Jallipalli

\* @version 11/17/2019

\*/

import java.io.IOException;

import java.io.PrintWriter;

import java.io.File;

public class GravityV1

{

public static double[] resultantGravity( double []kg, double[]km)

{ double gravitationNum = 6.67E-11; //given gravitational constant

double []resultantGravities= new double[9]; //initialize array

for(int iteration = 0; iteration < resultantGravities.length;iteration++)

{

resultantGravities[iteration] = ((gravitationNum \* kg[iteration]) / (Math.pow(km[iteration]/2,2))); //given formula from lecture notes

}

return resultantGravities; //requires return statement to execute

}

public static void outputResults(String [] planet, double [] diameter, double [] kg, double[]gForce)

{

System.out.printf("%32s\n","Planetary Data");

System.out.printf("%-12s %-17s %7s %11s\n","Planet","Diameter(Km)","Mass(Kg)","g(m/s^2)");

System.out.println("--------------------------------------------------");

for(int iterate = 0; iterate < 9; iterate++)

{

System.out.printf("%-14s %-14.0f %5.2e %11.2f\n",planet[iterate],diameter[iterate],kg[iterate],gForce[iterate]);

}

}

public static void tableData(double[] gForce)throws IOException

{

PrintWriter outFile = new PrintWriter(new File("gravity1.txt"));

double []resultGravity = gForce;

for(int index = 0; index < resultGravity.length;index++)

{

outFile.println(resultGravity[index]);

}

outFile.close(); //close file writing to end loop

}

public static void main(String[]args) throws IOException

{

String [] planet = {"Mercury","Venus","Earth","Mars","Jupiter","Saturn","Uranus","Neptune","Pluto"};

//data from given table

double [] kg = {3.30E23,4.869E24,5.972E24,6.4219E23,1.900E27,5.68E26,8.683E25,1.0247E26,1.27E22};

double [] km = {4880000,12103600,12756300,6794000,142984000,120536000,51118000,49532000,2274000};

double [] diameter = {4880,12103.6,12756.3,6794,142984,120536,51118,49532,2274 };

double [] gForce = resultantGravity(kg, km);

outputResults(planet, diameter, kg, gForce);

tableData(gForce);

}

}

**Weight**

/\*\*

\* Calculates weight using static methods given planet stats.

\*

\* @author Anika Jallipalli

\* @version 11/17/2019

\*/

import java.util.Scanner;

import java.io.File;

import java.io.IOException;

public class WeightOnPlanetsV1 {

public static double [] gravity()throws IOException {

int iterate = 0;

double [] gravities = new double[9]; //initialize array

File fileName = new File("gravity1.txt");

Scanner inFile = new Scanner(fileName);

while (inFile.hasNext()){

gravities[iterate] = inFile.nextDouble();

iterate++;

}

inFile.close();

return gravities;

}

// Convert pounds to kilograms

static double gramsPerPound = 10.2040816;

static int pound = 160;

static double grams = pound / gramsPerPound;

public static double[] pounds(double[]gravities, double[]weight) {

for (int iterate = 0; iterate < 9; iterate++){

weight[iterate] = grams \* gravities[iterate];

}

return weight;

}

public static void print(double[] gravities, double[] weight, String[] planetNames) {

System.out.printf("%21s\n","My Weight on the Planets");

System.out.println("Planet Gravity Weight(lbs)");

for (int iterate = 0; iterate< 9; iterate ++){

System.out.printf(planetNames[iterate] + " %4.1f %4.1f",gravities[iterate],weight[iterate]);

System.out.println();

}

}

public static void main(String[] args) throws IOException {

String [] planet = {"Mercury", "Venus", "Earth", "Mars", "Jupiter", "Saturn", "Uranus", "Neptune", "Pluto"};

double gravities[] = gravity();

double[] weight = new double[9];

weight = pounds(gravities, weight);

print(gravities, weight, planet);

}

}