1. Nick Wright Problem 1
2. Java Code

import java.lang.Math;  
public class Main {  
  
  
 public static void main(String[] args) {  
 Solids[] MySolids = new Solids[10]; //creating the array MySolids  
  
 Cubes cube1 = new Cubes(4, 4, 4); //creating each of the solids  
 Cubes cube2 = new Cubes(8, 8, 8);  
 Spheres sphere1 = new Spheres(6, 6, 6);  
 Spheres sphere2 = new Spheres(3, 3, 3);  
 Cones cone1 = new Cones(5, 6, 6);  
 Cones cone2 = new Cones(3, 12, 12);  
 Parallelepipeds brick1 = new Parallelepipeds(3, 6, 9);  
 Parallelepipeds brick2 = new Parallelepipeds(2, 4, 6);  
 TCones tcone1 = new TCones(5, 6, 3); //5 is radius 1, 6 is height, 3 is radius 2  
 TCones tcone2 = new TCones(8, 4, 6); //8 is radius 1, 4 is height, 6 is radius 2  
  
 MySolids[0] = cube1; //assigning the different solids to the array  
 MySolids[1] = cube2;  
 MySolids[2] = sphere1;  
 MySolids[3] = sphere2;  
 MySolids[4] = cone1;  
 MySolids[5] = cone2;  
 MySolids[6] = brick1;  
 MySolids[7] = brick2;  
 MySolids[8] = tcone1;  
 MySolids[9] = tcone2;  
  
  
 for(int i=0; i<MySolids.length; i++){ //this loop will run through the array and calculate the area and volumes of all the solids  
 MySolids[i].CalcArea();  
 MySolids[i].CalcVolume();  
 }  
  
 System.*out*.println("Print each solid using array MySolids");  
  
 for(int i=0; i<MySolids.length; i++){ //this loop will run through the array and print the shapes  
// if(i>=1 && MySolids[i].getName().equals(MySolids[i-1].getName())){  
// System.out.println(MySolids[i].getName() + "2 " + MySolids[i].getArea() + " " + MySolids[i].getVolume());  
// }  
 System.*out*.println("Name: " + MySolids[i].getName() + " Length: " + MySolids[i].getLen() + " Width: " +  
 MySolids[i].getWid() + " Height: " + MySolids[i].getHig() + " Area: " + MySolids[i].getArea() + " Volume: " + MySolids[i].getVolume());  
// System.out.println(MySolids[i].getLen() + " " + MySolids[i].getWid() + " " + MySolids[i].getHig());  
 }  
  
 System.*out*.println("");  
 System.*out*.println("Calculate and print the total area and total volume of all solids");  
 double totalArea = 0.0; //creating a double to hold the value for the total area of all solids  
 double totalVolume = 0.0; //creating a double to hold the value for the total volume of all solids  
 for(int i=0; i<MySolids.length; i++){  
 totalArea = totalArea + MySolids[i].getArea();  
 totalVolume = totalVolume + MySolids[i].getVolume();  
 }  
 System.*out*.println("The total Area of all solids is " + totalArea + " and the total Volume of all solids is " + totalVolume);  
  
 System.*out*.println("\nFind and print the object with the maximum area");  
 //double maxArea = 0.0;  
 Solids maxArea = new Solids(); //I am making a new object of type Solids rather than a double of area so that I can return the name of the object along with the area  
 for(int i=0; i<MySolids.length; i++){ //for loop will iterate through the MySolids array and anytime there is an object with a larger area than the current largest area,  
 //the loop will assign that object from the array to Object maxArea  
 if(MySolids[i].getArea()>maxArea.getArea()){  
 maxArea = MySolids[i];  
 }  
 }  
 System.*out*.println("The shape with the maximum area is " + maxArea.getName() + " with an area of " + maxArea.getArea());  
  
 System.*out*.println("\nFind and print the object with the minimum volume");  
 Solids minVolume = new Solids();  
 minVolume = MySolids[0]; //setting the minVolume to the first Object in the array because none of the volumes are smaller than 0.  
 for(int i=1; i<MySolids.length; i++){ //int i can be set to 1 since we have already assigned minVolume the values from MySolids[0]  
 if(MySolids[i].getVolume()<minVolume.getArea()){  
 minVolume = MySolids[i];  
 }  
 }  
 System.*out*.println("The shape with the minimum volume is " + minVolume.getName() + " with a volume of " + minVolume.getVolume());  
  
 System.*out*.println("\nSort the array MySolids based on volume from smallest to largest and print the array");  
 *sortSmall*(MySolids,9); //calling upon the sortSmall method  
 for(int i=0; i<MySolids.length; i++){ //this loop will run through the array and print the shapes  
 System.*out*.println("Name: " + MySolids[i].getName() + " Length: " + MySolids[i].getLen() + " Width: " +  
 MySolids[i].getWid() + " Height: " + MySolids[i].getHig() + " Area: " + MySolids[i].getArea() + " Volume: " + MySolids[i].getVolume());  
 //System.out.println(MySolids[i].getName() + " " + MySolids[i].getArea() + " " + MySolids[i].getVolume());  
 }  
  
 }  
 public static void sortSmall(Solids []x, int xlast){ //this method will use the compareTo function to sort an array of object from smallest to largest  
 Solids xsave;  
 int isw=1;  
 while(isw==1){  
 isw=0;  
 for(int i=0; i<=xlast-1; i++){  
 switch(x[i].compareTo(x[i+1])){  
 case 1: //the objects are in the wrong order  
 xsave=x[i];  
 x[i]=x[i+1];  
 x[i+1]=xsave;  
 isw=1;  
 break;  
 case -1: //the objects in the array are in the right order  
 break;  
 default: //objects are equal, no change  
 }  
 }  
 }  
 }  
}  
class Solids extends Shapes{  
 protected int len; //the length of the solid  
 protected int wid; //the width of the solid  
 protected int hig; //the height of the solid  
 protected double area; //the surface area of the solid  
 protected double volume; //the volume of the solid  
  
  
 public int compareTo(Object x){  
 if(getVolume()>((Solids)x).getVolume()){  
 return 1;  
 } else if(getVolume()<((Solids)x).getVolume()){  
 return -1;  
 } else {  
 return 0;  
 }  
 }  
  
 public void CalcVolume() {  
 return;  
 //we will need to make this calculate the volume  
 }  
 public void CalcArea(){  
 return;  
 //we will need to make this calculate the area  
 }  
  
 @Override  
 public double getArea() {  
 return area;  
  
 }  
  
 @Override  
 public double getVolume() {  
 return volume;  
  
 }  
  
 @Override  
 public String getName() {  
 return null;  
  
 }  
  
 @Override  
 public int getLen() {  
 return len;  
 }  
  
 @Override  
 public int getWid() {  
 return wid;  
 }  
  
 @Override  
 public int getHig() {  
 return hig;  
 }  
}  
abstract class Shapes{  
 public abstract double getArea();  
 public abstract double getVolume();  
 public abstract String getName();  
 public abstract int getLen();  
 public abstract int getWid();  
 public abstract int getHig();  
 //abstract class shapes needs at least 3 functions  
}  
  
class Cubes extends Solids{  
 public Cubes(int len, int wid, int hig){  
 this.len = len;  
 this.wid = wid;  
 this.hig = hig;  
 }  
 public void CalcArea(){area = 6\*wid\*wid;}  
 public void CalcVolume(){volume = wid\*wid\*wid;}  
 public String getName(){  
 return "Cube";  
 }  
 //cube class that extends solids  
}  
class Spheres extends Solids{  
 public Spheres(int len, int wid, int hig){  
 this.len = len;  
 this.wid = wid;  
 this.hig = hig;  
 }  
 public void CalcArea(){area = 3.1416\*2.0\*(len/2.0)\*2.0\*(len/2.0);}  
 public void CalcVolume(){volume = (4.0/3.0)\*3.1416\*((len/2.0)\*(len/2.0)\*(len/2.0));}  
 public String getName(){ //this is the getName function that will return name Sphere  
 return "Sphere";  
 }  
 //spheres class that extends solids  
}  
class Cones extends Solids{  
 public Cones(int len, int wid, int hig){  
 this.len = len; //radius  
 this.wid = wid;  
 this.hig = hig; //height  
 }  
 public void CalcArea(){area = 3.1416\*len\*(len+Math.*sqrt*((hig\*hig)+(len\*len)));}  
 public void CalcVolume(){volume = 1.0/3.0\*3.1416\*(len\*len)\*hig;}  
 public String getName(){  
 return "Cone";  
 }  
 //cones class that extends solids  
}  
class Parallelepipeds extends Solids{  
 public Parallelepipeds(int len, int wid, int hig){  
 this.len = len;  
 this.wid = wid;  
 this.hig = hig;  
 }  
 public void CalcArea(){area = 2\*len\*wid+2\*wid\*hig+2\*hig\*len;}  
 public void CalcVolume(){volume = len\*wid\*hig;}  
 public String getName(){  
 return "Brick";  
 }  
 //parallelepipeds class that extends solids  
}  
class TCones extends Solids{  
 public TCones(int len, int wid, int hig){  
 this.len = len; //radius 1 (larger radius)  
 this.wid = wid; //height  
 this.hig = hig; //radius 2 (smaller radius)  
 }  
 public void CalcArea(){area = 3.1416\*(len\*len + hig\*hig) + 3.1416\*(len + hig)\*Math.*sqrt*(wid\*wid + ((len\*len)-(hig\*hig)));}  
 public void CalcVolume(){volume = 1.0/3.0\*3.1416\*wid\*(hig\*hig + len\*len + hig\*len);}  
 public String getName(){  
 return "TCone";  
 }  
 //TCones class that extends solids  
}

1. No input files were needed for this assignment.
2. Output Files:

Print each solid using array MySolids

Name: Cube Length: 4 Width: 4 Height: 4 Area: 96.0 Volume: 64.0

Name: Cube Length: 8 Width: 8 Height: 8 Area: 384.0 Volume: 512.0

Name: Sphere Length: 6 Width: 6 Height: 6 Area: 113.0976 Volume: 113.09759999999999

Name: Sphere Length: 3 Width: 3 Height: 3 Area: 28.2744 Volume: 14.137199999999998

Name: Cone Length: 5 Width: 6 Height: 6 Area: 201.22340190914173 Volume: 157.07999999999998

Name: Cone Length: 3 Width: 12 Height: 12 Area: 144.85273770096396 Volume: 113.0976

Name: Brick Length: 3 Width: 6 Height: 9 Area: 198.0 Volume: 162.0

Name: Brick Length: 2 Width: 4 Height: 6 Area: 88.0 Volume: 48.0

Name: TCone Length: 5 Width: 6 Height: 3 Area: 288.0495981919627 Volume: 307.87679999999995

Name: TCone Length: 8 Width: 4 Height: 6 Area: 605.9062363586546 Volume: 619.9423999999999

Calculate and print the total area and total volume of all solids

The total Area of all solids is 2147.403974160723 and the total Volume of all solids is 2111.2316

Find and print the object with the maximum area

The shape with the maximum area is TCone with an area of 605.9062363586546

Find and print the object with the minimum volume

The shape with the minimum volume is Sphere with a volume of 14.137199999999998

Sort the array MySolids based on volume from smallest to largest and print the array

Name: Sphere Length: 3 Width: 3 Height: 3 Area: 28.2744 Volume: 14.137199999999998

Name: Brick Length: 2 Width: 4 Height: 6 Area: 88.0 Volume: 48.0

Name: Cube Length: 4 Width: 4 Height: 4 Area: 96.0 Volume: 64.0

Name: Sphere Length: 6 Width: 6 Height: 6 Area: 113.0976 Volume: 113.09759999999999

Name: Cone Length: 3 Width: 12 Height: 12 Area: 144.85273770096396 Volume: 113.0976

Name: Cone Length: 5 Width: 6 Height: 6 Area: 201.22340190914173 Volume: 157.07999999999998

Name: Brick Length: 3 Width: 6 Height: 9 Area: 198.0 Volume: 162.0

Name: TCone Length: 5 Width: 6 Height: 3 Area: 288.0495981919627 Volume: 307.87679999999995

Name: Cube Length: 8 Width: 8 Height: 8 Area: 384.0 Volume: 512.0

Name: TCone Length: 8 Width: 4 Height: 6 Area: 605.9062363586546 Volume: 619.9423999999999

Process finished with exit code 0