Sahil A. Nawab – Introduction Exercises

BookStore.java

```
import java.util.Scanner;

public class BookStore {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		System.out.print("Please enter price: ");

		Scanner keyBoard = new Scanner(System.in);
		double price = keyBoard.nextDouble();
		double discount, finalPrice = -1;

		if (price < 128.0) {
			discount = price * 0.08;
		}
		else {
			discount = price * 0.16;
		}

		finalPrice = price - discount;
		System.out.println("Final Price: $" + finalPrice);
	} // end main method

} // end class BookStore
```

AWTGraphics.java

```
import java.applet.Applet;
import java.awt.Graphics;

public class AWTGraphics extends Applet {

	public void paint(Graphics g) {
		int x, y, width, height;

		// Cube
		g.drawRect(10, 10, 100, 100);
		g.drawRect(50, 50, 100, 100);

		g.drawLine(10, 10, 50, 50);
		g.drawLine(110, 10, 150, 50);
		g.drawLine(10, 110, 50, 150);
		g.drawLine(110, 110, 150, 150);

		// Initials
		g.fillRect(210, 10, 100, 10);
		g.fillRect(210, 10, 10, 50);
		g.fillRect(210, 50, 100, 10);
		g.fillRect(300, 50, 10, 80);
		g.fillRect(210, 130, 100, 10);
```

```
            g.fillRect(320, 10, 10, 130);
            g.fillRect(440, 10, 10, 130);

            x = 330; y = 20; width = 10; height = 10;
            for (int i = 0; i <= 10; i++) {
                  g.fillRect(x, y, width, height);
                  x += 10; y += 10;
            }

            // Pacman
            g.fillArc(160, 210, 100, 100, 30, 300);
            g.fillArc(10, 210, 100, 100, 210, 300);
            g.fillArc(85, 150, 100, 100, 120, 300);
            g.fillArc(85, 270, 100, 100, 300, 300);

            // Sphere
            x = 25; y = 25; width = 100; height = 100;
            for(int i = 0; i < 5; i++) {
                  g.drawOval(x, 25, width, 100);
                  g.drawOval(25, y, 100, height);
                  x += 10; y += 10; width -= 20; height -=20;
            }

            // Inscribed
            g.drawOval(400, 200, 200, 200);
            g.drawLine(500, 200, 400, 300);
            g.drawLine(400, 300, 500, 400);
            g.drawLine(500, 200, 500, 400);
            g.drawOval(418, 258, 82, 84);

      } // end method paint

} // end class AWTGraphics
```

Bricks.java

```
public class Bricks {

      public static void main(String[] args) {
            // TODO Auto-generated method stub
            System.out.print(makeBricks(4,1,8));
      } // end main method

      public static boolean makeBricks(int smallBricks, int largeBricks, int
goalLength) {
            final int smallBrickSize = 1; final int largeBrickSize = 5;

            int smallBricksLength = smallBricks * smallBrickSize;
            int largeBricksLength = largeBricks * largeBrickSize;

            if (smallBricksLength + largeBricksLength == goalLength) {
                  System.out.println("Method 1");
                  return true;
            }

            if (goalLength % smallBricksLength == 0 || goalLength %
largeBricksLength == 0) {
```

```
                System.out.println("Method 2");
                return true;
            }

            return false;
        } // end method makeBricks

} // end class Bricks
```

CreditCards.java

```
import java.util.Scanner;

public class CreditCards {

      public static void main(String[] args) {
            // TODO Auto-generated method stub
            Scanner keyBoard = new Scanner(System.in);

            System.out.print("Please enter credit card number: ");
            long cardNum = keyBoard.nextLong();

            long cardNumLength = Long.toString(cardNum).length();

            // sum every other digit
            long otherDigits = cardNum, otherDigitsSum = 0;
            for (long i = cardNumLength; i > 0; i -= 2) {
                  otherDigitsSum += otherDigits % 10;
                  otherDigits /= 100;
            }

            // double remaining digits and sum their digits
            long doubleDigits = cardNum / 10, doubleDigitsSum = 0;
            long doubleDigitsInternal;
            for (long i = cardNumLength - 1; i > 0; i -= 2) {
                  doubleDigitsInternal = (doubleDigits % 10) * 2;
                  for (byte j = 0; j < 2; j++) {
                        doubleDigitsSum += (doubleDigitsInternal % 10);
                        doubleDigitsInternal /= 10;
                  }
                  doubleDigits /= 100;
            }

            // check if last digit of sum equals zero
            long checkSum = otherDigitsSum + doubleDigitsSum;
            long checkSumLast = checkSum % 10;

            if (checkSumLast == 0) {
                  System.out.println("VALID");
            }
            else {
                  System.out.println("INVALID");
            }
      } // end main method

} // end class CreditCards
```

Darts.java

```
public class Darts {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		double numInside = 0, numOutside = 0;

		int iterations = 10000000;

		for (int i = 0; i < iterations; i++) {
			double distFromOrigin = Math.sqrt(Math.pow(genRandPoint(),
2) + Math.pow(genRandPoint(), 2));

			if (distFromOrigin <= 1) { // inside circle
				numInside++;
			}
			else { // outside circle
				numOutside++;
			}
		}

		System.out.println("Approximation for Pi: " + (numInside /
iterations) * 4);
	} // end main method

	public static double genRandPoint() {
		return Math.random() * 2 - 1;
	} // end method genRandPoint

} // end class Darts
```

DaysAlive.java

```
import java.util.GregorianCalendar;
import java.util.Calendar;
import java.util.Scanner;

public class DaysAlive {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Scanner userInput = new Scanner(System.in);

	    System.out.println("Enter date of birth");
	    System.out.print("Year: ");
	    int yearBorn = userInput.nextInt();
	    System.out.print("Month: ");
	    int monthBorn = userInput.nextInt();
	    System.out.print("Day: ");
	    int dayBorn = userInput.nextInt();
	    Calendar born = new GregorianCalendar(yearBorn, monthBorn - 1,
dayBorn);

	    long millis = System.currentTimeMillis();

	    double diff = millis - born.getTimeInMillis();
```

```
            diff = diff / (24 * 60 * 60 * 1000);

            System.out.println(); // blank line
            System.out.println("You have been alive for:");
            System.out.println("Approximately " + Math.round(diff) + " days");

      } // end main method

} // end class DaysAlive
```

DieSimulator.java

```
import java.applet.Applet;
import java.awt.Graphics;
import java.util.Random;

public class DieSimulator extends Applet {

      public void paint(Graphics g) {
            // Create die
            g.drawRect(10, 10, 100, 100);

            // Generate random number
            Random randy = new Random();
            long rand = randy.nextInt(6) + 1;
            System.out.println(rand);

            // Create ellipses
            for (int i = 0; i < rand; i++) {
                  g.fillOval(55, i * 15 + 20, 10, 10);
            }
      } // end paint method

} // end class DieSimulator
```

Fibonacci.java

```
public class Fibonacci {

      public static void main(String[] args) {
            // TODO Auto-generated method stub

            int sum = 0, x = 1, use;

            do {
                  use = genFib(x);
                  if (use % 2 == 0) {
                        sum += use;
                  }
                  x++;
            } while (use <= 4000000);

            System.out.println(sum);

      } // end main method

      public static int genFib(int n) {
```

```
		if (n == 1) return 1;
		if (n == 2) return 2;

		if (n > 2) {
			return genFib(n - 1) + genFib(n - 2);
		}

		return -1;
	} // end method genFib

} // end class Fibonacci
```

Gamble.java

```
import java.util.Scanner;

public class Gamble {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Scanner keyBoard = new Scanner(System.in);

		System.out.print("Please enter amount of money: $");
		double amount = keyBoard.nextDouble();

		int roll1, roll2, numRolls = 0;
		double greatest = amount;
		int greatestRoll = 0;

		while (amount > 0) {
			roll1 = (int) (Math.random() * 6) + 1;
			roll2 = (int) (Math.random() * 6) + 1;
			numRolls++;

			if (roll1 + roll2 == 7) {
				amount += 4;
				if (amount > greatest) {
					greatest = amount;
					greatestRoll = numRolls;
				}
			}
			else {
				amount -= 1;
			}
		}

		System.out.println("You would be broke after " + numRolls + "
rolls!");
		System.out.println("You should have quit after " + greatestRoll +
" rolls,");
		System.out.println("When you could have won $" + greatest);

	} // end main method

} // end class Gamble
```

Guessing.java

```
import java.util.Scanner;

public class Guessing {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		System.out.println("Created by Sahil Nawab and Miles Arnett");

		int compNum, userNum, guesses;
		boolean replay = false;
		boolean playAgain = false;

		Scanner keyBoard = new Scanner(System.in);

		do {
			compNum = (int) (Math.random() * 100) + 1;
			guesses = 0;

			System.out.print("Please guess a number between 1 and 100
(or enter -1 to quit): ");

			do {
				userNum = keyBoard.nextInt();

				if (userNum == -1) {
					replay = false; playAgain = false;

					System.out.println("Thank you for playing!");
					System.out.println("Created by Sahil Nawab and
Miles Arnett");
					return;
				}

				if (userNum == compNum) {
					System.out.println("You guessed correctly: " +
userNum);
					replay = false; guesses++;

					System.out.println("You guessed " + guesses + "
times!");

					keyBoard.nextLine(); // absorb extra newline
					System.out.print("Would you like to play again?
(Y/N): ");
					String userPlayAgain = keyBoard.nextLine();

					if (userPlayAgain.equalsIgnoreCase("y") ||
userPlayAgain.equalsIgnoreCase("yes")) {
						playAgain = true;
					}
					else {
						playAgain = false;
					}
				}
				else if (userNum > compNum) {
```

```
                                    replay = true; guesses++;
                                    System.out.println("You guessed incorrectly: "
+ userNum + " is too big!");
                                    System.out.print("Please guess again: ");
                              }
                              else if (userNum < compNum) {
                                    replay = true; guesses++;
                                    System.out.println("You guessed incorrectly: "
+ userNum + " is too small!");
                                    System.out.print("Please guess again: ");
                              }
                        } while(replay);
                  } while (playAgain);

                  System.out.println("Thank you for playing!");
                  System.out.println("Created by Sahil Nawab and Miles Arnett");
            } // end main method

} // end class Guessing
```

LineArt.java

```
import java.applet.Applet;
import java.awt.Color;
import java.awt.Graphics;

public class LineArt extends Applet {

      public void paint(Graphics g) {
            int height = 630, width = 980;
            int xInc = width / 70, yInc = height / 70;
            int x1 = 0, y1 = 0, x2 = 0, y2 = 0;
            g.drawRect(10, 10, width, height);

           // bottom left corner
            x1 = 10; y1 = 10; x2 = 10; y2 = height + 10;
            for (int i = 0; i < 710; i += 10) {
                  g.setColor(genRandColor());
                  g.drawLine(x1, y1, x2, y2);
                  y1 += yInc; x2 += xInc;
            }

            // bottom right corner
            x1 = width + 10; y1 = 10; x2 = width + 10; y2 = height + 10;
            for (int i = 0; i < 710; i += 10) {
                  g.setColor(genRandColor());
                  g.drawLine(x1, y1, x2, y2);
                  y1 += yInc; x2 -= xInc;
            }

            // top right corner
            x1 = width + 10; y1 = height + 10; x2 = width + 10; y2 = 10;
            for (int i = 0; i < 710; i += 10) {
                  g.setColor(genRandColor());
                  g.drawLine(x1, y1, x2, y2);
                  y1 -= yInc; x2 -= xInc;
            }
```

```
            // top left corner
            x1 = 10; y1 = height + 10; x2 = 10; y2 = 10;
            for (int i = 0; i < 710; i += 10) {
                  g.setColor(genRandColor());
                  g.drawLine(x1, y1, x2, y2);
                  y1 -= yInc; x2 += xInc;
            }

            /*------------------------*/
            height = 630 / 2; width = 980 / 2;
            xInc = width / 35; yInc = height / 35;
            x1 = 0; y1 = 0; x2 = 0; y2 = 0;
            g.drawRect(980/4+10, 630/4+10, width, height);

            // bottom left corner / 4
            x1 = 980/4 + 10; y1 = 630/4 + 10; x2 = 980/4 +  10; y2 = 630/4 +
height + 10;
            for (int i = 0; i < 710 / 2; i += 10) {
                  g.setColor(genRandColor());
                  g.drawLine(x1, y1, x2, y2);
                  y1 += yInc; x2 += xInc;
            }

            // bottom right corner / 4
            x1 = 980/4 + width + 10; y1 = 630/4 + 10; x2 = 980/4 + width +
10; y2 = 630/4 + height + 10;
            for (int i = 0; i < 710 / 2; i += 10) {
                  g.setColor(genRandColor());
                  g.drawLine(x1, y1, x2, y2);
                  y1 += yInc; x2 -= xInc;
            }

            // top right corner / 4
            x1 = 980/4 + width + 10; y1 = 630/4 + height + 10; x2 = 980/4 +
width + 10; y2 = 630/4 + 10;
            for (int i = 0; i < 710 / 2; i += 10) {
                  g.setColor(genRandColor());
                  g.drawLine(x1, y1, x2, y2);
                  y1 -= yInc; x2 -= xInc;
            }

            // top left corner / 4
            x1 = 980/4 + 10; y1 = 630/4 + height + 10; x2 = 980/4 + 10; y2 =
630/4 + 10;
            for (int i = 0; i < 710 / 2; i += 10) {
                  g.setColor(genRandColor());
                  g.drawLine(x1, y1, x2, y2);
                  y1 -= yInc; x2 += xInc;
            }

      } // end paint method

      public Color genRandColor() {
            int randRed = (int) (Math.random() * 256),
                  randGreen = (int) (Math.random() * 256),
                  randBlue = (int) (Math.random() * 256);
```

```
            return new Color(randRed, randGreen, randBlue);
      } // end method genRandColor

} // end class LineArt
```

Manipulator.java

```
import java.awt.Color;
import java.util.Scanner;

public class Manipulator {

      public static void main(String[] args) {
            // TODO Auto-generated method stub
            Scanner keyBoard = new Scanner(System.in);
            Picture pic = new Picture();

            System.out.print("Load an image from computer or web: ");
            String userInput = keyBoard.nextLine();

            if (userInput.equalsIgnoreCase("computer")) {
                  pic.pick();
            }
            else if (userInput.equalsIgnoreCase("web")) {
                  System.out.print("Please enter url of image: ");
                  String image = keyBoard.nextLine();

                  pic.load(image);
            }
            else {
                  System.out.println("INVALID INPUT");
            }

            int height = pic.getHeight(), width = pic.getWidth();

            for (int i = 0; i < width; i++) {
                  for (int j = 0; j < height; j++) {
                        Color orig = pic.getColorAt(i, j);
                        pic.setColorAt(i, j, getNegative(orig));
                  }
            }

      } // end main method

      public static Color getNegative(Color original) {
            int red = 255 - original.getRed();
            int green = 255 - original.getGreen();
            int blue = 255 - original.getBlue();
            Color negative = new Color(red, green, blue);
            return negative;
      }

} // end class Manipulator
```

MiddleString.java

```
import java.util.Scanner;

public class MiddleString {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		System.out.println("Please enter string input: ");

		Scanner keyBoard = new Scanner(System.in);
		String input = keyBoard.nextLine();
		int length = input.length(), halfLength = length / 2;

		if (length % 2 == 0) {
			System.out.println(input.substring(halfLength - 1,
halfLength + 1));
		}
		else {
			System.out.println(input.substring(halfLength, halfLength +
1));
		}
		// S T R I N G S --> length = 7
		// 0 1 2 3 4 5 6 --> index  = 6
	}

} // end class MiddleString
```

MinOfThree.java

```
import java.util.Scanner;

public class MinOfThree {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		Scanner keyBoard = new Scanner(System.in);

		System.out.print("Please enter first integer: ");
		int one = keyBoard.nextInt();

		System.out.print("Please enter second integer: ");
		int two = keyBoard.nextInt();

		System.out.print("Please enter third integer: ");
		int three = keyBoard.nextInt();

		if (one < two && one < three) {
			System.out.println("The first integer, " + one + ", is the
smallest integer.");
		}
		else if (two < one && two < three) {
			System.out.println("The second integer, " + two + ", is the
smallest integer.");
		}
		else if (three < one && three < two) {
```

```
                  System.out.println("The third integer, " + three + ", is
the smallest integer.");
            }

      } // end main method

} // end class MinOfThree
```

Mortgage.java

```
public class Mortgage {

      public static void main(String[] args) {
            // TODO Auto-generated method stub
            System.out.println("Lab03: The Mortgage Payment Program");

            double principal = 259000, annualRate = 5.75, years = 30, months
= years * 12;
            double monthlyRate = annualRate / 100.0; monthlyRate /= 12;

            System.out.println("Principal:    $" + principal);
            System.out.println("N of Years:    " + years);
            System.out.println("Annual Rate:   " + annualRate);
            System.out.println(); // add blank line for organization

            double monthly = calcMonthly(principal, monthlyRate, months);
            double rmonthly = (Math.round(monthly*100))/100.0;
            double total = months * rmonthly;
            double interest = total - principal;

            System.out.println("Monthly Payment:  $" + rmonthly);
            System.out.println("Total Payment:    $" + total);
            System.out.println("Total Interest:   $" + interest);
      } // end main method

      public static double calcMonthly(double p, double r, double m) {
            double n = Math.pow((1 + r), m);
            return ((r * n) / (n - 1)) * p;
      } // end method calcMonthly

} // end class Mortgage
```

Multiples.java

```
public class Multiples {

      public static void main(String[] args) {
            // TODO Auto-generated method stub

            int i, x = 0;

            for (i = 0; i < 1000; i++) {
                  if (i % 3 == 0 || i % 5 == 0) {
                        x += i;
                  }
            }
```

```
            System.out.println(x);
      }

} // end class Multiples
```

One.java

```
public class One {

      public static void main(String[] args) {
            // TODO Auto-generated method stub
            animal(); System.out.println();
            spikey(); System.out.println();
      } // end main method

      public static void animal() {
            System.out.println("  /\\_/\\      -----");
            System.out.println(" ( ' ' )   / Hello \\");
            System.out.println(" (  u  ) < Amazing >");
            System.out.println("  | | |    \\ Coder /");
            System.out.println(" (__|__)     -----");
      }

      public static void spikey() {
            System.out.println("   \\/");
            System.out.println(" \\\\\//");
            System.out.println("\\\\\\\\\///");
            System.out.println("///\\\\\\\");
            System.out.println(" //\\\\");
            System.out.println("   /\\");
      }

} // end class One
```

Paycheck.java

```
import java.util.Scanner;

public class Paycheck {

      public static void main(String[] args) {
            // TODO Auto-generated method stub
            Scanner keyBoard = new Scanner(System.in);
            double weeklyPay = 0;

            System.out.print("Please enter hourly wage: ");
            double wage = keyBoard.nextDouble();

            System.out.print("Please enter hours worked: ");
            double hours = keyBoard.nextDouble();

            if (hours < 0) {
                  System.out.println("An invalid input is received.");
            }
            else if (hours <= 40) {
                  weeklyPay = wage * hours;
            }
```

```
            else if (hours > 40) {
                  weeklyPay = wage * 40 + (hours - 40) * (1.5 * wage);
            }

            System.out.println("Weekly Pay: $" + weeklyPay);

      } // end main method

} // end class Paycheck
```

Picture.java

```
import java.awt.Color;
import java.io.File;
import java.net.URL;
import java.awt.image.BufferedImage;
import java.awt.image.ColorModel;
import java.awt.image.Raster;
import java.awt.image.WritableRaster;
import javax.imageio.ImageIO;
import javax.swing.ImageIcon;
import javax.swing.JFileChooser;
import javax.swing.JFrame;
import javax.swing.JLabel;

/**
   A picture whose pixels can be read and written.
*/
public class Picture
{
   private String source;
   private JFrame frame;
   private JLabel label;
   private BufferedImage image;

   /**
      Constructs a picture with no image.
   */
   public Picture()
   {
      frame = new JFrame();
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      label = new JLabel("(No image)");
      frame.add(label);
      frame.pack();
      frame.setVisible(true);
   }

   /**
      Gets the width of this picture.
      @return the width
   */
   public int getWidth() { return image.getWidth(); }

   /**
      Gets the height of this picture.
      @return the height
```

```
*/
public int getHeight() { return image.getHeight(); }

/**
   Loads a picture from a given source.
   @param source the image source. If the source starts
   with http://, it is a URL, otherwise, a filename.
*/
public void load(String source)
{
   try
   {
      this.source = source;
      BufferedImage img;
      if (source.startsWith("http://"))
         img = ImageIO.read(new URL(source).openStream());
      else
         img = ImageIO.read(new File(source));

      setImage(img);
   }
   catch (Exception ex)
   {
      this.source = null;
      ex.printStackTrace();
   }
}

/**
   Reloads this picture, undoing any manipulations.
*/
public void reload()
{
   load(source);
}

/**
   Displays a file chooser for picking a picture.
*/
public void pick()
{
   JFileChooser chooser = new JFileChooser(".");
   if (chooser.showOpenDialog(null) == JFileChooser.APPROVE_OPTION)
   {
      load(chooser.getSelectedFile().getAbsolutePath());
   }
}

private void setImage(BufferedImage image)
{
   this.image = image;
   label.setIcon(new ImageIcon(image));
   label.setText(" ");
   frame.pack();
}

/**
```

```
      Gets the color of a pixel.
      @param x the column index (between 0 and getWidth() - 1)
      @param y the row index (between 0 and getHeight() - 1)
      @return the color of the pixel at position (x, y)
   */
   public Color getColorAt(int x, int y)
   {
      Raster raster = image.getRaster();
      ColorModel model = image.getColorModel();
      int argb = model.getRGB(raster.getDataElements(x, y, null));
      return new Color(argb, true);
   }

   /**
      Sets the color of a pixel.
      @param x the column index (between 0 and getWidth() - 1)
      @param y the row index (between 0 and getHeight() - 1)
      @param c the color for the pixel at position (x, y)
   */
   public void setColorAt(int x, int y, Color c)
   {
      WritableRaster raster = image.getRaster();
      ColorModel model = image.getColorModel();
      Object colorData = model.getDataElements(c.getRGB(), null);
      raster.setDataElements(x, y, colorData);
      label.repaint();
   }
}
```

Prog01.java

```
public class Prog01 {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		int hours = 52;
		double rate = 8.25;
		double weeklyPay = 0;

		if (hours < 0) {
			System.out.println("An invalid input is received.");
		}
		else if (hours <= 40) {
			weeklyPay = rate * hours;
		}
		else if (hours > 40) {
			weeklyPay = rate * 40 + (hours - 40) * (1.5 * rate);
		}

		System.out.println("Weekly Pay: $" + weeklyPay);
	} // end main method

} // end class Prog01
```

Prog02.java

```
import java.applet.Applet;
```

```
import java.awt.Color;
import java.awt.Graphics;

public class Prog02 extends Applet {

	public void paint(Graphics g) {
		// draw maze
		g.setColor(Color.black);
		g.fillRect(0,0,800,600);
		g.setColor(Color.cyan);
		g.fillRect(0,80,800,20);
		g.fillRect(0,500,800,20);

		// draw snacks
		g.setColor(Color.white);

		for (int i = 0; i < 3; i++) {
			g.fillOval((i+5)*100, (600 - 80) / 2, 80, 80);
		}

		// draw pacman
		g.setColor(Color.yellow);
		g.fillArc((800 - 300) / 2, (600 - 300) / 2, 300, 300, 45, 270);

	} // end paint method

} // end class Prog02
```

Prog03.java

```
public class Prog03 {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		double a, b;

		a = 64; b = 0;
		System.out.println("The square root of " + a + " is " +
Math.sqrt(a));

		a = 7; b = 0;
		System.out.println("The absolute value of " + a + " is " +
Math.abs(a));

		a = -7; b = 0;
		System.out.println("The absolute value of " + a + " is " +
Math.abs(a));

		a = 3; b = 5;
		System.out.println(a + " to the " + b + "th power is " +
Math.pow(a, b));

		a = 5; b = 3;
		System.out.println(a + " to the " + b + "th power is " +
Math.pow(a, b));

		a = 69.999; b = 0;
```

```
            System.out.println(a + " rounded down is " + Math.floor(a));

            a = 69.001; b = 0;
            System.out.println(a + " rounded up is " + Math.ceil(a));

            a = 69.499; b = 0;
            System.out.println(a + " rounded to the nearest integer is " +
Math.round(a));

            a = 69.500; b = 0;
            System.out.println(a + " rounded to the nearest integer is " +
Math.round(a));

      } // end main method

} // end class Prog03
```

QuadraticFormula.java

```
import java.util.Scanner;

public class QuadraticFormula {

      public static void main(String [] args) {
          Scanner s = new Scanner(System.in);
          System.out.print("Please enter value for a: ");
          double a = s.nextDouble();
          System.out.print("Please enter value for b: ");
          double b = s.nextDouble();
          System.out.print("Please enter value for c: ");
          double c = s.nextDouble();

          if (b > 0 && c > 0 ){
              System.out.println(a + "x^2 + " + b + "x + " + c + " = 0");}
          if (b < 0 && c > 0 ){
              System.out.println(a + "x^2 " + b + "x + " + c + " = 0");}
          if (b > 0 && c < 0 ){
              System.out.println(a + "x^2 + " + b + "x " + c + " = 0");}
          if (b < 0 && c < 0 ){
              System.out.println(a + "x^2 " + b + "x " + c + " = 0");}

          double answer1 = (-b + Math.sqrt(Math.pow(b, 2) - (4 * a * c))) /
(2 * a);
          double answer2 = (-b - Math.sqrt(Math.pow(b, 2) - (4 * a * c))) /
(2 * a);

          if (Double.isNaN(answer1) || Double.isNaN(answer2)) {
              System.out.println("Answer contains imaginary numbers");
          }
          else if (answer1 == answer2) {
            System.out.println("The value is: " + answer1);
          }
          else {
            System.out.println("The values are: " + answer1 + ", " +
answer2);
          }
      } // end main method
```

```
} // end class QuadraticFormula
```

RockPaperScissors.java

```
import java.util.Scanner;

public class RockPaperScissors {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		int userWins = 0, userLosses = 0;
		boolean replay ;

		int computerChoice;

		String userChoiceStr;
		int userChoiceNum;

		do {
			replay = false;

			// compute computerChoice
			computerChoice = (int) (Math.random() * 3) + 1;

			// get user input
			System.out.print("Please enter your choice: ");

			Scanner keyBoard = new Scanner(System.in);
			userChoiceStr = keyBoard.nextLine();

			if (computerChoice == 1) {
				System.out.println("The computer chose Rock!");
			}
			else if (computerChoice == 2) {
				System.out.println("The computer chose Paper!");
			}
			else if (computerChoice == 3) {
				System.out.println("The computer chose Scissors!");
			}

			int length = userChoiceStr.length();
			userChoiceNum = -1;

			if (length == 4) {
				userChoiceNum = 1; // equates to rock
				if (computerChoice == 2) {
					System.out.println("Sorry, you lost!");
					userLosses++;
				}
				else if (computerChoice == 3) {
					System.out.println("Congratulations, you
won!");

					userWins++;
				}
			}
			else if (length == 5) {
				userChoiceNum = 2; // equates to paper
```

```
                        if (computerChoice == 3) {
                              System.out.println("Sorry, you lost!");
                              userLosses++;
                        }
                        else if (computerChoice == 1) {
                              System.out.println("Congratulations, you
won!");
                              userWins++;
                        }
                  }
                  else if (length == 8) {
                        userChoiceNum = 3; // equates to scissors
                        if (computerChoice == 1) {
                              System.out.println("Sorry, you lost!");
                              userLosses++;
                        }
                        else if (computerChoice == 2) {
                              System.out.println("Congratulations, you
won!");
                              userWins++;
                        }
                  }
                  else {
                        System.out.println("You have entered an invalid
input.");
                  }

                  // compare userChoiceNum to computerChoice
                  if (userChoiceNum == computerChoice) {
                        System.out.println("It\'s a Tie!");
                  }

                  // play again?
                  System.out.print("Would you like to play again? (Y/N): ");
                  String again = keyBoard.nextLine();

                  again = again.toLowerCase();

                  if (again.equals("y") || again.equals("yes")) {
                        replay = true;
                  }
            } while(replay);

            System.out.println("Your Score:");

            System.out.println("Wins: " + userWins + " | Losses: " +
userLosses);
      } // end main method

} // end class RockPaperScissors
```

Six.java

```
public class Six {

      public static void main(String[] args) {
            // TODO Auto-generated method stub
```

```
            System.out.println("Hello, world!");
            message();

      } // end main method

      public static void message() {
            System.out.println("This program surely cannot ");
            System.out.println("have any errors in it");
      }

} // end class Six
```

Summation.java

```
import java.util.Scanner;

public class Summation {

      public static void main(String[] args) {
            // TODO Auto-generated method stub
            Scanner keyBoard = new Scanner(System.in);

            System.out.print("Please enter a number: ");
            int userInput = keyBoard.nextInt();

            long sum = 0;

            for (int i = 0; i <= userInput; i++) {
                  sum += i;
            }

            System.out.println(sum);
      } // end main method

} // end class Summation
```

Taxes.java

```
import java.util.Scanner;

public class Taxes {

      public static void main(String[] args) {
            // TODO Auto-generated method stub
            Scanner keyBoard = new Scanner(System.in);
            double tax = 0;

            System.out.print("Please enter total income: ");
            double income = keyBoard.nextDouble();

            keyBoard.nextLine(); // absorb extra newline char
            System.out.print("Please enter marital status (Y/N): ");
            String married = keyBoard.nextLine();

            if (married.equalsIgnoreCase("y") ||
married.equalsIgnoreCase("yes")) {
                  if (income <= 64000 && income > 0) {
```

```
                        tax = income * 0.10;
                  }
                  else if (income > 64000) {
                        tax = income * 0.25;
                  }
                  else {
                        System.out.println("You have entered an invalid
input.");
                        return;
                  }
            }
            else if (married.equalsIgnoreCase("n") ||
married.equalsIgnoreCase("no")) {
                  if (income <= 32000 && income > 0) {
                        tax = income * 0.10;
                  }
                  else if (income > 32000) {
                        tax = income * 0.25;
                  }
                  else {
                        System.out.println("You have entered an invalid
input.");
                        return;
                  }
            }
            else {
                  System.out.println("You have entered an invalid input.");
                  return;
            }

            System.out.println("Tax Payment: $" + tax);
      } // end main method

} // end class Taxes
```

TaxesAdvanced.java

```
import java.util.Scanner;

public class TaxesAdvanced {

      public static void main(String[] args) {
            // TODO Auto-generated method stub
            Scanner keyBoard = new Scanner(System.in);
            double tax = 0;

            System.out.print("Please enter total income: ");
            double income = keyBoard.nextDouble();

            keyBoard.nextLine(); // absorb extra newline char
            System.out.print("Please enter marital status (Y/N): ");
            String married = keyBoard.nextLine();

            if (married.equalsIgnoreCase("n") ||
married.equalsIgnoreCase("no")) {
                  if (income >= 0 && income <= 9705 ) {
                        tax = income * 0.10;
```

```
                }
                else if (income > 9075 && income <= 36900) {
                      tax = (income - 9075) * 0.15 + 907.50;
                }
                else if (income > 36900 && income <= 89350) {
                      tax = (income - 39600) * 0.25 + 5081.25;
                }
                else if (income > 89350 && income <= 186350) {
                      tax = (income - 89350) * 0.28 + 18193.75;
                }
                else if (income > 186350 && income <= 405100) {
                      tax = (income - 186350) * 0.33 + 45353.57;
                }
                else if (income > 405100 && income <= 406750) {
                      tax = (income - 405100) * 0.35 + 117541.25;
                }
                else if (income > 406750) {
                      tax = (income - 406750) * 0.396 + 118118.75;
                }
                else {
                      System.out.println("You have entered an invalid
input.");
                      return;
                }
          }
          else if (married.equalsIgnoreCase("y") ||
married.equalsIgnoreCase("yes")) {
                if (income > 0 && income <= 18150 ) {
                      tax = income * 0.10;
                }
                else if (income > 18150 && income <= 73800) {
                      tax = (income - 1815.00) * 0.15 + 1815.00;
                }
                else if (income > 73800 && income <= 148850) {
                      tax = (income - 73800) * 0.25 + 10162.50;
                }
                else if (income > 148850 && income <= 226850) {
                      tax = (income - 148850) * 0.28 + 28925.00;
                }
                else if (income > 226850 && income <= 405100) {
                      tax = (income - 226850) * 0.33 + 50765.00;
                }
                else if (income > 405100 && income <= 457600) {
                      tax = (income - 405100) * 0.35 + 109587.50;
                }
                else if (income > 457600) {
                      tax = (income - 457600) * 0.396 + 127962.50;
                }
                else {
                      System.out.println("You have entered an invalid
input.");
                      return;
                }
          }
          else {
                System.out.println("You have entered an invalid input.");
                return;
```

```
            }

            System.out.println("Tax is approximately: $" + Math.round((tax) *
100.0) / 100.0);
      } // end main method

} // end class TaxesAdvanced
```

Three.java

```
public class Three {

      public static void main(String[] args) {
            // TODO Auto-generated method stub
            System.out.println("An Emergency Broadcast");
      } // end main method

} // end class Three
```

TimeDisplay.java

```
public class TimeDisplay {

      public static void main(String[] args) {
            // TODO Auto-generated method stub
            System.out.println("Lab02: The Time Display Program");
            int milliseconds = 10000123;
            int hours = 0, minutes = 0, seconds = 0;

            seconds = milliseconds / 1000;
            milliseconds %= 1000;

            minutes = seconds / 60;
            seconds %= 60;

            hours = minutes / 60;
            minutes %= 60;

            System.out.println("Hours:        " + hours);
            System.out.println("Minutes:      " + minutes);
            System.out.println("Seconds:      " + seconds);
            System.out.println("Milliseconds: " + milliseconds);
      } // end main method

} // end class TimeDisplay
```

Uppercase.java

```
import java.util.Scanner;

public class Uppercase {

      public static void main(String[] args) {
            // TODO Auto-generated method stub
            Scanner keyBoard = new Scanner(System.in);

            System.out.print("Please enter a string: ");
```

```
            String userInput = keyBoard.nextLine();

            int counter = 0;

            for (int i = 0; i < userInput.length(); i++) {
                  if (Character.isUpperCase(userInput.charAt(i))) {
                        counter++;
                  }
            }

            if (counter == 0) {
                  System.out.println("No uppercase characters found!");
            }
            else {
                  System.out.println(counter + " uppercase characters
found!");
            }
      } // end main method

} // end class Uppercase
```