Week 1

Task 2: Run, understand, and modify four simple java programs

1. Input data

The program 'InputData.java' imports the Scanner so that a user input can be read and I edited the code to import the Calendar so that the current date could be used. The variables are defined at the start of the code. The user is asked to input their name, which is assigned to the string variable 'name' and then the program says hello to them! Next, the user is asked to input their height in meters and this is assigned to the double 'height'. The program prints their name and height in cm (height [m] x 100 = height [cm]). Finally, the user is asked for their age, which is assigned to the integer 'age', and estimates their age by subtracting it from the current date. I accessed the current year using `int year = Calendar.getInstance().get(Calendar.YEAR)`. The output is shown below:

```
run:
Please type in your name
Scarlett

Hello Scarlett
Please type in your height in metres
1.7
Thank you, your height is 170.0 cm
Please type in your age in years
23
You were probably born in 1997
BUILD SUCCESSFUL (total time: 14 seconds)
```

I then edited the code to convert the input height from meters into feet and inches and report that result to the user (rather than their height in cm). I did this using the following method: I multiplied the height by 3.281 ft/m, extracted the integer part of the decimal result using the Math.floor() function, extracted the decimal part by finding the remainder when divided by 1, then multiplied this by 12 inches/ft to convert the remainder to inches. This is also shown in the code below:

```
height = keyboard.nextDouble(); //the height of the user in m
feetDec = height * 3.281; //calculates the height in feet as a decimal
feet = (int)Math.floor(feetDec); //extracts the integer part of the decimal
inches = (int)Math.round((feetDec % 1) * 12); //calculates the remainder of feetDec
 / 1 and * by 12 to convert to inches, then rounds to the nearest integer
```

This was the output of the edited code:

```
run:
Please type in your name
Scarlett

Hello Scarlett
Please type in your height in meters
1.7
```

```
Thank you, your height is 5 feet and 7 inches
Please type in your age in years
23
You were probably born in 1997
BUILD SUCCESSFUL (total time: 22 seconds)
```

2. Primitive data types

This program declares some primitive data types and then prints them. The output is shown below:

```
run:
The value of b = false
The value of c = R
The value of j = 127
The value of k = 32767
The value of m = 2147483647
The value of n = 9223372036854775807
The value of x = 3.1415927
The value of y = 3.141592653589793
BUILD SUCCESSFUL (total time: 0 seconds)
```

I then added 1 to the value of m and observed the effect. As shown in the output below, m then became negative and increased in magnitude by 1. This is because int is a 32 bit integer between -2,147,483,648 (-2^{31}) and 2,147,483,647 ($2^{31}-1$). Adding 1 to m = 2,147,483,648 adds 1 in binary, making m = -2,147,483,648.

```
run:
The value of b = false
The value of c = R
The value of j = 127
The value of k = 32767
The value of m = -2147483648
The value of n = 9223372036854775807
The value of x = 3.1415927
The value of y = 3.141592653589793
BUILD SUCCESSFUL (total time: 0 seconds)
```

3. Math Examples

This program calculates the area of a circle with a radius, r = 12.33 and prints the result. It then calculates the angle between a line from the origin to (x, y) = (-1, -2) and the x-axis. It also calculates the length of the line, and prints both results. There were a few errors in the code, which are highlighted, described and corrected below:

```
1.  package mathexamples;
2.  
3.  public class MathExamples {
4.  
```

```java

    public static void main(String[] args) {
         // Find the square root of a number
        double r = 12.33;
        double ans = Math.sqrt(R);
        System.out.println(" The square root of " + r + "  is = " + ans);

        double area = Math.PI * r * r;
        System.out.println(" Area of circle with radius " + r + " is =  + area);

        /* The method Atan2(y,x) gives the angle of line from the
           origin to (x,y) in the correct quadrant */
        double x = -1;
        double y = -2;
        double angle = Math.atan2(y, x) // Math.sqrt(x*x+y*y) would work as well
        double length = Math.sqrt(Math.pow(x, 2) + Math.pow(y, 2));

        // returns angle in RADIANS in the range -PI < angle <PI
        System.out.println(" Angle = " + angle + " rad, or "
                + angle * 180 / Math.PI + " degrees");
        System.out.println(" Length = " + length);

        // convert back
        System.out.println(" (x,y) = ("
                + (length * Math.sin(angle)) + ", "
                + (length * Math.cos(angle)) + ")");
    }

}
```

The errors were:

- R needed to be switched to lower case r
- “ is = “ needs the parenthesis to be closed
- A semicolon was required after “x)”
- The sine and cosine functions needed to be switched around as sine corresponds to y and cosine to x

This is the corrected code and output:

```java
package mathexamples;

public class MathExamples {


    public static void main(String[] args) {
         // Find the square root of a number
        double r = 12.33;
        double ans = Math.sqrt(r);
        //switched R -> r
        System.out.println(" The square root of " + r + "  is = " + ans);

        double area = Math.PI * r * r;
        System.out.println(" Area of circle with radius " + r + " is  = " + area);

        //added missing ending double quotation marks

        /* The method Atan2(y,x) gives the angle of line from the
           origin to (x,y) in the correct quadrant */
        double x = -1;
        double y = -2;
        double angle = Math.atan2(y, x); // Math.sqrt(x*x+y*y) would work as well
        //added semicolon to end of line
        double length = Math.sqrt(Math.pow(x, 2) + Math.pow(y, 2));

        // returns angle in RADIANS in the range -PI < angle <PI
        System.out.println(" Angle = " + angle + " rad, or "
                + angle * 180 / Math.PI + " degrees");
        System.out.println(" Length = " + length);

        // convert back
        System.out.println(" (x,y) = ("
                + (length * Math.cos(angle)) + ", "
                + (length * Math.sin(angle)) + ")");
        //swapped sine and cosine around
    }

}
```

```
run:
 The square root of 12.33  is = 3.5114099732158874
 Area of circle with radius 12.33 is  = 477.6128753733373
 Angle = -2.0344439357957027 rad, or -116.56505117707799 degrees
 Length = 2.23606797749979
 (x,y) = (-1.0, -2.0)
BUILD SUCCESSFUL (total time: 0 seconds)
```

4. Simple functions

The program ‘SimpleFunctions.java’ contains a function which calculates and returns the average of the two user inputted doubles x and y. Using the same method, I added a function which calculates the absolute difference between x and y and returns the result. The average and difference were then printed to the screen. The code and output are shown below:

```java
package simplefunctions;

import java.util.Scanner;

public class SimpleFunctions {
    static Scanner keyboard = new Scanner(System.in);

    static double average(double x, double y) {
        /*
        This function takes two double inputs and calculates the average
        */
        double sum;
        sum = x + y;
        return (sum / 2); //average = sum / (number of inputs)
    }

    static double difference(double x, double y) {
        /*
        This function takes two double inputs and calculates the difference between
 them
        */
        return Math.abs(x - y); //the difference = the absolute value of x minus y

    }

    // Example of program with simple methods method
    public static void main(String[] args) {
        System.out.println("Please type in the first number    ");
        double first = keyboard.nextDouble();
        System.out.println("Please type in the second number   ");
        double second = keyboard.nextDouble();

        double av = average(first, second); //the average of the inputs
        double diff = difference(first, second); //the difference between the inputs
        System.out.println("The average of these two numbers is " + av + " and the difference between them is " + diff);
    }

}
```

```
run:
Please type in the first number
9
Please type in the second number
8
The average of these two numbers is 8.5 and the difference between them is 1.0
BUILD SUCCESSFUL (total time: 11 seconds)
```

Task 3: Relativistic Kinematics

In this task, a Java Class was made called ‘FourVector,’ which contained four double variables to store E, p_x, p_y and p_z of a particle’s four vector. I then added extra variables to store, including the magnitude of the field in T. I calculated the magnitude of the particle’s momentum, its mass, the Lorentz factor (gamma), particle speed in units of c (beta), and the radius of its path. This code is shown below:

```java
package RelativisticKinematics;

class FourVector {
    // store the components of the FourVector
    double E;
    double px;
    double py;
    double pz;
    double field;
    double m;
    double p;
    double B;

    double momentum()
    {
        p = Math.sqrt(px*px+py*py+pz*pz);
        return p;
    }

    double mass()
    {
        m = Math.sqrt(E*E - px*px - py*py - pz*pz);
        return m;
    }

    double gamma()
    {
        return E/m;
    }

    double beta()
    {
        B = p/E;
        return B;
    }

    double radius()
    {
        return p/field;
    }
}
```

Task 3.1

The following piece of code defines the four vector values, asks the user to input the magnetic field, and prints out the values calculated in the 'FourVector' class. The output is shown underneath.

```java
//Task 3.1: Four-vector calculations
package RelativisticKinematics;

import java.util.Scanner;

public class relativistickinematics {
    static Scanner keyboard = new Scanner(System.in);


    public static void main(String[] args) {
        FourVector particle = new FourVector();
        particle.E = 247.5;
        particle.px = 100.;
        particle.py = 0.0;
        particle.pz = 200.;
        //take input magnetic field:
        System.out.println("Please type in the magnetic field in T ");
        particle.field = keyboard.nextDouble(); //strength of magnetic field in T

        double c = 3E8; //speed of light in m/s
        double t = 2.19E-6; // lifetime of particle in s

        System.out.println("Particle with fourvector (E,px,py,pz) = ("
                + particle.E + ", "
                + particle.px + ", "
                + particle.py + ", "
                + particle.pz + ") MeV in B = " + particle.field + " T.");


        //take input magnetic field here
        double B = 300; //strength of magnetic field

        System.out.println("Momentum = " + particle.momentum() + " MeV");

        System.out.println("Mass = " + particle.mass() + " MeV");

        System.out.println("Lorentz gamma factor = " + particle.gamma());

        double speed = particle.beta()*c; // calculating the speed in m/s
        System.out.println("v = " + speed + " m/s");

        System.out.println("Radius of curvature in a magnetic field = " + particle.
radius());
```

```

        double distance = speed*t*particle.gamma(); //calculating the distance travelled by the particle
        // the particle sees proper time. We see (proper time) * gamma, so the distance travelled is this times the speed
        System.out.println(" Distance travelled = " + distance + " m.");
    }

}
```

```
run:
Please type in the magnetic field in T
1.8
Particle with fourvector (E,px,py,pz) = (247.5, 100.0, 0.0, 200.0) MeV in B = 1.8 T.
Momentum = 223.60679774997897 MeV
Mass = 106.09547586961472 MeV
Lorentz gamma factor = 2.332804466650052
v = 2.7103854272724724E8 m/s
Radius of curvature in a magnetic field = 124.2259987499883
 Distance travelled = 1384.6930316074909 m.
BUILD SUCCESSFUL (total time: 4 seconds)
```

<u>Task 3.2</u>

This piece of code asks the user to input E, p_x, p_y and p_z for two particles. It then prints their four vector, momentum and mass. Finally, it adds the two four vectors, calculates that particle's mass and momentum, and prints the results. The program was tested in two cases:

- Case 1: four-vector 1 = (5, 4, 0, 0) and four-vector 2 = (5, -4, 0, 0)
- Case 2: four-vector 1 = (60000, 60000, 0, 0) and four-vector 2 = (92531, -40000, 45400, 70000)

The output in each case is shown beneath the code.

```java
//Task 3.2: Four-vector calculations
package task32;

import java.util.Scanner;

public class relativistickinematics2 {
    static Scanner keyboard = new Scanner(System.in);


    public static void main(String[] args) {
        FourVector2 particle1 = new FourVector2(); //FourVector2 is the same as FourVector, I just made a new folder and gave it a new name for this exercise
        //particle 1 is the first particle

        //the user is asked to input the energy and the x, y and z components of the momentum individually:
        System.out.println("Type in E of particle 1 in MeV");
        particle1.E = keyboard.nextDouble(); //each value is stored as the variables defined in FourVector2

        System.out.println("Type in px of particle 1 in MeV");
        particle1.px = keyboard.nextDouble();

        System.out.println("Type in py of particle 1 in MeV");
        particle1.py = keyboard.nextDouble();

        System.out.println("Type in pz of particle 1 in MeV");
        particle1.pz = keyboard.nextDouble();

        //Printing the fourvector, momentum and mass (the latter two are calculated
 in Fourvector2):
        System.out.println("Particle 1 has fourvector (E,px,py,pz) = ("
                + particle1.E + ", "
                + particle1.px + ", "
                + particle1.py + ", "
                + particle1.pz + ") MeV, momentum = " + particle1.momentum() + " Me
V and mass = " + particle1.mass() + " MeV");


        //The same is done for the second particle, particle2:
        FourVector2 particle2 = new FourVector2();

        System.out.println("Type in E of particle 2 in MeV");
        particle2.E = keyboard.nextDouble();

        System.out.println("Type in px of particle 2 in MeV");
        particle2.px = keyboard.nextDouble();

        System.out.println("Type in py of particle 2 in MeV");
        particle2.py = keyboard.nextDouble();

        System.out.println("Type in pz of particle 2 in MeV");
        particle2.pz = keyboard.nextDouble();

        System.out.println("Particle 2 has fourvector (E,px,py,pz) = ("
                + particle2.E + ", "
                + particle2.px + ", "
                + particle2.py + ", "
                + particle2.pz + ") MeV, momentum = " + particle2.momentum() + " Me
V and mass = " + particle2.mass() + " MeV");


        //The fourvector of the third paticle is calculated by adding the fourvecto
rs of particle1 and particle2:
        FourVector2 particle3 = new FourVector2();

        particle3.E = particle1.E + particle2.E;
        particle3.px = particle1.px + particle2.px;
        particle3.py = particle1.py + particle2.py;
        particle3.pz = particle1.pz + particle2.pz;

        System.out.println("Particle 3 (combined) has fourvector (E,px,py,pz) = ("

                + particle3.E + ", "
                + particle3.px + ", "
                + particle3.py + ", "
                + particle3.pz + ") MeV, momentum = " + particle3.momentum() + " Me
V and mass = " + particle3.mass() + " MeV");
    }

}
```

Case 1

```
run:
Type in E of particle 1 in MeV
5
```

```
Type in px of particle 1 in MeV
4
Type in py of particle 1 in MeV
0
Type in pz of particle 1 in MeV
0
Particle 1 has fourvector (E,px,py,pz) = (5.0, 4.0, 0.0, 0.0) MeV, momentum = 4.0 MeV and
mass = 3.0 MeV
Type in E of particle 2 in MeV
5
Type in px of particle 2 in MeV
-4
Type in py of particle 2 in MeV
0
Type in pz of particle 2 in MeV
0
Particle 2 has fourvector (E,px,py,pz) = (5.0, -4.0, 0.0, 0.0) MeV, momentum = 4.0 MeV and
mass = 3.0 MeV
Particle 3 (combined) has fourvector (E,px,py,pz) = (10.0, 0.0, 0.0, 0.0) MeV, momentum =
0.0 MeV and mass = 10.0 MeV
BUILD SUCCESSFUL (total time: 33 seconds)
```

<u>Case 2</u>

```
run:
Type in E of particle 1 in MeV
60000
Type in px of particle 1 in MeV
60000
Type in py of particle 1 in MeV
0
Type in pz of particle 1 in MeV
0
Particle 1 has fourvector (E,px,py,pz) = (60000.0, 60000.0, 0.0, 0.0) MeV, momentum =
60000.0 MeV and mass = 0.0 MeV
Type in E of particle 2 in MeV
92531
Type in px of particle 2 in MeV
-40000
Type in py of particle 2 in MeV
45400
Type in pz of particle 2 in MeV
70000
Particle 2 has fourvector (E,px,py,pz) = (92531.0, -40000.0, 45400.0, 70000.0) MeV,
momentum = 92526.53673406348 MeV and mass = 908.823965353027 MeV
Particle 3 (combined) has fourvector (E,px,py,pz) = (152531.0, 20000.0, 45400.0, 70000.0)
MeV, momentum = 85797.2027516049 MeV and mass = 126113.22674882282 MeV
BUILD SUCCESSFUL (total time: 53 seconds)
```