

# Task Sheet

## General Instructions

You will have the afternoon to complete these tasks.

Create a new project for all of the following tasks. As a suggestion, you can call your main project “tasks” but you can call it whatever suits you.

Each task will go into its own separate package and will be run individually. Each task will require its own `main` method. This would mean the following structure with each `App.java` file having its own `main`.

```
| - com.sparta
|   |
|   | - day1
|   |   |
|   |   | - debug
|   |   |   |
|   |   |   | - DebugApp.java
|   |   |
|   |   | - calculator
|   |   |   |
|   |   |   | - CalculatorApp.java
|   |
|   | - day2
```

For all tasks you should be able to explain and justify the solution that you arrive at.

# Tasks

## Debugging Code

---

Package: com.sparta.day1.debug

Class: DebugApp

```
public class DebugApp {  
    public static void main(String[] args) {  
        var name;  
        name = "Josh Crabtree"  
        occupation = "trainee";  
        String location = 'London';  
        var activity level = "moderate";  
  
        System.out.println("My name is", name);  
        System.out.printf("I am a %s in %s\n" + location, occupation));  
  
        int age == 21;  
  
        print("I am " + age + " years of age with a ", activity level, "  
activity level";  
    }  
}
```

Copy and paste the above code into `App.java`. You will see that it contains a lot of bugs which you will need to correct.

Correct the code so that when it runs, the resulting output is as follows.

```
My name is Josh Crabtree  
I am a trainee in London  
I am 21 years of age with a moderate activity level
```

## Simple Calculator

---

Package: com.sparta.day1.calculator

Class: CalculatorApp

Write a program in `CalculatorApp.main` that has two hard-coded integer values.

Your program should divide the first integer by the second integer and then output the result to the console.

## Core Java

For example, if you hard-code the integers 15 and 10, the result should be 1.5.

### Breakeven

---

Package: com.sparta.day1.breakeven

Class: BreakEvenApp

If an item costs £50 to produce and is then sold at £100, it makes a profit of £50 per item.

Given fixed costs of £1000, the produce will break even when the total profit made is equal to the fixed costs. For fixed costs of £1000, twenty items would need to be sold at £50 profit per item.

Write a program that works out the breakeven point using the following data:

- Cost to produce each item = £20.00
- The sale price per item (item cost + profit per item) = £40.00
- Fixed costs = £50000.00

Your program should generate the following output. Try to match it exactly.

```
Sale price for each item: 40.0
Fixed costs: 50000.0
Profit per item: 20.0
Breakeven: 2500
```

### Basic Data Types

---

In a text document called datatypes.txt, write down the most appropriate data type for the following data with a brief justification for your decision:

- a) A company's profit
- b) A person's weight
- c) A book title
- d) Average rainfall in September
- e) Number of runs scored by a cricketer
- f) A telephone number, including area code
- g) A dress size

## Sweet Tooth

---

Package: com.sparta.day1.sweettooth

Class: SweetToothApp

A teacher has bought a packet of 40 sweets that she is going to share out equally between her 14 students. Because a single sweet cannot be shared, the teacher will keep what is not given to the children.

The teacher will keep the minimum number of sweets possible.

Write a program to determine and output the number of sweets each child will receive. As a single sweet cannot be shared between pupils, the program must calculate and output the number of sweets per child and the number that the teacher keeps for herself.

The calculation must be done using the modulus (%) operator.

## Cans of Paint

---

Package: com.sparta.day1.paint

Class: PaintApp

Write a program that uses methods from the [`Math` class](#) to solve the following problem.

A can of paint covers 5.1 m<sup>2</sup> of wall. Each can of paint has a diameter of 15 cm and a height of 30 cm. The shop that sells the paint helps customers by packing the cans into boxes with internal measurements of 0.60 x 0.30 x 0.35 metres (L x W x H).

Solve and output the following:

1. The minimum number of cans that must be bought to paint the walls of a hall whose floor measures 40 x 30 metres, and whose ceiling is 3.4 metres above the floor.
2. The number of full boxes given to the customer who buys this quantity of paint
3. The number of cans not packed into boxes.

Your output should be:

```
Minimum number of cans required: 94
Number of full boxes: 11
Number of cans not packed: 6
```

## Generate a JSON String

---

Package: com.sparta.day1.jsongenerator

Class: JsonGeneratorApp

Given the following data:

- First name: Alex
- Last name: Chen
- Email: [alex.chen@example.com](mailto:alex.chen@example.com)
- Phone: +44 20 7946 0832
- Occupation: Software Engineer
- Is Active: true
- Years of Service: 4

Create a program that accepts this data as arguments when you run it and outputs the results in correctly formatted JSON.

While it isn't necessary to understand JSON, you can get more information on the notation here: [https://www.w3schools.com/js/js\\_json\\_datatypes.asp](https://www.w3schools.com/js/js_json_datatypes.asp). Although there are numerous other sources of information on Json available.

Your output should look 'pretty' so that it can be easily read. It should look like this:

```
{
  "first_name": "Alex",
  "last_name": "Chen",
  "email": "alex.chen@example.com",
  "phone": "+44 20 79746 0832",
  "occupation": "Software Engineer",
  "is_active": true,
  "years_service": 4
}
```