Survival Camp Simulation

Estimated time: <1 hour
Goal: Practice core programming concepts such as enums, arrays, HashMaps, loops, operators, and basic object-oriented programming through designing a small survival camp simulation using Java.
Actively participate in this survival camp creation to get 1 point

Background Story

Last week, you designed and created a magical artifact in the Realm of Enchantments. However, during a portal experiment, something went wrong. You got teleported into a dangerous valley where infected creatures exist and they are now approaching from the surrounding areas. Fortunately, your magical artifact also survived the portal jump.

You have found a safe location and must now build a survival camp. There are also other survivors who got transported into this dangerous territory. While some of your peers are searching for an answer to bring you guys back, your mission here is to place survivors, tents, fires, water sources and your magical artifact on a grid-based map and make sure you survive. If your camp is strong enough, you will survive the night or even a week.

Learning Goals:

1) Use enums to represent survival objects or item types
2) Use 2D arrays to model a grid-based map
3) Use HashMap to count and store inventory data
4) Apply loops to traverse arrays and collections
5) Use operators and conditional statements to calculate scores
6) Be able to apply this knowledge for week 3 exercise

Phase 1: Planning and Design Phase (5-10 minutes)

Before you go straight into coding, think and plan how you will build your survival camp.

1) Identify game objects
   - What types of things exist in your camp? Plan your items such as TENT, SURVIVOR, FIRE, WATER

2) Camp map design
   - What grid size will you use? (Minimum 5x5 but of course, you can make a larger camp)
   - What data type will each cell store?

3) Inventory Tracking (HashMap planning)
   - How will you count objects?

4) Score Logic (Operators Planning)
   - How will you design score rules?
   - What operators will you use to combine scores? (Multiplication? Addition?)

5) Program Flow
   - How are you going to execute this program?
   - What are the steps involved?

Phase 2: Coding Phase (30 - 60 minutes)

Basic Requirements

You need to create **THREE** separate files:

- CampMap.java (Grid and map handling)
- Item.java (Enum for camp items)
- Main.java (Program entry point)

File 1: CampMap.java (Your Class)

Purpose: Represent your camp and provide methods to interact with it.

Your class must include:

1. Attributes (at least 3) that describe your camp or grid:
   - Example attributes: size (int), grid (Item [ ] [ ]), campName (String)
   - Use private access modifiers
   - Choose appropriate data types

2. Constructor that initializes all attributes:
   - Takes parameters for at least 3 attributes
   - Assigns parameter values to instance variables

```
5       public CampMap(String campName, int size) {
6       this.campName = campName;
7       this.size = size;
8       this.grid = new Item[size][size];
9       // optionally fill with EMPTY items
10  }
```

3. Methods (at least 2) that make your camp “do something”:
   Example:
   - placeItem(int row, int col, Item item)
   - displayCamp()
   - calculateScore()
   Methods should print descriptive messages to show what they do.

4. Comments: Include brief comments at the top of your class and above each method.

File 2: Item.java (Enum)

Purpose: Define all possible objects/items that can exist in your camp.

1. Create an enum named Item (or whichever you prefer).
2. Include at least 5 item types:
   Examples:
   - EMPTY
   - TENT
   - SURVIVOR
   - ALIEN
   - FIRE
   - MAGIC

3. Assign a symbol to each item (optional, of course, for visualization but you can also use alphabets if you want)

```java
public enum Item {
    EMPTY(symbol: "🌳"),
    ITEM1(symbol: "🏕️"),
    ITEM2(symbol: "🪵"),
    ITEM3(symbol: "🔥"),
    ITEM4(symbol: "💧"),
    ITEM5(symbol: "⚔️");

    public String symbol;

    // constructor
    Item(String symbol) {
        this.symbol = symbol;
    }
}
```

Note: You need to have specific fonts to print out emojis in the terminal. You may use letters in case emojis don't work.

File 3: App.java (Main Program)

Purpose: Run your camp simulation.

Your App.java file must include:

1. A main method that:
   - Creates at least one instance of your **CampMap** class
   - Calls all the action methods to demonstrate functionality
   - Shows your camp on the console in a clear, readable way

2. Comments: Explain what each block of code does

Bonus Ideas

Similar to week 2, if you're a programming god and potentially wanna show off your coding skills, feel free to do the following:

1) Add more attributes (number of survivors, fire status, water supply, or whatever quirky stuff your brain can imagine)
2) Add more methods (the more the merrier) to upgrade your survival camp
3) Change grid size dynamically
4) Add random placement of items
5) Include survival message based on the score

Note: Unfortunately, you'll not get additional points for doing these bonus ideas but hey, at least (hopefully) your thirst for intellect and creation will be satisfied.

Example Basic Code Structure

CampMap.java

```java
package main;

public class CampMap {
    /*
    TODO: Add at least 3 private attributes (e.g., size, campName, grid)
    private datatype attributeName;
    Hint: grid/map should be Item[][]
    private Item[][] grid;
    */

    // TODO: Create a constructor that initializes all attributes
    // Hint: call a method to fill the grid with EMPTY items

    // TODO: Method to fill grid with EMPTY items
    // Hint: use nested loops to set each cell to Item.EMPTY

    // TODO: Method to place an item at a given row and column
    // Hint: update grid[row][col] and print a message

    // TODO: Method to display the camp grid
    // Hint: use nested loops and Item.symbol

    // TODO: Getter for grid (needed for counting items)
    // Hint: return grid;
}
```

App.java

```java
package main;

import java.util.HashMap;

public class App {
    Run | Debug
    public static void main(String[] args) {

        // TODO: Create an instance of CampMap
        // Hint: CampMap instanceName = new CampMap("CampName", size);
        System.out.println(x: " Welcome to your survival camp!");

        // TODO: Place at least one tent, two survivors, one fire, one magic item, two water sources
        // Hint: camp.placeItem(row, col, Item.TENT);

        // TODO: Display the camp
        // Hint: camp.displayCamp();

        // TODO: Use a HashMap to count each item in the grid
        // Hint: initialize count to 0 for each Item, then loop through grid and increment

        // TODO: Display inventory counts
        // Hint: print each Item and its count

        /*
            TODO: Calculate and display camp score
            Hint: multiply each count by points assigned per item and sum
            Example:
            int score = 0;
            score += inventory.get(Item.TENT) * 5;
        */

        // TODO: Conditional message depending on score (survive or need more resources)
        // Hint: if (score >= threshold) { ... else { ... }

    }
}
```