

# Cooking Companion(---RETIRED---)

**Grade settings:** Maximum grade: 100

**Disable external file upload, paste and drop external content:** Yes

**Based on:** [Cooking Companion\(---RETIRED---\)](#)

**Run:** Yes **Evaluate:** Yes

**Automatic grade:** Yes

Alice loves to cook and experiment with new recipes. She has a collection of recipe books with hundreds of recipes, but she struggles to decide what to cook based on the amount of time she has available. Alice wanted to organize her recipes and search for dishes based on their preparation time.

You being the software developer, develop a Java program based on the requirement.

## Component Specification: CookBookMain Class

Type (Class)	Attributes	Methods
<b>CookBookMain</b>	private Map<String, Integer> <b>recipeMap</b>	Getter and setter methods for the attribute are included in the code skeleton.

**Note:** Here the recipe, holds the Key as recipeName and Value as preparationTime( in minutes).

**Requirement 1: Count the number of recipeNames based on the given preparationTime .**

Type (Class)	Methods	Responsibilities
<b>CookBookMain</b>	public int <b>countRecipesUnderTime</b> (int preparationTime )	This method accepts <b>preparationTime</b> as an argument. If the <b>preparationTime</b> matches the <b>preparationTime</b> present in the Map, it must count the recipe's and return the count.  <b>Condition:</b> <ul style="list-style-type: none"><li>• <i>If preparationTime is n minutes is less</i></li></ul>

		<i>than or equal to zero, return -1.</i>
--	--	--

**Requirement 2: Filter the recipeName based on the preparationTime.**

Type (Class)	Methods	Responsibilities
<b>CookBookMain</b>	public List<String> <b>getRecipesUnderTime</b> (int preparationTime )	This method accepts <b>preparationTime</b> as parameter and filters the <b>recipeName</b> and returns the list of <b>preparationTime</b> that are less than or equal to the given <b>preparationTime</b> .

**You are provided with the main method as code template and it is excluded from evaluation.**

**Note:**

- In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user, and the rest of the text represents the output.
- Ensure to follow the object-oriented specifications provided in the question description.
- Ensure to provide the names for the classes, attributes, and methods as specified in the question description.
- Adhere to the code template, if provided.

**Sample Input / Output 1**

Enter number of recipes to be added

**10**

Enter the recipe (Recipe name : Preparation time)

**LemonGarlicSalmon:25**

**CrispyChickenTenders:30**

**BeefStroganoff:45**

**VegetableStirFry:20**

**ButternutSquashSoup:40**

**SpicyShrimpPasta:35**

**ChickenPotPie:50**

**TeriyakiSalmon:20**

**BroccoliCheddarSoup:35**

**VegetableLasagna:60**

Enter the Preparation time to be searched

**20**

The Recipes with preparation time less than 20 minutes are 2

Enter the Preparation time to identify the Recipe Names

**35**

Recipes with preparation time less than 35 minutes are

SpicyShrimpPasta

TeriyakiSalmon

LemonGarlicSalmon

CrispyChickenTenders

VegetableStirFry

BroccoliCheddarSoup

## **Sample Input / Output 2**

Enter number of recipes to be added

**4**

Enter the recipe (Recipe name : Preparation time)

**CreamyMushroomPasta:50**

**BakedPotatoes:40**

**MediterraneanQuinoaBowl:30**

**StuffedBellPeppers:60**

Enter the Preparation time to be searched

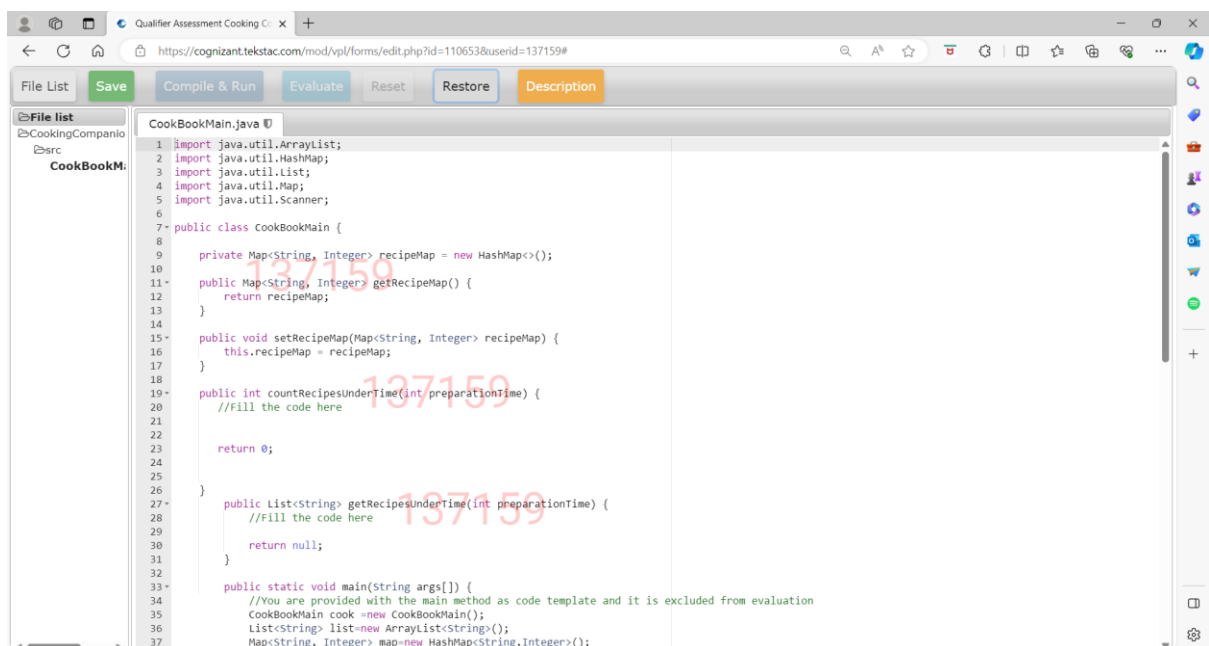
**10**

No recipes were found with preparation time less than 10 minutes

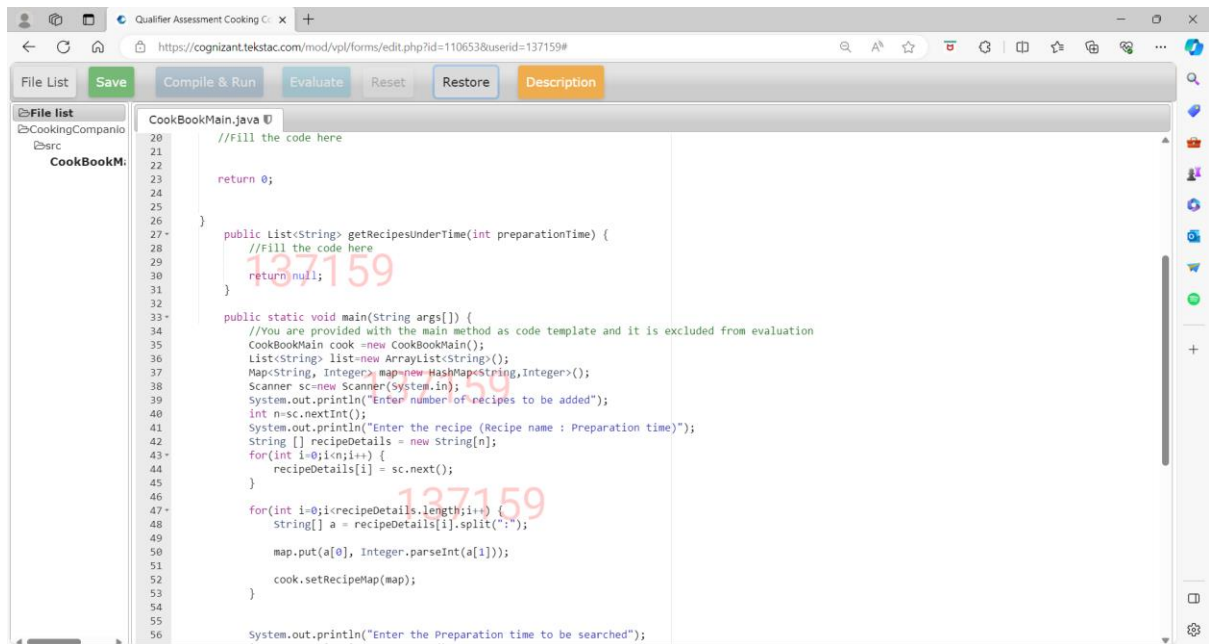
Enter the Preparation time to identify the Recipe Names

**20**

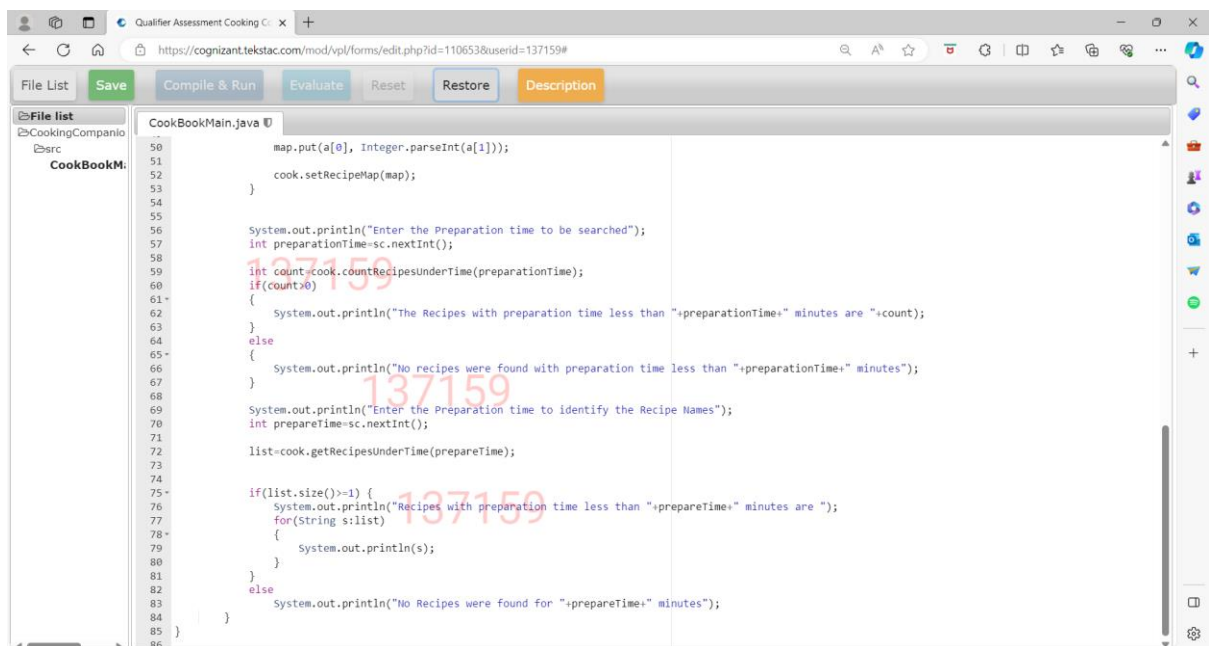
No Recipes were found for 20 minutes



```
1 import java.util.ArrayList;
2 import java.util.HashMap;
3 import java.util.List;
4 import java.util.Map;
5 import java.util.Scanner;
6
7 public class CookBookMain {
8
9     private Map<String, Integer> recipeMap = new HashMap<>();
10
11     public Map<String, Integer> getRecipeMap() {
12         return recipeMap;
13     }
14
15     public void setRecipeMap(Map<String, Integer> recipeMap) {
16         this.recipeMap = recipeMap;
17     }
18
19     public int countRecipesUnderTime(int preparationTime) {
20         //Fill the code here
21
22         return 0;
23     }
24
25     public List<String> getRecipesUnderTime(int preparationTime) {
26         //Fill the code here
27         return null;
28     }
29
30     public static void main(String args[]) {
31         //You are provided with the main method as code template and it is excluded from evaluation
32         CookBookMain cook = new CookBookMain();
33         List<String> list = new ArrayList<>();
34         Map<String, Integer> map = new HashMap<>();
35     }
```



```
20 //Fill the code here
21
22
23 return 0;
24
25 }
26
27 public List<String> getRecipesUnderTime(int preparationTime) {
28     //Fill the code here
29     return null;
30 }
31
32
33 public static void main(String args[]) {
34     //You are provided with the main method as code template and it is excluded from evaluation
35     CookbookMain cook =new CookbookMain();
36     List<String> list=new ArrayList<String>();
37     Map<String, Integer> map=new HashMap<String,Integer>();
38     Scanner sc=new Scanner(System.in);
39     System.out.println("Enter number of recipes to be added");
40     int n=sc.nextInt();
41     System.out.println("Enter the recipe (Recipe name : Preparation time)");
42     String [] recipeDetails = new String[n];
43     for(int i=0;i<n;i++) {
44         recipeDetails[i] = sc.next();
45     }
46
47     for(int i=0;i<recipeDetails.length;i++) {
48         String[] a = recipeDetails[i].split(":");
49
50         map.put(a[0], Integer.parseInt(a[1]));
51     }
52     cook.setRecipeMap(map);
53
54
55
56     System.out.println("Enter the Preparation time to be searched");
```



```
50     map.put(a[0], Integer.parseInt(a[1]));
51
52     cook.setRecipeMap(map);
53
54
55
56     System.out.println("Enter the Preparation time to be searched");
57     int preparationTime=sc.nextInt();
58     int count=cook.countRecipesUnderTime(preparationTime);
59     if(count>0)
60     {
61         System.out.println("The Recipes with preparation time less than "+preparationTime+" minutes are "+count);
62     }
63     else
64     {
65         System.out.println("No recipes were found with preparation time less than "+preparationTime+" minutes");
66     }
67
68     System.out.println("Enter the Preparation time to identify the Recipe Names");
69     int prepareTime=sc.nextInt();
70
71     list=cook.getRecipesUnderTime(prepareTime);
72
73
74
75     if(list.size()>1) {
76         System.out.println("Recipes with preparation time less than "+prepareTime+" minutes are ");
77         for(String s:list)
78         {
79             System.out.println(s);
80         }
81     }
82     else
83     {
84         System.out.println("No Recipes were found for "+prepareTime+" minutes");
85     }
86 }
```