



Screenshot of a Gmail inbox showing an email from 'Recruitment' regarding a 'NEXT Interview Assignment Task'.

Email Details:
Subject: NEXT Interview Assignment Task
From: Recruitment <recruitment@hashagile.com>
To: Recruitment
Sent: 11:17 AM (10 hours ago)
Attachments: None

Email Content:

Dear Candidate,

Greetings!

We appreciate your participation and continued effort in completing the online test!

Due to the overwhelming response, we received, we are looking for one more online test with an enhanced set of questions to help achieve the final milestone. Your participation in this test is crucial, and based on your program output and the results, you will be invited for an in-person technical interview at our office in Coimbatore Location.

Please stick to your previous assignment and complete the next assignment with the following details.

- **Function Definitions**
1. `createCollection(p_collection_name)`

System Status Bar:
Type here to search | 9:34 PM | 29°C Haze | 03/10/2024 |

The screenshot shows a browser window with the URL programiz.com/java-programming/online-compiler/. The page features the Programiz logo and a banner for "Premium Coding Courses by Programiz". On the left, there's a file selector with "Main.java" selected. The code editor contains the following Java code:

```
1- import java.util.*;
2 import java.util.stream.Collectors;
3
4- class Employee {
5     String id;
6     String name;
7     String department;
8     String gender;
9
10-    public Employee(String id, String name, String
11                     department, String gender) {
12        this.id = id;
13        this.name = name;
14        this.department = department;
15        this.gender = gender;
16    }
17
18- class CollectionManager {
19     Map<String, List<Employee>> collections = new HashMap
```

The screenshot shows a browser window with the URL programiz.com/java-programming/online-compiler/. The page features a promotional banner for "Born between 1956 to 1996? You can..." with "No previous experience needed" and "Sponsored by: Deal-Peat". On the left, there's a file selector with "Main.java" selected. The code editor contains the following Java code:

```
17
18- class CollectionManager {
19     Map<String, List<Employee>> collections = new HashMap
20         <>();
21
22-     public void createCollection(String p_collection_name) {
23         collections.put(p_collection_name, new ArrayList
24             <>());
25         System.out.println("Created collection: " +
26                           p_collection_name);
27     }
28-     public void indexData(String p_collection_name, String
29         p_exclude_column) {
30         List<Employee> employees = collections.get
31             (p_collection_name);
32
33         if (p_exclude_column.equalsIgnoreCase("Department"))
34         {
35             for (Employee emp : employees) {
36                 emp.department = null;
```

The screenshot shows a web browser window with the URL programiz.com/java-programming/online-compiler/. The page features a header with the Programiz logo, a Bitsler advertisement for a 200% bonus up to \$2,000, and a "Programiz PRO" button. The main area contains a code editor with Java code and an output panel.

```
30     emp.department = null;
31   }
32   } else if (p_exclude_column.equalsIgnoreCase
("Gender")) {
33     for (Employee emp : employees) {
34       emp.gender = null;
35     }
36   }
37   System.out.println("Indexed data into collection: "
+ p_collection_name + " excluding column: " +
p_exclude_column);
38 }
39 public List<Employee> searchByColumn(String
p_collection_name, String p_column_name, String
p_column_value) {
40   List<Employee> employees = collections.get
(p_collection_name);
41   List<Employee> result = new ArrayList<>();
42
43   if (p_column_name.equalsIgnoreCase("Department")) {
44     result = employees.stream().filter(emp ->

```

The screenshot shows a web browser window with the URL programiz.com/java-programming/online-compiler/. The page features a header with the Programiz logo, a Bitsler advertisement for a 200% bonus up to \$2,000, and a "Programiz PRO" button. The main area contains a code editor with Java code and an output panel.

```
43   if (p_column_name.equalsIgnoreCase("Department")) {
44     result = employees.stream().filter(emp ->
p_column_value.equals(emp.department
)).collect(Collectors.toList());
45   } else if (p_column_name.equalsIgnoreCase("Gender"))
{
46     result = employees.stream().filter(emp ->
p_column_value.equals(emp.gender)).collect
(Collectors.toList());
47   }
48   return result;
49 }
50 public int getEmpCount(String p_collection_name) {
51   return collections.get(p_collection_name).size();
52 }
53 public void delEmpById(String p_collection_name, String
p_employee_id) {
54   List<Employee> employees = collections.get
(p_collection_name);
55   employees.removeIf(emp -> emp.id.equals

```

The screenshot shows a browser window with the URL programiz.com/java-programming/online-compiler/. The page features a 'WELCOME BONUS' banner for Bitsler with a 200% up to \$2,000 + 500 FREE SPINS offer. The main content is a Java code editor for 'Main.java'. The code implements methods to filter employees by department or gender, get employee count, and delete an employee by ID. The code editor includes syntax highlighting and line numbers. To the right is an 'Output' panel and a 'Run' button. The bottom of the screen shows a Windows taskbar with various pinned icons.

```
if (p_column_name.equalsIgnoreCase("Department")) {  
    result = employees.stream().filter(emp ->  
        p_column_value.equals(emp.department  
    )).collect(Collectors.toList());  
}  
else if (p_column_name.equalsIgnoreCase("Gender"))  
{  
    result = employees.stream().filter(emp ->  
        p_column_value.equals(emp.gender)).collect  
(Collectors.toList());  
}  
return result;  
}  
public int getEmpCount(String p_collection_name) {  
    return collections.get(p_collection_name).size();  
}  
public void delEmpById(String p_collection_name, String  
p_employee_id) {  
    List<Employee> employees = collections.get  
(p_collection_name);  
    employees.removeIf(emp -> emp.id.equals
```

The screenshot shows a browser window with the same URL as the previous one. A promotional message about earning extra income with companies is displayed in a floating box. The code editor contains Java code for an 'EmployeeCollection' class. It defines a main method that initializes a 'CollectionManager' and creates two collections: 'v_nameCollection' and 'v_phoneCollection'. It then adds four employees to each collection. The code editor includes syntax highlighting and line numbers. To the right is an 'Output' panel and a 'Run' button. The bottom of the screen shows a Windows taskbar with various pinned icons.

```
public class EmployeeCollection {  
    public static void main(String[] args) {  
        CollectionManager manager = new CollectionManager();  
        String v_nameCollection = "Hash_AnushruthiK";  
        String v_phoneCollection = "Hash_7883";  
        manager.createCollection(v_nameCollection);  
        manager.createCollection(v_phoneCollection);  
        manager.collections.get(v_nameCollection).add(new  
            Employee("E02001", "Aravind", "IT", "Male"));  
        manager.collections.get(v_nameCollection).add(new  
            Employee("E02002", "Jaya", "HR", "Female"));  
        manager.collections.get(v_nameCollection).add(new  
            Employee("E02003", "Dhanush", "IT", "Male"));  
        manager.collections.get(v_phoneCollection).add(new  
            Employee("E02004", "Aswin", "Sales", "Male"));  
        manager.collections.get(v_phoneCollection).add(new
```

Screenshot of the Programiz Online Java Compiler interface. The code editor shows Java code for managing employee collections. The code includes printing employee counts, indexing data, deleting an employee by ID, and searching for IT employees. The output panel is currently empty.

```
81     System.out.println("Employee Count (Name Collection ): " + manager.getEmpCount(v_nameCollection));
82
83     manager.indexData(v_nameCollection, "Department");
84     manager.indexData(v_phoneCollection, "Gender");
85
86     manager.delEmpById(v_nameCollection, "E02003");
87
88     System.out.println("Employee Count (Name Collection )
89     after deletion: " + manager.getEmpCount
89     (v_nameCollection));
90
91
92     List<Employee> itEmployees = manager.searchByColumn
92     (v_nameCollection, "Department", "IT");
93     System.out.println("IT Employees (Name Collection):
93     " + itEmployees.size());
94
95     List<Employee> maleEmployees = manager
```

Screenshot of the Programiz Online Java Compiler interface. The code editor shows Java code for managing employee collections, similar to the first screenshot but with additional code for printing male employee counts and searching for IT employees. The output panel is currently empty.

```
96     System.out.println("Male Employees (Name Collection
96     ): " + maleEmployees.size());
97
98     List<Employee> itPhoneEmployees = manager
98     .searchByColumn(v_phoneCollection, "Department",
98     "IT");
99     System.out.println("IT Employees (Phone Collection):
99     " + itPhoneEmployees.size());
100
101    Map<String, Long> nameCollectionFacet = manager
101    .getDepFacet(v_nameCollection);
102    System.out.println("Department Facet (Name
102    Collection): " + nameCollectionFacet);
103
104    Map<String, Long> phoneCollectionFacet = manager
104    .getDepFacet(v_phoneCollection);
105    System.out.println("Department Facet (Phone
105    Collection): " + phoneCollectionFacet);
106  }
107 }
```

The screenshot shows a Java application running on the Programiz Online Java Compiler. The code in Main.java prints statistics about employee collections. The output window shows the execution command, collection creation, employee counts, indexed data, deleted employees, and a successful execution message.

```
Main.java
97
98     List<Employee> itPhoneEmployees = manager
99         .searchByColumn(v_phoneCollection, "Department",
100             "IT");
101
102     System.out.println("IT Employees (Phone Collection):
103         " + itPhoneEmployees.size());
104
105     Map<String, Long> nameCollectionFacet = manager
106         .getDepFacet(v_nameCollection);
107
108     System.out.println("Department Facet (Name
109         Collection): " + nameCollectionFacet);
110
111     Map<String, Long> phoneCollectionFacet = manager
112         .getDepFacet(v_phoneCollection);
113
114     System.out.println("Department Facet (Phone
115         Collection): " + phoneCollectionFacet);
116
117 }
```

```
java -cp /tmp/g9nmWzGwgm/EmployeeCollection
Created collection: Hash_AnushruthiK
Created collection: Hash_7883
Employee Count (Name Collection): 3
Indexed data into collection: Hash_AnushruthiK excluding column:
    Department
Indexed data into collection: Hash_7883 excluding column: Gender
Deleted employee with ID: E02003
Employee Count (Name Collection) after deletion: 2
IT Employees (Name Collection): 0
Male Employees (Name Collection): 1
IT Employees (Phone Collection): 0
Department Facet (Name Collection): {}
Department Facet (Phone Collection): {Sales=1, HR=1}

*** Code Execution Successful ***
```