Lösningsförslag, Programmeringsteknik/Programmering

Moodle-tenta, 2019–06–03

1. Totalt **5p**

```
public class Employee {
    private static int nextCardNbr = 1;

    private String name;
    private int cardNbr;

    public Employee(String name) {
        this.name = name;
        this.cardNbr = nextCardNbr;
        nextCardNbr++;
    }

    public int getEmployeeNbr() {
        return cardNbr;
    }

    public String getName() {
        return name;
    }

    public String toString() {
        return name + " (" + cardNbr + ")";
    }
}
```

2. Totalt **17p**

```
public class Tracker {
    private ArrayList<Employee> employees;
    private ArrayList<Employee> mine;

    public Tracker() {
        employees = new ArrayList<Employee>();
        mine = new ArrayList<Employee>();
    }

    public int addEmployee(String name) {
        Employee p = new Employee(name);
        employees.add(p);
        return p.getEmployeeNbr();
    }

    public boolean enterMine(int employeeNbr) {
        Employee p = findEmployee(employeeNbr);
        if (p != null) {
            mine.add(p);
            return true;
        } else {
            return false;
        }
    }

    void removeEmployee(int employeeNbr) {
        Employee p = findEmployee(employeeNbr);
        // det är okej att anta att p != null här
        employees.remove(p);
    }

    public void exitMine(int employeeNbr) {
        Employee p = findEmployee(employeeNbr);
        // det är okej att anta att p != null här
        mine.remove(p);
    }

    public ArrayList<Employee> getPersonsInMine() {

        // ok att sortera den interna listan
        for (int i = 0; i < mine.size() - 1; i++) {
            for (int k = i + 1; k < mine.size(); k++) {
                Employee pi = mine.get(i);
                Employee pk = mine.get(k);
                if (pk.getName().compareTo(pi.getName()) < 0) {
                    mine.set(i, pk);
                    mine.set(k, pi);
                }
            }
        }
        return mine;
    }

    private Employee findEmployee(int employeeNbr) {
        for (Employee p : employees) {
            if (p.getEmployeeNbr() == employeeNbr) {
                return p;
            }
        }
        return null;
    }
```

```
}
```

Alternativ implementation av `enterMine` och `getPersonsInMine` (ger samma totalpoäng):

```
public boolean enterMine(int employeeNbr) {
    Employee p = findEmployee(employeeNbr);
    if (p != null) {
        int pos = 0;
        while (pos < mine.size() && mine.get(pos).getName().compareTo(p.getName()) < 0) {
            pos++;
        }
        mine.add(pos, p);
        return true;
    } else {
        return false;
    }
}

public ArrayList<Employee> getPersonsInMine() {
    return mine;
}
```

3. Totalt **5p**

```
public static int compare(String s1, String s2) {
    int pos = 0;
    while (pos < s1.length() && pos < s2.length()) {
        int diff = s1.charAt(pos) - s2.charAt(pos);
        if (diff != 0) {
            return diff;
        }
        pos++;
    }
    return s1.length() - s2.length();här 1p
}
```

4. Totalt **13p**

   a) Lösningsförslag för uppgiften i den ursprungliga tentan (2020-06-03):

```
private static void printUpTo999(int n) {
    int n100 = (n / 100) % 10;
    if (n100 != 0) {
        System.out.print(UP_TO_19[n100]);
        System.out.print("hundra");
    }
    int n0 = n % 100;
    if (n0 >= 20) {
        System.out.print(TENS[n0 / 10]);
        System.out.print(UP_TO_19[n0 % 10]);
    } else {
        System.out.print(UP_TO_19[n0]);
    }
}

public static void main(String[] args) {
    Scanner scan = new Scanner(System.in);
    int n = scan.nextInt();

    if (n == 0) {
        System.out.print("noll");
    } else {
        if (n >= 1000) {
            printUpTo999(n / 1000);
            System.out.print("tusen");
        }
        printUpTo999(n % 1000);
    }
}
```

b) Lösningsförslag för den första kompletteringsuppgiften (2020-06-04):

```
public class Words {

    private static final String[] DELIMITERS = {",", ".", ":" , ";"};

    public static void main(String[] args) {
        Scanner scan  =  new Scanner(System.in);
        ArrayList<String> res = uniqueWords(scan.nextLine());
        for (String s : res) {
            if(s.length() > 0)
               System.out.println(s);
        }
    }

    public static ArrayList<String> uniqueWords(String s) {

        for (int i = 0; i < DELIMITERS.length;i++) {
            s = s.replace(DELIMITERS[i], " ");
        }

        String [] u =  s.split(" ");
        ArrayList<String> words = new ArrayList<String>();

        for(String w:u) {
            add(w, words);
        }
        return words;
    }

    private static void add(String s, ArrayList<String> words ) {
        for (String s1 : words) {
            if(s1.equals(s)) {
                return;
            }
        }
        words.add(s);
    }

}
```

c) Lösningsförslag för den andra kompletteringsuppgiften (2020-06-09):

```
private static ArrayList<String> translate(String w) {
    ArrayList<String> translations = new ArrayList<String>();
    for (int i = 0; i < TRANSLATIONS.length; i++) {
        if (TRANSLATIONS[i][0].equals(w)) {
            translations.add(TRANSLATIONS[i][1]);
        }
    }
    return translations;
}

public static void main(String[] args) {
    Scanner scan = new Scanner(System.in);
    String line = scan.nextLine();
    String[] words = line.split(" ");

    for (String w : words) {
        ArrayList<String> translations = translate(w);

        if (translations.size() == 0) {
            System.out.print("??? ");
        } else if (translations.size() == 1) {
            System.out.print(translations.get(0) + " ");
        } else {
            System.out.print(translations.toString() + " ");
        }
    }
}
```