Problem 1

A phone company stores information on one phone call in *one* 64-bit variable of type **long** which contains:

1. identifier of the calling customer (caller): 17-bit number, i.e., from interval $[0, 2^{17} - 1] = [0, 131071]$;
2. zone number of the caller (caller_zone): 7-bit number, i.e., from interval $[0, 2^7 - 1] = [0, 127]$;
3. identifier of the receiving customer (callee): 17-bit number;
4. zone number of the callee (callee_zone): 7-bit number;
5. duration of the call in seconds: 13-bit number, i.e., from interval $[0, 2^{13} - 1] = [0, 8191]$;
6. tariff number: 3-bit number, i.e., from interval $[0, 2^3 - 1] = [0, 7]$;

Write static functions

- **encode** — taking six numbers described above (as **int**s) and packing their values into *one* number of type **long**;
- **info** — printing information on one phone call passed to it as a single number of type **long**.

For example, the program

download *Telephones.java*

```
public class Telephones {
    public static void main(String[] args) {
        info(encode(130999,101,7777,99,7000,6));
    }

    public static long encode(int caller, int caller_zone,
                              int callee, int callee_zone,
                              int duration, int tariff) {
        // ...
    }

    public static void info(long res) {
        // ...
    }
}
```

should print

```
Caller:      130999
Caller_zone: 101
```

```
Callee:      7777
Callee_zone: 99
Duration   : 7000
Tariff     : 6
```

Do *not* use any tools from packages other than ***java.lang***.

Problem 2

Create class **Person** with fields name of type **String** and age of type **int**. Define the constructor, getters and override the **toString** method. Also, define a non-static method **compareTo** which takes the reference to another object of the same class and returns an **int** with value

- negative (otherwise arbitrary), if the object on which the method has been invoked (the *receiver*) corresponds to the younger person;
- zero, if both objects correspond to persons of the same age;
- positive, if the person passed to the function as the argument is younger.

In class **Person** add the static function **sort** which sorts an array of references to **Person**s in ascending order according to their age; use the **compareTo** method to compare persons.

Problem 3

Create a class **Task**, objects of which represent tasks to be done. Objects contain private field descr of type **String** with a description of a task and (also private) field next of type **Task** with the reference to the next task (or **null**). In the class, define constructors and methods

```
public Task(String d, Task n)
public Task(String d)
public void setNextTask(Task t)
public void printTasks() {
public static void printTasks(Task head)
public void printTasksRev()
public static void printTasksRev(Task head)
```

where

- the first constructor takes a description and a reference to the next task;
- the second takes only a description — the field next of the object being created is then set to **null** (this constructor should reuse the previous one!);
- method **setNextTask** sets the field next to the value of the passed reference t;
- method **printTasks** prints in one line descriptions of *this* task and of all the tasks that follow it;
- *static* function **printTasks** prints in one line descriptions of the task represented by head and of all the tasks that follow it (the function may reuse the method of the same name that has already been written);

- *recursive* method **printTasksRev** is analogous to **printTasks**, but prints descriptions in the reverse order, from the latest task to the one on which it has been invoked;
- static function **printTasksRev** printing descriptions in the reverse order, from the latest task to the one represented by **head** (the function may reuse the method of the same name that has already been written).

In a separate class write **main** function which tests the functionality of class **Task**. For example,

download *QueueOfTasks.java*

```
public static void main (String[] args) {
    Task t2 = new Task("Wash the dishes!");
    Task t1 = new Task("Walk the dog!",t2);
    t2.setNextTask(new Task("Clean the room!"));
    Task head = new Task("Get rest!",t1);

    head.printTasks();
    System.out.println();
    head.printTasksRev();
    System.out.println();

    System.out.println();

    Task.printTasks(head);
    System.out.println();
    Task.printTasksRev(head);
    System.out.println();
}
```

should print

```
Get rest! Walk the dog! Wash the dishes! Clean the room!
Clean the room! Wash the dishes! Walk the dog! Get rest!

Get rest! Walk the dog! Wash the dishes! Clean the room!
Clean the room! Wash the dishes! Walk the dog! Get rest!
```

Do not use arrays or any other collections!
