January 14, 2021

Task 12

Create a class **StringTask**, simulating a lengthy calculation (here, just concatenation of strings). The constructor takes, as argument, a string to replicate, and a (big) number indicating how many times it is to be replicated. The class should implement the **Runnable** interface and in its **run** function concatenates the string repeatedly, `s = s + word`, using + operator to string variables (this is the lengthy calculation).

Objects of class **StringTask** are treated as tasks that are to be performed in parallel with each other. Possible states of a task are:

- CREATED — the task has been created, but not yet started,
- RUNNING — the task is running in a separate thread,
- ABORTED — the task has been interrupted,
- READY — the task has been completed and its result is ready.

Define the following methods in class **StringTask**:

- `public String getResult()` — returns the result of the concatenation,
- `public TaskState getState()` — returns the status of the task,
- `public void start()` — launches the the task in a separate thread,
- `public void abort()` — aborts the execution of the task and the thread on which it is running,
- `public boolean isDone()` — returns **true** if the task has been completed normally or interrupted, **false** otherwise.

The following program:

download *SStringTask.java*

```
public class Main {
    public static void main(String[] args)
            throws InterruptedException {
        StringTask task = new StringTask("A", 70000);
        System.out.println("Task " + task.getState());
        task.start();
        if (args.length > 0 && args[0].equals("abort")) {
            /*<-
                here add the code creating a new
                thread which sleeps for a second and
                then aborts the task 'task'
            */
        }
        while (!task.isDone()) {
            Thread.sleep(500);
            switch(task.getState()) {
                case RUNNING:
                    System.out.print("R.");
                    break;
                case ABORTED:
                    System.out.println(" ... aborted.");
                    break;
                case READY:
                    System.out.println(" ... ready.");
                    break;
                default:
                    System.out.println("unknown state");
            }
        }
        System.out.println("Task " + task.getState());
        System.out.println(task.getResult().length());
    }
}
```

run without an argument should print something like:

```
Task CREATED
R.R.R.R.R.R.R.R.R. ... ready.
Task READY
70000
```

and run with argument "abort" may output:

```
Task CREATED
R. ... aborted.
Task ABORTED
34789
```

As the result of calculations is a very long string, the program prints only its length.
Note:

- File ***Main.java*** can be modified only in place marked by `/*<- ...  */`;
- Do not use the method **System.exit**.

Deadline: Jan 23 (inclusive)
