W12H01 - Work Life Balance of Threaduins

Hausaufgabe Easy

Points: 8 of 8 Assessment: automatic

Submission due: 6 months ago

Complaint due: 6 months ago

100% (7 months ago) GRADED

Recent results:

100% (7 months ago)	75% (7 months ago)	100% (7 months ago)	100% (7 months ago)	100% (7 months ago)
GRADED	GRADED	GRADED	GRADED	GRADED

Show all results

Tasks:

Work Life Balance of Threaduins

Wie auch an der TUM ist die PUM eine hoch diversifizierte Hochschule. Von überengagierten Pinguinen bishin zu Schlafmützen findet man alles. Du als ausgewogenes Individuum hast dich mit vielen Pinguinen angefreundet. Weil du so ein guter Freund bist, sorgst du dich um ihre Gesundheit und ihren Erfolg. Zeit deine Pingu-Freunde zu unterstützen!

Aufgabe

Hinweis 1: Alle `String`s, die in dieser Aufgabe benötigt werden, sind in der `Threaduins` Klasse als `final String` deklariert. Benutze am Besten diese `String`s, damit du dir keine Sorgen um Typos machen musst. (z.B. `Threaduins.STOP_MSG`)
Hinweis 2: Du kannst bei allen Methoden davon ausgehen, dass dir keine `null`-Values übergeben werden. Du erhältst also nur echte `Thread`s oder `PrintStream`s, die voll funktional sind. In anderen Worten: Mach dir keine Gedanken über Edge Cases.
Hinweis 3: Zum Testen eignet sich das vorgegebene `ConsoleSignal` nicht. Daher enthält das Template die Methode `setSignal`, mit der du eigene Implementierungen verwenden kannst. Achte darauf, dass die Methode wirklich nur zum Testen verwendet wird.

1. **Workaholic Penguin** 1 of 1 tests passing
   Ein Workaholic Pinguin ist ein `Thread`, der ständig arbeitet, außer er wird dazu gezwungen aufzuhören. Implementiere die Methode `getWorkaholic(PrintStream)`, die einen Workaholic Pinguin zurückgeben soll. Dieser soll, solange er arbeitet, ständig die Nachricht `WORKAHOLIC_WORKING_MSG` an den spezifizierten `PrintStream` senden (Jede Nachricht soll dabei in einer neuen Zeile stehen, benutze also am besten `PrintStream.println(String)`). Wird der Pinguin bei seiner Arbeit gestört, soll er einmalig die Nachricht `WORKAHOLIC_STOP_MSG` an den `PrintStream` senden und anschließend seine Arbeit niederlegen.
2. **Save a Workaholic** 1 of 1 tests passing
   Implementiere die Methode `stopWorkaholic(Thread)`. Die Methode soll den gegebenen Workaholic Pinguin zunächst starten. In der Konsole soll nun die Nachricht `STOP_MSG` ausgegeben und auf das `Signal` zum stoppen des `Thread`s gewartet werden. Das `Signal signal` ist bereits als `static` Variable gesetzt. Ein `Signal` implementiert die Methode `await()` die den aktuellen `Thread` auf ein Signal warten lässt. Erhältst du das Signal, wird da Programm wie gewohnt nach dem Aufruf von `await` weiter ausgeführt. Sobald du das Signal erhalten hast, soll die Arbeit des Pinguins unterbrochen werden. Achte darauf, dass er seine Arbeit beendet, bevor du die Nachricht `STOPPED_MSG` in der Konsole ausgibst. *Tipp:* Wenn du deine Implementierung ausprobieren möchtest, wirf einen Blick in die `main` Methode, dort findest du schon den entsprechenden Code. Das Signal ist in diesem Fall eine beliebige Eingabe in der Konsole.
3. **Procrastinating Penguin** 1 of 1 tests passing
   Ein prokrastinierender Pinguin ist ein `Thread` und prokrastiniert auf höchstem Niveau! Implementiere die Methode `getLuckyProcrastinator(PrintStream)`, die einen prokrastinierenden Pinguin zurückgibt. Dieser sendet an den spezifizierten `PrintStream` die Nachricht `PROCRASTINATOR_PROCRASTINATING_MSG`, sobald er mit dem Prokrastinieren beginnt. Ab dann wartet er unbegrenzt, bis ein lieber Freund ih an die Deadline der PGdP Hausaufgaben erinnert (i.e. bis ein anderer Thread ihn benachrichtigt). Obwohl er es eigentlich besser wissen sollte, ist e natürlich über die Deadline überrascht und sendet, nachdem sein Warten durch die Benachrichtigung beendet wurde, die Nachricht `LUCKY_PROCRASTINATOR_WORKING_MSG` an den `PrintStream`. (`PrintStream.println(String)`)
4. **Friend of a Procrastinator** 1 of 1 tests passing
   Implementiere die Methode `stopProcrastinator(Thread)`. Die Methode soll den gegebenen prokrastinierenden Pinguin zunächst starten. In der Konsole soll nun die Nachricht `STOP_MSG` ausgegeben und auf das `Signal` zum Erinnern des `Thread`s gewartet werden (`await`). Sobald du das Signal erhalten hast, musst du den Pinguin an die annähernde PGdP Deadline erinnern (i.e. benachrichtigen). Achte darauf, dass er seine Nachricht an der `PrintStream` senden konnte, bevor du die finale Ausgabe `STOPPED_MSG` in der Konsole tätigst. (*Wichtig:* der Pinguin soll erinnert werden, aber nicht dazu gezwungen werden.)

Tests

Hier werden dir die Ergebnisse der automatischen Tests direkt angezeigt:

Öffentliche/Grading Tests 4 of 4 tests passing
Testet deine Abgabe vor der Deadline und verteilt Punkte.

Versteckte Tests 14 of 14 tests passing
Testet deine Abgabe nach der Deadline.

Du bekommst bei dieser Aufgabe von den PublicTests (den Tests, die direkt nach jedem Push ausgeführt werden), zwar kein Feedback darüber, was du falsch gemacht hast, allerdings teilen sie dir nach jedem Push mit, ob du die jeweilige Teilaufgabe korrekt gelöst hast oder nicht. Daher kannst du dir auch sicher sein, dass du alle Punkte bekommst, wenn du alle PublicTests bestehst (ausgenommen du hältst dich nicht an eine der in der Aufgabenstellung geforderten Einschränkungen). Die HiddenTests sind deckungsgleich mit den PublicTests und geben dir nach der Deadline ein Feedback, warum sie fehlgeschlagen sind.

Viel Erfolg!

Lösungsvorschlag

Tests

Ein Paar Tage nach der Deadline werden diese beiden Links zu der Musterlösung bzw. unseren PublicTests und HiddenTests führen.

Exercise details

Release date:	Dec 22, 2022 18:30
Start date:	Dec 22, 2022 18:30
Submission due:	Feb 5, 2023 18:00
Complaint due:	Feb 12, 2023 18:00

Every student is allowed to complain once per exercise. In total 1000 complaints are possible in this course. You still have **998** complaints left.