

Отчет по лабораторной работе №6

Студент: Стоколяс Юрий Юрьевич

Группа: 6201-120303D

Выполненные задания

Задание 1

1. Добавлен численный метод интегрирования (трапеции) в `functions/Functions.integrate()`.
2. Проверяются корректность шага и попадание пределов в область определения функции.
3. В `Main.task1()` вычислен интеграл `exp` на `[0;1]`; шаг уменьшается до достижения точности в 7-м знаке.

Задание 2

1. Реализован последовательный сценарий `Main.nonThread()`.
2. В каждой из ≥ 100 задач случайно выбираются основание логарифма, границы интегрирования и шаг.
3. На каждой итерации выводятся сообщения `Source ...` и `Result ...`, интеграл считается новым методом.

Задание 3

1. В пакете `threads` созданы `Task`, `SimpleGenerator`, `SimpleIntegrator`.
2. Поток использует синхронизированные блоки при доступе к общему заданию, что исключает `NullPointerException` и смешивание данных.
3. Метод `Main.simpleThreads()` запускает оба потока и дожидается завершения.

Задание 4

1. Добавлен одноместный семафор `ReadWriteSemaphore` с разделением операций записи и чтения.
2. Поток `Generator` и `Integrator` используют семафор вместо синхронизации для передачи задания.

3. В `Main.complicatedThreads()` потоки запускаются, через 50 мс прерываются методом `interrupt()`, обработка прерывания реализована в потоках.

Результаты тестирования

Подбор шага интегрирования (task1):

Интеграл \exp на $[0,1]$: 1.7182818625982395, шаг: 4.8828125E-4
Точное значение: 1.718281828459045

NoThread :

```
Source 39.06109527452079 156.31954169071867 0.8405635658387679
Result 39.06109527452079 156.31954169071867 0.8405635658387679 788.25
Source 43.206937228599344 106.71622842940015 0.213148952090219
Result 43.206937228599344 106.71622842940015 0.213148952090219 159.50
Source 51.57867270438062 163.5628108026674 0.6627137182965686
Result 51.57867270438062 163.5628108026674 0.6627137182965686 329.69
...
Source 89.17062519729784 139.53346951062974 0.9857258312172914
Result 89.17062519729784 139.53346951062974 0.9857258312172914 196.50
Source 87.81322669622092 149.25292164664447 0.10098058413813127
Result 87.81322669622092 149.25292164664447 0.10098058413813127 148.80
Source 21.19058952680997 119.09403839460597 0.9667564624267146
Result 21.19058952680997 119.09403839460597 0.9667564624267146 178.00
Source 41.110967850072306 172.85719801391429 0.25269652833298006
Result 41.110967850072306 172.85719801391429 0.25269652833298006 465.00
```

SimpleThreads (SimpleGenerator + SimpleIntegrator):

```
Source 12.044109112880396 185.90635294750332 0.6268474987818626
Result 12.044109112880396 185.90635294750332 0.6268474987818626 390.40
Source 97.25622152249988 101.32781801511031 0.6150837816446524
Result 97.25622152249988 101.32781801511031 0.6150837816446524 7.8450
```

```
Source 70.3909149345068 194.66942738777847 0.11946406025301197
Result 70.3909149345068 194.66942738777847 0.11946406025301197 300.50
...
Source 69.01543847060661 125.26013294045501 0.3824550214509183
Result 69.01543847060661 125.26013294045501 0.3824550214509183 81.459
Source 50.278491290252846 112.23843743869564 0.2147504413293703
Result 50.278491290252846 112.23843743869564 0.2147504413293703 20.60
```

ComplicatedThreads (Generator + Integrator + ReadWriteSemaphore):

```
Source 74.91365278077807 186.0338432392602 0.21193913882830848
Result 74.91365278077807 186.0338432392602 0.21193913882830848 187.50
Source 89.63793677625015 118.33532383270534 0.7604482563060078
Result 89.63793677625015 118.33532383270534 0.7604482563060078 28.337
Source 26.92545309897818 191.47610724655114 0.46735044272342314
Result 26.92545309897818 191.47610724655114 0.46735044272342314 498.1
...
Source 77.57336967929649 122.3407179694488 0.26183474408536155
Result 77.57336967929649 122.3407179694488 0.26183474408536155 52.557
Source 55.214905843504754 110.74210047320738 0.10756231507348337
Result 55.214905843504754 110.74210047320738 0.10756231507348337 22.1
```