Задачи за задължителна самоподготовка

по

Структури от данни и програмиране

"Реактивни" безкрайни потоци

email: kalin@fmi.uni-sofia.bg

21 ноември 2017 г.

ВНИМАНИЕ: Решението на следите задачи е публикувано към кода от лекции. Разгледайте решенията, осмислете ги и ги пресъздайте самостоятелно (или предложете друго решение на задачата за реализация на операции над безкрайни потоци).

1. Към класа `StreamBase` от реализираната на лекции йерархия да се добави метод за печатане на първите `n` елемента от потока. Методът да връща остатъка от потока.

   Следният пример отпечатва първите 5 нечетни числа:

   ```
   ints.filter(odd).print(2).print(3);
   ```

2. Да се дефинира поток `RepeatStream`, състоящ се от безкрайно повторение на дадено число.

   Следният пример отпечатва 5 единици:

   ```
   RepeatStream ones(1);
   ones.print(5);
   ```

3. Да се дефинира клас `SumStream`, който позволява по дадени два потока `A` и `B` да се генерира нов поток, всеки от елементите на който е сумата (получена с оператора `+`) на двата съответни елемента на

`A` и `B`. В клас `StreamBase` да се добави метод `sum`, с който да може всеки поток да се сумира с друг поток.

Следният пример отпечатва първите 5 четни числа:

```
ints.sum(ints).print (5);
```

Следният пример отпечатва 5 двойки:

```
ones.sum(ones).print (5);
```

4. Да се дефинира клас `ZipStream`, който позволява по дадени два потока $A = (a_i)$ и $B = (b_i)$ да се генерира нов поток $C = (c_i)$, всеки от елементите на който е резултат от приложението на някаква функция $f : int \times int \to int$ над двата съответни елемента на `A` и `B`, т.е. $c_i = f(a_i, b_i)$. В клас `StreamBase` да се добави метод `zip`, с който да може всеки поток да се комбинира с друг поток.

   Следният пример отпечатва първите 5 точни квадрата:

   ```
   ints.zip(ints,mult).print (5);
   ```

   Където $mult(x, y) = x * y$

5. Да се дефинира клас `ZigZagStream`, който позволява по дадени два потока $A = (a_i)$ и $B = (b_i)$ да се генрира нов поток $C = (c_i)$ от алтернативно разменянищите се последователни елементи на A и B, т.е. $c_0 = a_0, c_1 = b_0, c_2 = a_1, c_3 = b_1, ...$ В клас `StreamBase` да се добави метод `zigzag`, с който да може всеки поток да се комбинира с друг поток.

   Следният пример отпечатва първите 5 естествени числа, последвани от вторите им степени:

   ```
   ints.zigzag(ints.zip(ints,mult)).print (5);
   ```

6. Да се дефинира клас `AxisStream`, чрез който по дадена фунцкия $f : int \to int$ и число y_0 да се генерира поток от елементите $(f^i(y_0))$, където $f^i(y_0) = f(f..f(y_0))$ е i-кратното приложение на функцията f над аргумента y_0 за $i = 0...$, т.е. потокът се състои от елементите $y_0, f(y_0), f(f(y_0)), ...$

   Следният пример отпечатва първите 5 степени на числото 2:

```
Axis powers2 (sqr,2);
powers2.print (5);
```

Където $sqr(x) = x * x$

7. Какво трябва да се промени в йерархията така, че да може да се конструират потоци с елементи, различни от int?