Programowanie funkcyjne — kolokwium nr 2, 26.01.2022

Instrukcja: Każde zadanie należy przesłać na Pegaza w oddzielnym pliku: zadanie1.hs, zadanie2.hs i zadanie3.erl. Plików nie należy zipować. Rozwiązania muszą się poprawnie kompilować. W rozwiązaniach nie można korzystać z modułów innych niż standardowe; niedozwolone jest użycie polecenia import. Zadania 1 i 2 napisać w Haskellu, zadanie 3 — w Erlangu. Rozwiązania nie spełniające powyższych wymogów nie będą oceniane. Punktacja: 10 punktów za każde zadanie. Uwaga: korzystanie z internetu poza wyznaczonym czasem skutkuje automatycznym otrzymaniem 0 punktów.

Zadanie 1. Napisać program, który rysuje (w sensie ASCII art) choinkę o zadanej wielkości w następujący sposób: pyta o wielkość choinki (liczba naturalna dodatnia), a następnie wypisuje na standardowe wyjście choinkę żądanej wielkości złożoną ze spacji oraz znaków /, \, ^ i |, składającą się z części górnej, dolnej i pnia, przy czym:

- Część górna powinna w pierwszej linijce zawierać odpowiednią liczbę spacji, po której następują znaki /\.
- Każda kolejna linijka części górnej powinna składać się ze spacji i znaków / oraz \ w taki sposób, aby znak / pojawił się o jeden znak wcześniej niż w poprzedniej linijce, a znak \ — o jeden znak później.
- Ostatnia linijka częsci górnej ma być tworzona analogicznie do poprzednich, ale ma nie zaczynać się od spacji (czyli choinka musi zostać wypisana „maksymalnie po lewej").
- Część dolna powinna składać się z wielokrotnie powtórzonego znaku ^, tyle razy, ile jest w sumie znaków od / do \ w ostatniej linijce części górnej.
- Pień powinien składać się ze spacji i dokładnie dwóch znaków | w taki sposób, że znaki | powinny być umieszczone dokładnie pod szczytem choinki (znakami /\).

Rozmiar choinki to liczba linijek jej części górnej (która — zgodnie z zasadami powyżej — implikuje szerokość części dolnej i położenie pnia). Przykładowe choinki dla liczb 1 i 3:

```
/\
^^
||
```

```
   /\
  /  \
 /    \
^^^^^^
  ||
```

Zadanie 2. Napisać bezpunktowo funkcję f :: [(Int, Int)] → Int → Int taką, że

$$f\ [(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)]\ u = u * (x_n - y_n) * (x_{n-1} - y_{n-1}) * \ldots * (x_2 - y_2) * (x_1 - y_1).$$

W rozwiązaniu należy użyć funkcji foldl lub foldr. Rozwiązanie trzeba uzasadnić, np. przedstawiając w komentarzach kolejne kroki. Listy podawane jako argumenty będą zawsze niepuste.

Zadanie 3. Napisać moduł uruchamiający n procesów, które roboczo numerujemy od 1 do n, przy czym $n \geq 3$. Proces o numerze i odbiera pojedynczą liczbę lub listę liczb x, wypisuje na ekran parę (i, x) i przesyła następnemu procesowi nową wartość y, wyliczoną tak:

- Jeżeli x jest pojedynczą liczbą, to y jest listą $[i, x]$.
- Jeżeli x jest listą, której suma nie przekracza n^2, to y jest listą $[i * h \mid x]$, gdzie h oznacza głowę listy x.
- Jeżeli x jest listą, której suma przekracza n^2, to y jest liczbą h.

Przez następny proces rozumiemy proces o numerze $(i \bmod n) + 1$.

Moduł ma udostępniać funkcję start(N, X), która uruchomi N procesów w stosownym cyklu tak, by całość wypisywała wyniki działania procesów, poczynając od procesu o numerze 1, któremu na początku zostanie przekazana pojedyncza liczba X. Przykładowo, wywołanie start(4, 3) powinno spowodować wypisanie następujących par: $(1, 3)$, $(2, [1, 3])$, $(3, [2, 1, 3])$, $(4, [6, 2, 1, 3])$, $(1, [24, 6, 2, 1, 3])$, $(2, 24)$, $(3, [2, 24])$, $(4, 2)$, ...