Kolokwium ze Wstępu do Informatyki

05.11.2012

Czas trwania: 1.5h.

1. (a) (3p.) Segmentem w tablicy liczb całkowitych nazwiemy spójny (być może pusty) fragment listy ograniczony z obu stron zerem lub końcem listy. Napisz funkcję sumy(l), która dla zadanej listy liczb całkowitych l zwróci listę sum jej kolejnych segmentów. Przyjmujemy, że sumy pustych segmentów są pomijane.
   Przykład: segmentami w liście l=[1, 2, -1, 0, 2, 3, 0, 0, -1, 0] są [1, 2, -1], [2, 3], [], [-1], [], a sumy(l)=[2, 5, -1].

   (b) (2p.) W wyniku działania funkcji sumy na liście l otrzymujemy listę sum segmentów, która jest krótsza od listy l. Napisz funkcję rekurencyjną suma_sum, która będzie wyliczała kolejne listy sum segmentów aż do otrzymania listy jednoelementowej. Jako wynik powinna zwracać wartość jedynego elementu końcowej listy.
   Przykład: dla l=[2, 0, 1, 2, -3, 0, 1, 2, 1, 0] dostaniemy w kolejnych krokach listy sum segmentów: [2, 0, 4], [2, 4], [6], zatem suma_sum(l)=6.

2. (5p.) Dane są dwie funkcje wzajemnie rekurencyjne zdefiniowane następująco:

```
def a_r(n):
    if n == 1:
        return 1
    else:
        return a_r(n-1) + b_r(n) + 1
def b_r(n):
    if n == 1:
        return 1
    else:
        return a_r(n-1) + b_r(n-1) + 2
```

   Napisz iteracyjną funkcję a_i(n), która dla zadanej liczby całkowitej dodatniej n zwróci wartość a_r(n).

3. (5p.) Dana jest lista liczb naturalnych l o długości n kodująca pewną permutację listy [0, 1, ..., n - 1]. Zasada kodowania jest następująca. Początkowo mamy daną pełną listę p=[0, 1, ..., n - 1]. Z listy p wybieramy następnie elementy o indeksach wskazywanych przez kolejne elementy listy l. Raz wybrana liczba jest następnie wyrzucana z listy p. Napisz funkcję dekoduj(l), która dla zadanej listy-kodu l zwróci odszyfrowaną permutację.
   Przykład: dekoduj([3, 3, 0, 1, 0]) = [3, 4, 0, 2, 1], co odpowiada skracaniu listy [0, 1, 2, 3, 4] w następujący sposób: [0, 1, 2, 4], [0, 1, 2], [1, 2], [1].

Powodzenia!