Kolokwium
Wstęp do programowania
12 stycznia 2012

Zadanie 1.(15pkt) Napisz w dowolnym języku program funkcję `robobalwanek(a,b,c)`, która rysuje roboto-bałwanka z gwiazdek, składającego się z trzech kwadratów, o bokach `a`, `b` i `c`. Możesz założyć, że `a<b<c`. Roboto-bałwanek powinien być symetryczny[1], powinien również przylegać do lewej krawędzi okna. Rysowanie segmentu bałwanka powinna realizować osobna funkcja, z dwoma parametrami.

```
   **
   **
  ****
  ****
  ****
  ****
********
********
********
********
********
********
********
********
********
```

Powyższy rysunek to wynik wywołania `robobalwanek(2,4,8)`.

Zadanie 2.(15pkt) Rozważmy ciąg liczb całkowitych dodatnich, w którym każdy kolejny wyraz liczymy na podstawie poprzedniego zgodnie z następującą, prostą zasadą:

- Jeżeli bieżący wyraz jest parzysty, to następny otrzymujemy dzieląc bieżący przez 2
- Jeżeli bieżący wyraz jest nieparzysty, to następny otrzymujemy mnożąc bieżący przez 3 i dodając 1.

Oczywiście to, jakie kontretne wyrazy znajdują się w ciągu zależy od tego, od jakiego wyrazu zaczniemy. Przykładowo, jeżeli zaczniemy od liczby 3, to ciąg wygląda tak:

3 10 5 16 8 4 2 1 4 2 1 ...

Istnieje hipoteza[2], której nikt nie umie udowodnić, że niezależnie od jakiej liczby zaczniemy, to zawsze dojdziemy do jedynki (po której moglibyśmy w nieskończoność powtarzać wartości 4, 2, 1).

Napisz w C lub Pythonie funkcję, która pomoże sprawdzić tę hipotezę. Powinna ona przyjmować dwa argumenty: pierwszy wyraz ciągu oraz maksymalną liczbę wyrazów m, jaką chcemy rozpatrzyć w eksperymencie. Powinna zwracać pozycję pierwszej jedynki w ciągu (numerując od 0) lub -1, jeżeli wśród pierwszych m wyrazów ciągu nie ma jedynki. Nie musicz przejmować się zakresem wybranego typu całkowitoliczbowego.

Zadanie 3.(15pkt) Funkcja `pomieszaj` zdefiniowana jest w następujący sposób:

```
def pomieszaj(s):
      L = list(s)
      for i in range(len(L)/2):
         p1 = 2*i
```

[1] Jeżeli różnica długości między kolejmymi segmentami jest nieparzysta, to dopuszczalne jest drobne pogwałcenie symetrii.

[2] Możesz poczytać o tym po kolokwium, przykładowo zaczynając od hasła Problem Collatza na wikipedii

```
        p2 = 2*i+1
        L[p1],L[p2] = L[p2],L[p1]
    return "".join(L)
```

Rozważ następującą sesję poleceń interpretera w pythonie:

```
>>> L = [1,2,3]
>>> L[-1] = pomieszaj("abanyn")
>>> L[0] = str(L[0]) + 3 * str(L[1])
>>> L[1] = L[1] * [1]
>>> L += [13]
>>> L.append([ pomieszaj(L[2][1:5]) ] )
```

Napisz, jaką wartość mają kolejne elementy listy `L` po zakończeniu tej sesji. Jeżeli nie znasz któregoś z tych elementów, to wstaw w jego miejsce znak zapytania.

Zadanie 4.(20pkt) Napisz w języku C funkcję `void sortujZnaki(char *s)`, która „sortuje" napis `s`, o którym wiemy, że składa się z samych znaków `'0'` oraz `'1'`. Przy czym napis jest posortowany, jeżeli każdy jego kolejny znak ma kod ascii większy bądź równy poprzednikowi. Czyli jeżeli przed wywołaniem tej funkcji napis `s` miał wartość `01101001010011`, to po wywołaniu tej funkcji napis `s` powinien mieć wartość `00000001111111`.

Nie wolno Ci korzystać z żadnych funkcji bibiotecznych.

Zadanie 5.(20pkt) Będziemy rozważać listę słów, składających się z małych i dużych liter alfabetu łacińskiego. Słowa, które różnią się tylko wielkością liter, nazwiemy równoważnymi. W zadanym ciągu słów, dla dwóch słów równoważnych *lepszym* nazwiemy to, które ma więcej małych liter, a jeżeli dwa różne (a równoważne) słowa mają po tyle samo małych liter, to za *lepsze* uznamy to, które w danym ciągu występiło jako pierwsze.

a) Napisz funkcję `zliczMale`, która dla słowa zwraca liczbę małych liter znajdujących się w tym słowie (cały czas obowiązuje założenie, że rozważamy tylko litery łacińskie)

b) Napisz funkcję `normalizacja`, która bierze listę słów L i zwraca listę tej samej długości, w której każde słowo w zostało zastąpione najlepszym słowem z listy wejściowej L, które jest równoważne w.

Przykładowo dla list (dla czytelności pomijamy cudzysłowy i przecinki)

```
[ Pojechal Janek by zobaczyc Ale  potem pojechal do Oli lecz o dziwo zastal tam ALE ]
[ Palac Kultury to PKiN a nie PKIN czy Pkin ]
[ AAAaaa aAaAaA AAAAaa AAAaAA BBB ]
```

wynikami powinny być odpowiednio

```
[ pojechal Janek by zobaczyc Ale  potem pojechal do Oli lecz o dziwo zastal tam Ale ]
[ Palac Kultury to Pkin a nie Pkin czy Pkin ]
[ AAAaaa AAAaaa AAAaaa AAAaaa BBB ]
```

Przypominam, że metoda `lower` wywołana dla napisu zmienia jego wielkość liter na małe.

Zadanie 6.(20pkt) Napisz w Pythonie funkcję `flatten`, która spłaszcza rekurencyjną listę liczb całkowitych[3], czyli przykładowo zmienia listę:

```
[ [1,2], [3,4], [[[5]]], [[6],[[7],[8]]] ]
```

na listę

```
[1, 2, 3, 4, 5, 6, 7, 8]
```

Przypominam, że typ danego obiektu pythonowego można sprawdzić za pomocą funkcji `type`.

[3] Rekurencyjną listą liczb całkowitych jest liczba całkowita lub lista, której wszystkie elementy są rekurencyjnymi liczbami całkowitymi. Dodatkowo zakładamy, że rekurencyjna lista liczb całkowitych nie zawiera referencji do samej siebie.