Računalništvo (Fizika): 3. izpit

5. september 2014

Čas reševanja: 120 minut. Doseženih 100 točk šteje za maksimalno oceno.

Navodila: Rešitev je potrebno oddati na spletni učilnici kot eno ZIP ali 7z datoteko, ki vsebuje rešitve posameznih nalog v ločenih datotekah. Posamezne datoteke poimenuj `PriimekIme_Nx`*, kjer je* `x` *zaporedna številka naloge. Pri reševanju nalog je dovoljena uporaba interneta (brskanje po referencah). Uporaba komunikacijskih orodij (email, facebook, ipd.) je prepovedana. Strogo prepovedana je kakršna koli medsebojna komunikacija in komunikacija s tretjimi osebami na kakršen koli način (tudi preko interneta).*

Naloga 1 [25 točk]

Pripravi delovni zvezek v *Mathematici*, v katerem rešiš naslednjo nalogo. Podana je parametrična krivulja:

$$x = t^2 - 5t + 6,$$
$$y = -t^2 - 4t.$$

(a) Nariši graf krivulje za parameter $t \in [-5, 4]$.

(b) Izračunaj vrednost parametra t za točko na krivulji, ki je najbližje izhodišču. Nariši tangento na krivuljo v tej točki.

(c) Izračunaj vsoto ploščin treh omejenih površin, ki jih objema na eni strani krivulja na drugi strani pa kaka od koordinatnih osi (ena ali obe).

Naloga 2 [25 + 10 točk]

Permutacije reda n so bijektivne preslikave na množici $0, \ldots, n-1$, ki jih podamo s seznamom slik. Npr. `[2,0,1]` predstavlja permutacijo, ki slika $0 \mapsto 2$, $1 \mapsto 0$ in $2 \mapsto 1$. Sestavi razred `Permutacija`, ki ima naslednje metode:

(a) konstruktor `__init__(self, s)`, ki ustvari permutacijo iz podanega seznama slik.

(b) metodo `__call__(self, m)`, ki vrne sliko števila m (če je m prevelik ali premajhen vrže izjemo).

(c) metodo `__mul__(self, p)`, ki vrne novo permutacijo, ki je kompozitum permutacije `self` in permutacije `p`. Pazi: gre za komponiranje od leve proti desni! Izraz `a*b` pomeni, da najprej slika permutacija `a`, potem pa še permutacija `b`.

(d) **Dodatna naloga [+10 točk]**: Napiši funkcijo `generiraj(n)`, ki vrne seznam vseh permutacij reda n.

Seveda lahko definirate še kakšne druge metode, ki olajšajo delo, na primer `__repr__`.

(Obrni!)

Naloga 3 [10 + 15 točk]

Dana je **urejena** tabela **različnih** celih števil $[a_0, a_1, \ldots, a_{n-1}]$.

(a) Napiši funkcijo `ujemanje(a)`, ki s časovno zahtevnostjo največ $O(\log n)$ ugotovi, ali v tabeli obstaja tak i, da velja: $a_i = i$. Primer: v tabeli `[-4,0,1,3,7,9]` je tak i enak 3.

(b) Urejeno tabelo krožno zavrtimo v desno za k mest. Npr. $[7, 9, 3, 5, 6]$ je tabela zavrtena za $k = 2$. Napiši funkcijo `maksimum(t)`, ki s časovno zahtevnostjo največ $O(\log n)$ najde maksimalni element v zavrteni tabeli `t`, če k ni znan.

Naloga 4 [5 + 10 + 10 + 10 = 35 točk]

Za dano zaporedje $A = [a_1, a_2, \ldots, a_n]$ označimo z $A_{ij} = [a_i, a_{i+1}, \ldots, a_j]$ strnjeno podzaporedje. Pravimo, da je A_{ij} *strnjeno palindromsko podzaporedje* zaporedja A, če velja $a_{i+h} = a_{j-h}$ za $0 \leq h \leq j - i$. Na primer, če je $A = [a, c, c, a, b, a]$, potem sta $A_{14} = [a, c, c, a]$ in $A_{46} = [a, b, a]$ strnjeni palindromski podzaporedji.

(a) Predlagaj algoritem in napiši funkcijo `maxPalindrom(A)`, ki vrne par (i, j), ki predstavlja indeksa začetka in konca najdaljšega strnjenega palindromskega podzaporedja. Kakšna je časovna zahtevnost?

(b) Opustimo pogoj strnjenosti. Najdaljše palindromsko podzaporedje zaporedja $A = [a, c, a, c, b, a]$ je potem $[a, c, a, c, a]$. Naj P_{ij} predstavlja dolžino najdaljšega palindromskega podzaporedja v strnjenem podzaporedju A_{ij}. S pomočjo metode dinamičnega programiranja izpelji in utemelji (v komentarju datoteke v Pythonu) rekurzivno zvezo P_{ij}. (*Namig: loči primere glede na enakost/neenakot a_i in a_j.*)

(c) Implementiraj funkcijo `P(A)`, ki izračuna tabelo vrednost P_{ij}, za $1 \leq i, j \leq n$. Časovna zahtevnost naj ne presega $O(n^2)$.

(d) **Dodatna naloga [+10 točk]**: Napiši funkcijo `maxPalindrom2(A)`, ki vrne seznam, ki predstavlja eno izmed najdaljših palindromskih podzaporedij.