Egzamin licencjacki/inżynierski — 14 września 2013

Z sześciu poniższych zestawów zadań (Matematyka I, Matematyka II, Programowanie, Matematyka dyskretna, Algorytmy i struktury danych i Metody numeryczne) należy wybrać i przedstawić na osobnych kartkach rozwiązania trzech zestawów. Za brakujące (do trzech) zestawy zostanie wystawiona ocena niedostateczna z urzędu. Egzamin uważa się za zaliczony, jeśli student rozwiąże z oceną dostateczną co najmniej 2 zestawy. Wtedy ocena z egzaminu jest średnią arytmetyczną ocen z trzech wybranych zestawów. Na rozwiązanie przeznacza się czas 3x40=120 minut. Po wyjściu z sali egzaminacyjnej w czasie egzaminu nie ma możliwości powrotu do tej sali i kontynuowania pisania egzaminu.

Matematyka I — Logika dla informatyków

Postacią ułamka egipskiego liczby wymiernej r nazywamy skończoną sumę o wartości r różnych ułamków o licznikach równych 1. Na przykład $\frac{9}{10}$ w postaci ułamka egipskiego to $\frac{1}{2} + \frac{1}{3} + \frac{1}{15}$. Udowodnij indukcyjnie, że każdą dodatnią liczbę wymierną $\frac{p}{q}$ możną przedstawić w takiej postaci.

Wskazówka: postać ułamka egipskiego można znaleźć algorytmem zachłannym wybierając w kolejnych iteracjach najmniejszy możliwy mianownik. Przyjrzyj sie różnicom pomiędzy tak konstruowanymi sumami częściowymi a początkową liczbą. Indukcja względem p.

Matematyka II — Algebra

Za zadania można otrzymać 14 punktów. Aby otrzymać ocenę dostateczną, należy zdobyć 5 punktów, próg dla dst+ to 6.5p, dla db – 8p, dla db+ 9.5p, dla bdb – 11p.

Zadanie 1. (6 punktów)
i. Wyznaczyć całkowite liczby a, b takie, że $21a + 23b = 1$.
ii. Rozwiązać równanie $21x \equiv_{23} 1$.
Odpowiedzi uzasadnić.

Zadanie 2. (4 punkty)
Stosując algorytm eliminacji Gaussa wyznaczyć macierze L, U takie, że

$$L\,U = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \end{bmatrix}.$$

L oznacza macierz trójkątną dolną z jedynkami na przekątnej; U to macierz trójkątna górna.

Zadanie 3. (4 punkty)
Dane jest przekształcenie liniowe $T : \mathbb{R}^3 \to \mathbb{R}^3$, określone wzorem

$$T(x_1, x_2, x_3) = (x_1 + x_2,\ 2x_1 - x_3,\ x_1 + x_2 - x_3).$$

Znaleźć macierz tego przekształcenia w bazie $f_1 = (1, 1, 1)$, $f_2 = (1, 2, 3)$, $f_3 = (1, 4, 9)$.

Programowanie

Za zadanie można otrzymać 20 punktów. Aby otrzymac ocenę dostateczną, należy zdobyć 7 punktów, próg dla dst+ to 9p, dla db – 11p, dla db+ 13p, dla bdb – 15p.

Część 1. Gramatyka G_1 z symbolem startowym S nad alfabetem $\{a, b, c\}$ dana jest za pomocą następującego zbioru produkcji:

$$\{S \to \varepsilon,\ S \to abSc,\ S \to cSab,\ S \to SS\}$$

Dla gramatyki G przez $L(G)$ rozumieć będziemy język generowany przez G. Dla wyrażenia regularnego r przez $\mathcal{L}(r)$ rozumiemy język opisany przez wyrażenie r.

a) Czy *ababccabc* należy do $L(G_1)$? Odpowiedź uzasadnij. **(1)**

b) Opisz jakie słowa należą do języka $L(G_1)$ **(1)**

c) Dla następujący zbiorów przedstaw wyrażenia regularne je reprezentujące, ew. gramatyki bezkontekstowe je generujące. Przy czym, jeżeli to tylko jest możliwe, powinieneś używać wyrażeń regularnych. **(4)**

- $L(G_1) \cap \mathcal{L}(a^*b^*c^*)$
- $L(G_1) \cap \mathcal{L}((a^*b)^*c^*)$ $(\star)$
- $L(G_1) \cap \mathcal{L}((abc)^*)$

d) Napisz w języku imperatywnym[1] funkcję, która bierze jako argument napis i zwraca wartość logiczną prawda wtedy i tylko wtedy, gdy słowo należy do języka opisanego w punkcie $(\star)$ **(4)**

Część 2. Napisz w Haskellu lub Prologu program, który dla zadanej liczby całkowitej dodatniej N zwraca listę wszystkich rosnących list liczb dodatnich, takich że ich suma wynosi N. Przykładowo, dla liczby 6 wynikiem powinna być lista:

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

(lub jakaś jej permutacja, bowiem przyjmujemy, że kolejność elementów na listach nie ma znaczenia) Możesz definiować dowolnie wiele funkcji (predykatów) pomocniczych, każda definicja powinna być opisana: czyli musisz powiedzieć, co funkcja (predykat) robi oraz jakie jest znaczenie jej argumentów. Efektywność rozwiązania nie ma znaczenia, skoncentruj się na czytelności kodu. **(10)**

Matematyka dyskretna

Niech a_n będzie liczbą wszystkich możliwych słów długości n na alfabecie $\{a, b, c\}$, które nie zawierają podciągu "aa". Napisz zależność rekurencyjną na a_n i ją uzasadnij.

[1] Wybranym ze zbioru: C, C++, C#, Java, Pascal, Python, Ruby

Algorytmy i struktury danych

Za rozwiązanie obydwu zadań z tej części można otrzymać w sumie do 9 punktów. Skala ocen: poniżej 3 punktów — ocena niedostateczna (egzamin niezdany), 3 punkty dają ocenę dostateczną, 4 — dostateczną z plusem, 5 — dobrą, 6 — dobrą z plusem, 7 albo więcej punktów daje ocenę bardzo dobrą.

Zadanie 1: słownik z operacjami *lessthan* i *greaterthan* (5 punktów)

Niech S będzie słownikiem, którego elementy pochodzą ze zbioru z porządkiem liniowym. Do operacji słownikowych *insert(x)*, *delete(x)* i *search(x)* dodajemy jeszcze operację *lessthan(z)* określającą ile jest w słowniku elementów mniejszych niż z oraz operację *greaterthan(z)* określającą ile jest w słowniku elementów większych niż z:

$$\begin{array}{lcl} S.lessthan(z) & = & |\{a \in S : a < z\}| \\ S.greaterthan(z) & = & |\{a \in S : a > z\}| \end{array}$$

Zaprojektuj taki słownik, w którym każda z wymienionych operacji ma pesymistyczną złożoność czasową $O(\log n)$, gdzie $n = |S|$. Procedury *lessthan(z)* i *greaterthan(z)* napisz w pseudokodzie i opisz ich działanie. Krótko ale precyzyjnie opisz, jak zmodyfikowałeś pozostałe procedury słownikowe.

Zadanie 2: podzbiór sumujący się do zadanej wartości (4 punkty)

Dany jest n–elementowy zbiór liczb naturalnych $A = \{a_1, a_2, \ldots, a_n\}$ oraz nieduża liczba naturalna T. Czy istnieje jakiś podzbiór A, który sumuje się do T? Każdą liczbę z wybranego podzbioru można wykorzystać tylko raz.

Opracuj algorytm o złożoności czasowej $O(nT)$ rozwiązujący ten problem. Uzasadnij poprawność opisanego algorytmu; oszacuj jego złożoność pamięciową.

Metody numeryczne

1. Niech dane będą macierze nieosobliwe $A, B \in \mathbb{R}^{n\times n}$. Opracuj oszczędny algorytm wyznaczania takiej macierzy $X \in \mathbb{R}^{n\times n}$, aby zachodziła równość

$$AXB = I,$$

gdzie $I \in \mathbb{R}^{n\times n}$ jest macierzą jednostkową. Podaj jego złożoność obliczeniową i pamięciową.