Egzamin licencjacki/inżynierski — 27 czerwca 2014

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

Zbiór $L(X)$ wszystkich skończonych list nad danym zbiorem X jest zdefiniowany indukcyjnie w następujący sposób:

- nil jest skończoną listą nad zbiorem X;
- jeśli x jest elementem zbioru X oraz xs jest skończoną listą nad zbiorem X to $x : xs$ jest skończoną listą nad zbiorem X.

Dla dowolnych zbiorów X i Y definiujemy funkcję $\mathsf{map} : Y^X \to L(Y)^{L(X)}$ w następujący sposób: dla dowolnej funkcji $f : X \to Y$

$$\begin{aligned} \mathsf{map}(f) &: L(X) \to L(Y) \\ (\mathsf{map}(f))(\mathsf{nil}) &= \mathsf{nil} \\ (\mathsf{map}(f))(x : xs) &= (f(x)) : \big((\mathsf{map}(f))(xs)\big). \end{aligned}$$

1. Sformułuj zasadę indukcji w takiej postaci, żeby można było jej użyć w dowodzie w punkcie 2.
2. Korzystając z zasady indukcji sformułowanej w punkcie 1 udowodnij indukcyjnie, że dla dowolnych zbiorów X, Y, Z i dowolnych funkcji $f : X \to Y$ i $g : Y \to Z$ oraz dla wszystkich list $xs \in L(X)$ zachodzi równość

$$(\mathsf{map}(g \cdot f))(xs) = \big((\mathsf{map}(g)) \cdot (\mathsf{map}(f))\big)(xs).$$

Wskazówka: Indukcja względem struktury listy xs. Jeśli w punkcie 1 przyjmiesz jakieś założenia (np. o zbiorach, własnościach lub porządkach) to nie zapomnij w punkcie 2 uzasadnić, że te założenia są spełnione. Dla przypomnienia: Y^X oznacza zbiór wszystkich funkcji z X w Y a $\cdot$ jest standardową operacją składania funkcji.

Matematyka II — Algebra

Za zadania można otrzymać 13 punktów. Aby otrzymać ocenę dostateczną, należy zdobyć 3 punkty, próg dla dst+ to 5p, dla db – 7p, dla db+ 9p, dla bdb – 11p.

Zadanie 1. (6 punktów)
Niech $G = \big\{A \in \mathbb{R}^{2\times 2} \,:\, \det(A) \neq 0\big\}$ oraz $H = \big\{A \in \mathbb{R}^{2\times 2} \,:\, \det(A) = 1\big\}$.

a) Udowodnić, że G jest grupą.
b) Wykazać, że $H < G$.

Zadanie 2. (3 punkty)
Obliczyć multiplikatywną odwrotność liczby 25 modulo 47, tzn. znaleźć x taki, że $25\,x \equiv_{47} 1$.

Zadanie 3. (4 punkty)
Niech f będzie homomorfizmem grup G oraz K. Wykazać, że:
i. $f(e_G) = e_K$.
ii. $f(a^{-1}) = f(a)^{-1}$.

Programowanie

Za zadanie można otrzymać 20 punktów. Aby otrzymać 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 $\{a, b\}$ dana jest za pomocą następującego zbioru produkcji:

$$\{S_1 \to abS_1b,\ S_2 \to bS_2ba,\ S_1 \to \varepsilon,\ S_2 \to \varepsilon,\ S \to S_1S_2\}$$

Gramatyka G_2 z symbolem startowym S nad alfabetem $\{a, b, c\}$ dana jest za pomocą następującego zbioru produkcji:

$$\{S \to aSa, S \to bSb, S \to cS, S \to \varepsilon,\ \}$$

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 *abbbba* należy do $L(G_1)$? Odpowiedź uzasadnij. **(1)**

b) Czy gramatyka G_1 jest jednoznaczna? Odpowiedź krótko uzasadnij. **(2)**

c) Przedstaw wyrażenie regularne lub gramatykę bezkontekstową generującą zbiór $A_1 = \mathcal{L}((ab)^*(ba)^*) \cap L(G_1)$ **(2)**

d) Przedstaw wyrażenie regularne lub gramatykę bezkontekstową generującą zbiór $A_2 = \mathcal{L}(a^*b^*c^*) \cap L(G_2)$ **(2)**

e) Napisz w języku imperatywnym funkcję, która bierze jako wejście napis i zwraca wartość logiczną, równą True wtedy i tylko wtedy, gdy ten napis należy do zbioru $L(G_1) \cap L(G_2)$. Możesz używać języka wybranego z następującej listy: C, C++, Java, C#, Python, Ruby, PHP, AWK, Pascal. **(3)**

Część 2. Napisz funkcję (w Haskellu) lub predykat (w Prologu), która bierze dwie listy elementów, łączy je w pary i umieszcza wynik w trzeciej liście. Jeżeli używasz Haskella podaj typ funkcji, jeżeli używasz Prologa podaj możliwie najogólniejsze sposoby użycia tego predykatu. **(4)**

Część 3. Co robią predykaty:

```
p1(A,L) :- append(X,_,L), append(_,A,X).
p2(L) :- \+ (append(A,[X|B],L), member(X,A),member(X,B))
```

Zaproponuj dla obu nazwy, które lepiej oddają ich działanie. **(4)**

Część 4. Jaki typ ma w Haskellu funkcja `map`. Czy i jaki typ ma wyrażenie „`map map`"? **(2)**

Matematyka dyskretna

Przedstaw zwarty wzór na sumę

$$\sum_{k=0}^{n} \frac{1}{k+1}\binom{n}{k}$$

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, 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: przyjęcie u Alicji (4 punkty)

Alicja chce zorganizwoać przyjęcie i zastanawia się, kogo zaprosić spośród n znajomych. Stworzyła już listę osób, które znają się nawzajem. Teraz chce wybrać możliwie z tej listy jak najwięcej osób, aby spełnione były dwa warunki: na przyjęciu każda osoba powinna *znać* co najmniej 5 innych osób oraz co najmniej 5 osób *nie znać*.

Podaj efektywny algorytm, który dostanie na wejściu listę n osób i listę par osób, które się znają, a daje na wyjściu możliwie najlepszą listę gości.

- Opisz algorytm, który rozwiązuje ten problem.
- Udowodnij, że Twój algorytm działa poprawnie.
- Przeanalizuj złożoność obliczeniową opisanej metody (oszacuj czas działania w zależności od parametru n).

Zadanie 2: słownik z operacjami `select` oraz `number` (5 punktów)

Niech S będzie słownikiem z dodatkowymi operacjami: `select(i)` i *number(x)*. Chcielibyśmy efektywnie wykonywać następujące operacje na tej strukturze danych:

1. $S =$ *new()* : $S \leftarrow \emptyset$ (utworzenie pustego zbioru);
2. *S.insert(x)* : $S \cup \{x\}$ (dodanie nowego elementu do zbioru);
3. *S.delete(x)* : $S \setminus \{x\}$ (usunięcie elementu ze zbioru);
4. *S.search(x)* : $x \in S$ (sprawdzenie czy element należy do zbioru);
5. $x =$ *S.select(i)* : wyznaczenie i–tego co do wielkości elementu w zbiorze S;
6. $i =$ *S.number(x)* : wyznaczenie numeru $x \in S$ względem wartości elementów w zbiorze.

Zaprojektuj taką strukturę danych dla S, która umożliwi wykonywanie wymienionych operacji w czasie logarytmicznym $O(\log n)$, gdzie $n = |S|$ (liczba elementów w S).

- Wykorzystaj jakąś znaną strukturę danych, która efektywnie realizuje operacje słownikowe. Opisz jak zaadoptować tę strukturę na potrzeby zadania.
- Napisz procedury *select(i)* oraz *number(x)* w pseudokodzie i wyjaśnij jak one działają.
- Krótko ale precyzyjnie opisz, jak należy zmodyfikować standardowe operacje słownikowe.
- Przeanalizuj złożoność czasową wszystkich operacji na tej strukturze.

Metody numeryczne

1. Znajdź postać Newtona wielomianu interpolacyjnego $L_3 \in \Pi_3$ dla następujących danych:

x_k	0	1	2	3
y_k	1	-1	3	11

2. Niech $\{x_0, x_1, \ldots, x_n\}$ będzie zbiorem parami różnych punktów. Ile operacji arytmetycznych należy wykonać, aby obliczyć ilorazy różnicowe $f[x_0, x_1, \ldots, x_k]$ dla $k = 0, 1, \ldots, n$?