Programowanie

Egzamin

25 czerwca 2001

W treści zadań podane są liczby punktów (tłustym drukiem w nawiasach), które można uzyskać za ich rozwiązanie. Jeżeli punktacja za zadanie wynosi n, oznacza to, że za to zadanie można otrzymać od $-n$ do n punktów. Punkty ujemne będą przyznawane za umieszczenie w rozwiązaniu odpowiedzi kompromitująco fałszywych. Za brak rozwiązania zadania otrzymuje się 0 punktów. Punktacja za zadania przelicza się na oceny według poniższej tabeli:

$-170 \div 29$	ndst
$30 \div 36$	dst
$37 \div 43$	dst+
$44 \div 50$	db
$51 \div 57$	db+
$58 \div 170$	bdb

Zadanie 1 (10). Rozważmy gramatykę $G = \langle \Sigma, V, S, P \rangle$, gdzie $\Sigma = \{a, b\}$, $V = \{S\}$ i

$$P = \{S \to \epsilon, S \to aSb, S \to aS\}$$

Wykaż (**8**), że językiem generowanym przez tę gramatykę jest

$$L = \{a^n b^m \mid n \geq m \geq 0\}$$

Wykaż (**1**), że gramatyka ta jest niejednoznaczna. Zdefiniuj (**1**) jednoznaczną gramatykę generującą język L.

Definicje do zadań 2–4. Niech $\mathcal{S} = \{obj, list\}$ będzie zbiorem gatunków. Rozważmy sygnaturę Σ nad tym zbiorem zawierającą następujące symbole:

$$\begin{aligned}
T &: obj \\
F &: obj \\
not &: obj \to obj \\
nil &: list \\
cons &: obj \times list \to list \\
hd &: list \to obj \\
tl &: list \to list \\
appn &: list \times list \to list \\
rev &: list \to list \\
inv &: list \to list \\
null &: list \to obj
\end{aligned}$$

Niech

$$\begin{aligned}
B &= \{0, 1\} \\
B^0 &= \{\epsilon\} \\
B^{k+1} &= B \times B^k \\
B^* &= \bigcup_{k \geq 0} B^k
\end{aligned}$$

Termy gatunku *obj* będziemy interpretować w zbiorze B, przypisując stałej F liczbę 0, zaś stałej T liczbę 1. Termy gatunku *list* będziemy interpretować w zbiorze B^* skończonych ciągów zerojedynkowych. Znaczenie symboli sygnatury jest następujące: *not* jest operacją dopełnienia, przypisującą wartości 0 wartość 1 i odwrotnie, *nil* reprezentuje ciąg pusty, *cons* dołącza element do ciągu, *hd* i *tl* ujawniają głowę i ogon ciągu (przy czym przyjmujemy, że głową ciągu pustego jest 0, zaś ogonem ciąg pusty ϵ), *appn* łączy ciągi, *rev* odwraca ciąg, *inv* tworzy ciąg dopełnień elementów podanego ciągu, wartością *null* dla ciągu pustego jest 1, dla niepustego zaś 0.

Zadanie 2 (8). Zadaj (**8**) semantykę denotacyjną powyższego języka, tj. określ interpretację $\cdot^{\mathfrak{M}}$ symboli sygnatury w algebrze $\mathfrak{M} = \langle \{B, B^*\}, \cdot^{\mathfrak{M}} \rangle$. *Wskazówka:* dla przykładu interpretację symbolu *cons* należy zdefiniować następująco: $cons^{\mathfrak{M}}(b, c) = \langle b, c \rangle$.

Zadanie 3 (10). Aby opisać semantykę operacyjną powyższego języka, będziemy rozważać formuły postaci $t \to t'$, gdzie t i t' są termami. Formuła $t \to t'$ oznacza, że wynikiem obliczenia (wartością) termu t jest term t'. Zadaj (**10**) semantykę operacyjną naszego języka, tj. podaj zestaw reguł wnioskowania dla formuł powyższej postaci. *Wskazówka:* dla przykładu jedna z reguł może być następująca:

$$\frac{e_1 \to e_1' \quad e_2 \to e_2'}{cons(e_1, e_2) \to cons(e_1', e_2')}$$

Mówi ona, że aby wyznaczyć wartość termu $cons(e_1, e_2)$, należy wyznaczyć wartości e_1' i e_2' termów e_1 i e_2 i wówczas wartością termu $cons(e_1, e_2)$ jest $cons(e_1', e_2')$.

Zadanie 4 (20). Zdefiniuj (**7**) semantykę algebraiczną powyższego języka, tj. podaj odpowiedni zbiór równości (specyfikację równościową). *Wskazówka:* dla przykładu jedna z równości może mieć postać $hd(cons(x, l)) = x$. Wśród symboli sygnatury wskaż (**1**) konstruktory, destruktory, operatory i obserwatory. Sformułuj (**2**) zasadę indukcji dla gatunku *list*. Korzystając z niej pokaż (**10**), że dla każdej listy l : *list* zachodzi $inv(rev(l)) = rev(inv(l))$.

Zadanie 5 (30).

1. Rozważmy język, w którym można deklarować procedury lokalne (np. Pascal). Jeżeli w treści procedury występuje odwołanie do zmiennej, wówczas należy ustalić jej adres w stosie rekordów aktywacji. Opisz (**10**) dokładnie, ale zwięźle, w jaki sposób kompilator dokonuje tłumaczenia takiego programu i jak przebiega wyznaczanie adresu zmiennej podczas wykonania programu. *Wskazówka:* opisz precyzyjnie, co to jest *link dostępu*, jak jest wyznaczany i jak można z jego pomocą wyznaczyć adres zmiennej nielokalnej.
2. W wielu językach programowania (m.in. w Pascalu) jako parametr można przekazać procedurę. W praktyce jest wówczas przekazywany adres początku kodu danej procedury. W Pascalu można w ten sposób przekazywać jedynie procedury globalne (zadeklarowane na najwyższym poziomie programu, tj. nie wewnątrz innych procedur). Opisz (**7**) tzw. *downward funarg problem*, tj. pokaż, że algorytmu, który opisałeś w punkcie 1 nie można zastosować, jeśli dopuścimy, by można było w ten sposób przekazywać procedury lokalne (zadeklarowane wewnątrz innych procedur). *Wskazówka:* napisz krótki program w Pascalu rozszerzonym o możliwość przekazywania procedur lokalnych jako parametrów i zwięźle uzasadnij, że metoda, którą opisałeś w punkcie 1 jest w przypadku takiego programu niewystarczająca.
3. Powyższy problem można łatwo rozwiązać, jeśli podczas przekazywania parametru przez procedurę będzie się przekazywać nie jeden, lecz dwa adresy: początek kodu procedury i adres pewnego miejsca w stosie rekordów aktywacji. Taka para adresów bywa nazywana *domknięciem funkcji* (*function closure*). Opisz (**7**), w jaki sposób należy zmodyfikować algorytm, który przedstawiłeś w punkcie 1, by prawidłowo przekazywać procedury lokalne jako parametry.

```
begin
    integer procedure P (x, e);
        integer x, e;
        begin
            x := x + 1;
            P := e
        end P;

    integer z;

    z := 2;
    outinteger (P(z, z × z + 1))
end
```

Tablica 1: Program do zadania 6

4. Podaj przykład (**6**), że nawet tak zmodyfikowany algorytm nie działa poprawnie, jeśli dopuścimy, by procedury były przekazywane jako wyniki procedur funkcyjnych (tzw. *upward funarg problem*). Uzasadnij, że gromadzenie rekordów aktywacji na stosie nie jest wówczas możliwe. *Wskazówka:* napisz odpowiednią procedurę w języku, który pozwala na przekazywanie procedur jako wyników (np. w SML-u) i zwięźle uzasadnij, że rekordu aktywacji tej procedury nie można usunąć z chwilą zakończenia jej pracy.

Zadanie 6 (20). Podaj (**6**) definicję przekazywania parametru przez nazwę (*uwaga:* proszę podać formalną definicję, taką, jaka występuje np. w opisie języka Algol 60; nie należy natomiast opisywać, w jaki sposób taki program jest kompilowany). Jaka (**3**) liczba zostanie wypisana przez program w Algolu 60 przedstawiony w tablicy 1 (w którym parametry x i e są przekazywane przez nazwę)? Jaka (**3**) wartość zostanie wypisana, jeśli w nagłówku procedury pojawiłaby się deklaracja **value** wskazująca, że a) parametr x, b) parametr e, c) oba parametry x i e są przekazywane przez wartość? Opisz (**5**), jak parametry przekazywane przez nazwę można zastąpić przez kombinację parametrów przekazywanych przez zmienną i przez procedurę (w sytuacji, gdy dopuszcza się przekazywanie procedur lokalnych). Przetłumacz (**3**) w ten sposób program algolowy z tablicy 1 na Pascal (w którym można przekazywać parametry przez wartość, zmienną i procedurę).

Zadanie 7 (12). Algorytm sortowania list *Mergesort* jest przedstawiony w tablicy 2. Zaprogramuj go a) w SML-u (**6**), w postaci polimorficznej funkcji

```
mrgsort : ('a * 'a -> bool) -> 'a list -> 'a list
```

sortującej listy zgodnie z podaną relacją porządku i b) w Prologu (**6**), w postaci predykatu `mrgsort/2` sortującego listy liczb całkowitych. Spośród predykatów standardowych, wolno korzystać jedynie z `</2` i `>=/2`.

Zadanie 8 (60). Rozważmy jednogatunkową sygnaturę Σ zawierającą stałe `0` i `1` oraz binarny symbol `+`, który będziemy zapisywać infiksowo i zakładać, że wiąże w lewo. Niech zbiorem zmiennych będzie $\mathcal{X} = \{x, y, z, \ldots\}$. Termom ze zbioru $\mathcal{T}(\Sigma, \mathcal{X})$ nadamy znaczenie przy pomocy semantyki algebraicznej. Przypomnijmy, że semantykę algebraiczną określa się podając skończony zbiór równości, które wraz z następującym aksjomatem i regułami wnioskowania

$$\frac{}{e = e}\ (\text{Refl}) \qquad \frac{e_1 = e_2}{e_2 = e_1}\ (\text{Sym}) \qquad \frac{e_1 = e_2 \quad e_2 = e_3}{e_1 = e_3}\ (\text{Trans}) \qquad \frac{e_1 = e_2 \quad e_3 = e_4}{e_1[x/e_3] = e_2[x/e_4]}\ (\text{Mon})$$

Mergesort $(l) =$
Jeśli lista l jest pusta,
to wynikiem jest lista pusta.
W przeciwnym przypadku:
- podziel listę l na dwie części l_1 i l_2 mniej więcej tej samej długości (tak, by $-1 \leq |l_1| - |l_2| \leq 1$);
- posortuj rekurencyjnie listy l_1 i l_2;
- scal posortowane listy.

Tablica 2: Algorytm *Mergesort* do zadania 7

definiuje relację równoważności termów $\sim$, taką, że $e_1 \sim e_2$, wtedy i tylko wtedy, gdy $\vdash e_1 = e_2$, tj. gdy równość $e_1 = e_2$ daje się wyprowadzić (udowodnić) w powyższym systemie wnioskowania. Za model dla naszego języka termów przyjmujemy algebrę $\mathfrak{P} = \langle \mathcal{T}(\Sigma, \emptyset)/_\sim, \cdot^{\mathfrak{P}} \rangle$, gdzie $0^{\mathfrak{P}} = [0]_\sim$, $1^{\mathfrak{P}} = [1]_\sim$ i $+^{\mathfrak{P}}([e_1]_\sim, [e_2]_\sim) = [e_1 + e_2]_\sim$. (Jak wiemy, algebra ta jest algebrą początkową w klasie algebr spełniających podany zbiór równości, choć do pojęcia algbery początkowej nie musimy się w tym miejscu odwoływać.) Udowodnij (**42**), że dla następującego zbioru równości:

$$0 + x = x \tag{1}$$
$$x + 0 = x \tag{2}$$
$$(x + y) + z = x + (y + z) \tag{3}$$

tak zdefiniowana algebra jest izomorficzna z algebrą liczb naturalnych $\mathfrak{N} = \langle \mathbb{N}, \cdot^{\mathfrak{N}} \rangle$, w której $0^{\mathfrak{N}}$ i $1^{\mathfrak{N}}$ są liczbami zero i jeden, zaś $+^{\mathfrak{N}}$ jest operacją dodawania liczb naturalnych (tj. że zbiór równości (1), (2) i (3) definiuje algebrę liczb naturalnych). Opisz (bez formalnych dowodów), jakie algebry definiują zbiory złożone a) (**6**) z równości (2) i (3), b) (**6**) z pojedynczej równości (3) i c) (**6**) z równości (3) i (4), gdzie

$$x + y = y + x \tag{4}$$

Uwaga: proszę wskazać, jakie algebry (np. algebra liczb wymiernych z zerem, jedynką i mnożeniem itp.) są izomorficzne z algebrami zdefiniowanymi przez te zbiory równości. W punkcie a) algebra ta jest dosyć dziwna, w punktach b) i c) natomiast są to algebry dobrze znane w matematyce. (Geneza tego zadania jest bardzo ciekawa. Otóż przyśniło mi się ono wraz z eleganckim rozwiązaniem 17 czerwca nad ranem.)