Egzamin poprawkowy z programowania
wrzesień, 2002

Ocena za egzamin poprawkowy liczona jest zgodnie z zasadą: **dst** za 30 punktów i każde dadatkowe 10 punktów powoduje podniesienie oceny o pół stopnia. Ocenę **bdb** można zatem otrzymać za 70p.

Zadanie 1.(12p) Odpowiedz na następujące pytania:

a) Czy rostrygalny jest następujący problem:
Czy dla danego programu P (w języku D^*) istnieje zmienna, której zawartośc nie zostanie podczas wykonywania programu zmieniona?
b) Niech r oraz s oznaczają dowolne wyrażenia regularne. Czy $L((r^*s^*)^*) = L((s^*r^*)^*)$
c) Niech r będzie dowolnym wyrażeniem regularnym. Czy język $\{(a+b)^*w \mid w^R \in L(r)\}$ jest regularny?
d) Czy gramatyka G zadana przez produkcje $S \to aSS \mid bSS \mid \varepsilon$ jest jednoznaczna?
e) Czy dla gramatyki G z poprzedniego zadania mamy $L(G) = (a+b)^*$?
f) Czy gramatyka $S \to SS+ \mid SS* \mid a$ jest jednoznaczna?

Zadanie 2.(23p)

a) Zamień poniższy programy na działający tak samo program, w których nie ma instrukcji `goto`, a jedynymi strukturami sterującymi są pętla `do while` i instrukcja `if`. (**8p.**)

```
L1: C1;
    while (b1) {
       C2;
       if (b2) goto L1;
       if (b3) break;
       C3;
    }
```

b) Przekształć program

```
int f(int x, int y) {
  if (x>y) return f(x-1,2*y);
  return g(x,y);
}
```

na obliczający to samo program z pętlą `while` nie zawierający rekurencji (`g` jest funkcją zadeklarowaną jako `int g(int,int)`).(**6p.**)

c) Przekształć funkcję

```
int h(int x) {
  int a=0;
  while (x>0) {
    x--;
    if (x % 4 == 0) continue;
    a += d(x);
  }
  return a;
}
```

na funkcję obliczającą to samo, w której jedynymi strukturmi sterującymi jest instrukcja warunkowa oraz rekurencja (funkcja `d` zadeklarowana jest jako `int d(int)`) (**9p.**).

Zadanie 3.(14p) Automat A działający nad alfabetem $\Sigma = \{a, b\}$ ma zbiór stanów Q, funkcję przejścia δ, stan początkowy q_0 oraz zbiór stanów końcowych F.

a) Co to znaczy, że słowo $w \in \Sigma^*$ jest akceptowane przez ten automat (**6p.**) ?

b) Automat zadany jest za pomocą globalnych zmiennych `Delta`, `Start` oraz `F`. Stała `N` to liczba stanów tego automatu. Zmienna globalna `Start` zawiera numer stanu początkowego. Tablica `Delta` opisuje funkcję przejścia. Jeżeli $\delta(a, q_i) = q_j$, to `Delta[0,i] == j`; natomiast jeżeli $\delta(b, q_i) = q_j$, to `Delta[1,i] == j`. `F[i] == true` wtedy i tylko wtedy, gdy $q_i \in F$.
Uzupełnij fragmenty oznaczone `(1)`, `(2)`, ... `(7)` tak, by powstała funkcja `Akceptuje`, która dla zadanego słowa `w` (zawierającego jedynie znaki 'a' i 'b') zwraca wartość logiczną mówiacą, czy to słowo jest akceptowane przez pewien automat.

```
int Delta[2][N];
int Start;
bool F[N];

bool Akceptuje(char *w) {
  int bs=(1);   // Biezacy stan
  int i=0;
  while ( (2) ) {
     if (w[i] == (3) ) bs= (4) ;
     if (w[i] == (5) ) bs= (6);
     i++;
  }
  return (7);
}
```

Każda dobra odpowiedź warta jest 1 punkt, dodatkowe 1p otrzymuje się za poprawne wypełnienie wszystkich pól.

Zadanie 4.(18p) Odpowiedz na następujące pytania (w pytaniach o typ funkcji możliwa jest odpowiedź: „nie ma typu"):
a) Jaka funkcja ma najogólniejszy typ równy `'a -> 'a * ('a list)`?
b) Jaka funkcja ma najogólniejszy typ równy `('a -> 'a) -> 'a list -> 'a` ?
c) Jaki jest najbardziej ogólny unifikator `app([b,c],[c,d],L)` oraz `app([X|Xs],Ys,[X|Zs])`?
d) Jaki najogólniejszy typ ma w SML-u funkcja `f` zdefiniowana jako `fun f x y = (y x, (y,y))`?
e) Jaki najogólniejszy typ ma w SML-u funkcja `f` zdefiniowana jako `fun f x y = [x=y]`?
f) Jaki najogólniejszy typ ma w SML-u funkcja `f` zdefiniowana jako `fun f x y = x y::[y]`?

Zadanie 5.(13)p) Zdefiniuj w Prologu następujšce predykaty:
a) (**6p.**) `duplicate(L1,L2)` oznaczajšcy, że lista `L2` składa się z elementów listy `L1` z których każdy wzięty jest dwukrotnie, np. `duplicate([1,2,3],[1,1,2,2,3,3])`.
b) (**7p.**) `shuffle(L1,L2)` oznaczający, że lista `L2` powstała przez zamienienie elementów na pozycjach parzystych z elementami na pozycjach nieparzystych, przykładowo
`shuffle([1,2, 3,4, 5,6, 7],[2,1, 4,3, 6,5, 7])`.

Zadanie 6.(20p) Zdefiniuj w SML-u funkcję `minus` przekształcającą listę liczb całkowitych na listę liczb przeciwnych (**4p.**). Udowodnij (**16p.**), że dla każdej listy liczb całkowitych `l`

```
rev (minus l) = minus (rev l)
```

Poniżej przypominam definicje potrzebnych predykatów:

```
fun [] @ xs = xs |
    (x::xs)@ys = x::(xs @ ys);
fun rev (x::xs) = (rev xs) @ [x]  |
    rev []      = [];
```