Programowanie 2009 Programming 2009

Lista zadań nr 8 Problem set no. 8

Na zajęcia 28–29 kwietnia 2009 Due April 29, 2009

Grupy zasadnicze Basic groups

Zadanie 1 (1 pkt). *Wyrażenia regularne* mają następujacą składnię abstrakcyjną:

Problem 1 (1 p). *Regular expressions* have the following abstract syntax:

$$e \quad ::= \quad a \mid \epsilon \mid \emptyset \mid e_1 \cdot e_2 \mid e_1 + e_2 \mid e^*$$

gdzie $a \in \Sigma$, Σ jest pewnym alfabetem. Wyrażenia regularne opisują języki nad alfabetem Σ. Wyrażenie a opisuje język złożony z jednoliterowego słowa a, ϵ — język złożony ze słowa pustego, $\emptyset$ — język pusty, $e_1 \cdot e_2$ — język złożony ze słów postaci $w_1 w_2$, gdzie słowo w_i należy języka opisanego przez wyrażenie e_i $(i = 1, 2)$, $e_1 + e_2$ — sumę mnogościową języków opisanych wyrażeniami e_1 i e_2, zaś e^* — język złożony ze słów postaci $w_1 \ldots w_n$, gdzie $n \geq 0$ i w_i należą do języka opisanego wyrażeniem e. Zdefiniuj semantykę denotacyjną tego języka.

where $a \in \Sigma$, Σ is an alphabet. Regular expressions describe languages over alphabet Σ. Expression a stands for the language consisting of a single letter word a, ϵ — the language consisting of the empty word, $\emptyset$ — the empty language, $e_1 \cdot e_2$ — the language consisting of words of the form $w_1 w_2$, where the word w_i belongs to the language described by the expression e_i $(i = 1, 2)$, $e_1 + e_2$ — the set union of the languages described by the expressions e_1 and e_2, and e^* — the language consisting of words of the form $w_1 \ldots w_n$, where $n \geq 0$ and w_i belong to the language described by the expression e. Define the denotational sematics of this language.

Zadanie 2 (1 pkt). Wzorując się na semantyce naturalnej języków programowania zdefiniuj semantykę operacyjną wyrażeń regularnych, tj. zadaj zbiór reguł wnioskowania dla sądów postaci $e \to w$. Sąd $e \to w$ powinien być dowodliwy wtedy i tylko wtedy, gdy $w \in [\![e]\!]$.

Problem 2 (1 p). Following the natural semantics of programming languages define the operational semantics of the regular expressions, *i.e.*, give a set of inference rules for judgements of the form $e \to w$. A judgement $e \to w$ should be provable if and only if $w \in [\![e]\!]$.

Zadanie 3 (1 pkt). Rozważmy następujący język wyrażeń:

Problem 3 (1 p). Consider the following language of expressions:

$$\begin{array}{rcl} e & ::= & n \mid x \mid e_1 \oplus e_2 \mid \texttt{let}\ x = e_1\ \texttt{in}\ e_2 \\ \oplus & ::= & \texttt{+} \mid \texttt{-} \mid \texttt{*} \mid \texttt{/} \end{array}$$

gdzie n jest stałą całkowitą, x — zmienną, zaś wyrażenie `let` ma podobne znaczenie, jak w Haskellu. Zdefiniuj semantykę denotacyjną tego języka.

where n is an integer constant, x — a variable, and the `let` expression has a similar meaning as in Haskell. Define the denotational sematics of this language.

Zadanie 4 (1 pkt). Rozważmy następujący język wyrażeń:

Problem 4 (1 p). Consider the following language of expressions:

$$\begin{array}{rcl} e & ::= & n \mid x \mid e_1 \oplus e_2 \mid f(e_1, \ldots, e_n) \\ f & ::= & \lambda\,(x_1, \ldots, x_n) \to e \\ \oplus & ::= & \texttt{+} \mid \texttt{-} \mid \texttt{*} \mid \texttt{/} \end{array}$$

gdzie n jest stałą całkowitą, x — zmienną, zaś wyrażenie $\lambda\ (x_1, \ldots, x_n) \to e$ ma podobne znaczenie, jak w Haskellu. Zdefiniuj semantykę denotacyjną tego języka.

where n is an integer constant, x — a variable, and the expression $\lambda\ (x_1, \ldots, x_n) \to e$ has a similar meaning as in Haskell. Define the denotational sematics of this language.

Zadanie 5 (1 pkt). Wyrażenia arytmetyczne języka While rozszerzamy o operatory postinkrementacji i postdekrementacji:

Problem 5 (1 p). We extend the arithmetic expressions of the While language by postincrementation and postdecrementation operators:

$$e \quad ::= \quad c \mid X \mid -e \mid e \oplus e \mid X\texttt{++} \mid X\texttt{--}$$

Teraz więc obliczenie wyrażeń może powodować skutki uboczne. Zadaj semantykę denotacyjną tak rozszerzonego języka.

Now a computation of expressions may cause side effects. Define the denotational semantics of such expressions.

Zadanie 6 (1 pkt). Na wykładzie zadaliśmy semantykę denotacyjną fragmentu języka programowania który nie zawierał operacji wejścia/wyjścia. Jeżeli program może się komunikować ze światem zewnętrznym, to stan obliczeń nie jest w pełni opisany przez zawartość pamięci, ale wymaga również uwzględnienia interakcji z otoczeniem. Niech $\mathbb{Z}^0 = \epsilon$, $\mathbb{Z}^{k+1} = \mathbb{Z} \times \mathbb{Z}^k$, $\mathbb{Z}^* = \bigcup_{k=0}^{\infty} \mathbb{Z}^k$ (skończone, być może puste ciągi liczb całkowitych) oraz $\mathbb{Z}^\omega = \mathbb{Z} \times \mathbb{Z}^\omega = \prod_{k=1}^{\infty} \mathbb{Z}$ (nieskończone ciągi liczb całkowitych). Przyjmujemy, że (x, s) oznacza ciąg złożony z liczby x i wszystkich elementów (skończonego lub nie) ciągu s. Każdy niepusty ciag liczb całkowitych można przedstawić w tej postaci. Stan obliczeń z operacjami wejścia/wyjścia można opisać za pomocą krotki $\langle \pi, i, o \rangle \in \mathbb{S} \times \mathbb{Z}^\omega \times \mathbb{Z}^*$, gdzie $\pi : \mathbb{A} \to \mathbb{Z}$ jest stanem pamięci, $i \in \mathbb{Z}^\omega$ reprezentuje nieskończony ciąg danych wejściowych, zaś $o : \mathbb{Z}^*$ reprezentuje skończony ciąg danych wyjściowych wypisanych przez program. Zdefiniuj semantykę denotacyjną języka z wykładu wzbogaconego o instrukcje `read` X oraz `write` e, gdzie X jest identyfikatorem, zaś e wyrażeniem arytmetycznym.

Problem 6 (1 p). During the lecture we defined the semantics of a fragment of a programming language that did not contain any input/oputput operations. If a program can communicate with an outside world, than a state of computation is not fully described by the contents of the memory, but needs the interaction with the outside world to be taken into consideration. Let $\mathbb{Z}^0 = \epsilon$, $\mathbb{Z}^{k+1} = \mathbb{Z} \times \mathbb{Z}^k$, $\mathbb{Z}^* = \bigcup_{k=0}^{\infty} \mathbb{Z}^k$ (finite, possibly empty sequences of integers) and $\mathbb{Z}^\omega = \mathbb{Z} \times \mathbb{Z}^\omega = \prod_{k=1}^{\infty} \mathbb{Z}$ (infinite sequences of integers). Let (x, s) stands for a sequence consisting of x and all elements of a (finite or infinite) sequence s. Every nonempty sequence can be represented in this way. A state of computation with input/output operations can be described by a tuple $\langle \pi, i, o \rangle \in \mathbb{S} \times \mathbb{Z}^\omega \times \mathbb{Z}^*$, $\langle \pi, i, o \rangle \in \mathbb{S} \times \mathbb{Z}^\omega \times \mathbb{Z}^*$, where $\pi : \mathbb{A} \to \mathbb{Z}$ is a memory state, $i \in \mathbb{Z}^\omega$ represents an infinite sequence of input data, and $o : \mathbb{Z}^*$ represents a finite sequence of output data written by a program. Define the denotational semantics of the language from the lecture extended by instructions `read` X and `write` e, where X is an identifier, and e is an arithmetic expression.

Zadanie 7 (1 pkt). Rozważmy program

Problem 7 (1 p). Consider a program

```
p :
p1 :  y := 1;
p2 :  while x <> 0 do
p21 :    y := y * n;
p22 :    x := x - 1;
      done
```

Przypuśćmy, że $\mathtt{x}, \mathtt{n} \in \mathrm{Dom}(\pi)$ oraz $\pi(\mathtt{x}) = 3$. Wyznacz $[\![P]\!]\pi$.

Assume that $\mathtt{x}, \mathtt{n} \in \mathrm{Dom}(\pi)$ and let $\pi(\mathtt{x}) = 3$. Compute $[\![P]\!]\pi$.

Zadanie 8 (1 pkt). Rozważmy program

Problem 8 (1 p). Consider a program

```
p :
p1 :    n := 3;
p2 :    s := 1;
p3 :    while n > 2 do
p31 :      s := 2 * s;
p32 :      n := n - 1;
        done
```

Niech π będzie dowolnym stanem pamięci. Wyznacz $[\![p]\!]\pi$.

Let π be an arbitrary memory state. Compute $[\![p]\!]\pi$.

Grupy rozszerzone

Extended groups

Zadanie 1 (1 pkt). Udowodnij, że $\mathcal{F} = \langle \mathbb{N}^{\subseteq \mathbb{N}}, \subseteq \rangle$ jest porządkiem zupełnym. Udowodnij, że funkcjonał dany wzorem

Problem 1 (1 p). Prove that $\mathcal{F} = \langle \mathbb{N}^{\subseteq \mathbb{N}}, \subseteq \rangle$ is a CPO. Prove that the functional given by the formula

$$\Phi(f)(n) = \begin{cases} 1, & \text{where } n = 0, \\ n \cdot f(n-1), & \text{where } n > 0 \text{ and } n-1 \in \mathrm{Dom}(f), \\ \text{undefined}, & \text{otherwise} \end{cases}$$

jest ciągły na $\mathcal{F}$. Z odpowiedniego twierdzenia o punkcie stałym wywnioskuj, że istnieje najmniejsza funkcja spełniająca równość

is continuous on $\mathcal{F}$. Infer from an appropriate fix point theorem that there exist the smallest function satisfying the equality

$$f(n) = \begin{cases} 1, & \text{where } n = 0, \\ n \cdot f(n-1), & \text{otherwise.} \end{cases}$$

Wyznacz $f_k = \Phi^k(\bot)$ dla $k \in \mathbb{N}$ i zauważ, że silnia faktycznie jest kresem górnym ciągu $(f_k)_{k \in \mathbb{N}}$.

Compute $f_k = \Phi^k(\bot)$ for $k \in \mathbb{N}$ and note, that the factorial is indeed the least upper bound of the sequence $(f_k)_{k \in \mathbb{N}}$.

Zadanie 2 (1 pkt). Powtórz czynności z poprzedniego zadania dla równości

Problem 2 (1 p). Repeat the tasks from the previous problem for the equation

$$f(n) = \begin{cases} 1, & \text{where } n = 0, \\ n \cdot (n-1) \cdot f(n-2), & \text{for } n \in \{2k \mid k \in \mathbb{N}_+\}. \end{cases}$$

Zadanie 3 (1 pkt). Niech X będzie pewnym zbiorem. Udowodnij, że relacja inkluzji jest porządkiem zupełnym na zbiorze $X^{\subseteq X}$ funkcji częściowych o dziedzinie zawartej w zbiorze X.

Niech dla ustalonych $f \in \{\mathsf{T}, \mathsf{F}\}^{\subseteq X}$ oraz $g \in X^{\subseteq X}$ funkcjonał $\Phi : X^{\subseteq X} \to X^{\subseteq X}$ będzie dany wzorem

Problem 3 (1 p). Let X be a set. Prove that the inclusion is a CPO on the set $X^{\subseteq X}$ of partial functions whose domains are subsets of the set X.

Let for fixed $f \in \{\mathsf{T}, \mathsf{F}\}^{\subseteq X}$ and $g \in X^{\subseteq X}$ the functional $\Phi : X^{\subseteq X} \to X^{\subseteq X}$ be given by the formula

$$\Phi(h)(x) \quad = \quad \begin{cases} x, & \text{where } x \in \mathrm{Dom}(f) \text{ i } f(x) = \mathsf{F}, \\ h(g(x)), & \text{where } x \in \mathrm{Dom}(f) \cap \mathrm{Dom}(g),\ g(x) \in \mathrm{Dom}(h) \text{ i } f(x) = \mathsf{T}, \\ \text{undefined}, & \text{w p.p.} \end{cases}$$

(„undefined" oznacza, że dana wartość argumentu x nie należy do dziedziny funkcji h, zatem $\mathrm{Dom}(\Phi(h)) = \{x \in \mathrm{Dom}(f) \mid f(x) = \mathsf{F}\} \cup \{x \in \mathrm{Dom}(f) \cap \mathrm{Dom}(g) \mid f(x) = \mathsf{T} \wedge g(x) \in \mathrm{Dom}(h)\}$). Pokaż, że Φ jest funkcjonałem ciągłym na $\langle X^{\subseteq X}, \subseteq \rangle$. Wywnioskuj stąd, że posiada najmniejszy punkt stały.

("undefined" means that a given value of the argument x does not belong to the domain of the function h, so $\mathrm{Dom}(\Phi(h)) = \{x \in \mathrm{Dom}(f) \mid f(x) = \mathsf{F}\} \cup \{x \in \mathrm{Dom}(f) \cap \mathrm{Dom}(g) \mid f(x) = \mathsf{T} \wedge g(x) \in \mathrm{Dom}(h)\}$). Prove that the functional Φ is continuous on $\langle X^{\subseteq X}, \subseteq \rangle$. Conclude that it possesses the smallest fixpoint.

Niech $\mathbb{A}$ będzie przeliczalnym zbiorem adresów i $\Pi = \mathbb{Z}^{\mathbb{A}}$ będzie przestrzenią stanów pamięci. Rozważmy zbiór $\mathcal{C} = \{f : P \to \Pi \mid P \subseteq \Pi\}$ funkcji częściowych działających na zbiorze Π. Z udowodnionego wyżej faktu wywnioskuj, że relacja inkluzji na zbiorze $\mathcal{C}$ jest porządkiem zupełnym oraz że dla dowolnych $g \in \mathcal{C}$ oraz $f : \Pi \to \{\mathsf{T}, \mathsf{F}\}$ operator $\Phi : \mathcal{C} \to \mathcal{C}$ dany wzorem

Let $\mathbb{A}$ be a countable set of addresses and $\Pi = \mathbb{Z}^{\subseteq \mathbb{A}}$ be the set of memory states. Consider a set $\mathcal{C} = \Pi^{\subseteq \Pi}$ of partial functions acting over the set Π. Infer from the fact proved above that the inclusion relation on $\mathcal{C}$ is a CPO and for arbitrary $g \in \mathcal{C}$ and $f : \Pi \to \{\mathsf{T}, \mathsf{F}\}$ the operator $\Phi : \mathcal{C} \to \mathcal{C}$ given by the formula

$$\Phi(h)(\pi) \quad = \quad \begin{cases} \pi, & \text{where } f(\pi) = \mathsf{F}, \\ h(g(\pi)), & \text{where } \pi \in \mathrm{Dom}(g),\ g(\pi) \in \mathrm{Dom}(h) \text{ and } f(\pi) = \mathsf{T}, \\ \text{undefined}, & \text{otherwise} \end{cases}$$

jest ciągły. Istnieje zatem najmniejszy punkt stały Φ, a więc i równania stałopunktowego definującego semantykę instrukcji `while`. Uzasadnij, że przyjęcie najmniejszego punktu stałego jako znaczenia instrukcji `while` zgadza się z zamierzonym znaczeniem instrukcji `while`, tj. że dziedziną denotacji instrukcji `while` jest zbiór tych stanów pamięci, dla których iteracja pętli się zatrzyma. Czy poza najmniejszym są jakieś inne punkty stałe? Jaką semantykę one definiują? (Proszę się nie martwić, jeżeli odpowiedź na ostatnie pytanie nie będzie zbyt mądra).

is continuous. Therefore there exists the smallest fixpoint of Φ, and hence as well of the fixpoint equality defining the semantics of the `while` instruction. Give an argument that choosing the smallest fixpoint conforms to the intended meaning of the `while` instruction, *i.e.*, that the domain of the denotation of the `while` instruction is the set of those memory states for which the iteration of the loop stops. Are there any other fixpoints? What semantics do they define? (Do not worry if your answer to the last question is not very clever.)

Zadanie 4 (1 pkt). Denotację instrukcji `repeat` p `until` b możemy zdefiniować jako najmniejszą funkcję $[\![$`repeat` p `until` $b]\!]$ spełniającą następującą zależność rekurencyjną:

Problem 4 (1 p). We can define the denotation of the instruction `repeat` p `until` b as the smallest function $[\![$`repeat` p `until` $b]\!]$ satisfying the following recursive equation:

$$[\![\texttt{repeat}\ p\ \texttt{until}\ b]\!]\pi \quad = \quad \begin{cases} [\![p]\!]\pi, & \text{if } \pi \in \mathrm{Dom}([\![p]\!]) \text{ and } [\![b]\!]([\![p]\!]\pi) = \mathsf{T}; \\ [\![\texttt{repeat}\ p\ \texttt{until}\ b]\!]([\![p]\!]\pi), & \text{if } \pi \in \mathrm{Dom}([\![p]\!]), \\ & [\![p]\!]\pi \in \mathrm{Dom}([\![\texttt{repeat}\ p\ \texttt{until}\ b]\!]), \\ & \text{and } [\![b]\!]([\![p]\!]\pi) = \mathsf{F}; \\ \text{undefined}, & \text{otherwise.} \end{cases}$$

Niech Π będzie przestrzenią stanów maszyny. Zdefiniuj funkcjonał $\Phi : \Pi^{\subseteq\Pi} \to \Pi^{\subseteq\Pi}$ o tej własności, że $h \in \Pi^{\subseteq\Pi}$ spełnia podaną wyżej zależność rekurencyjną wtedy i tylko wtedy, gdy jest punktem stałym Φ. Pokaż, że Φ jest ciągły. Wywnioskuj stąd (przytocz odpowiednie twierdzenie omawiane w poprzednim semestrze), że definicja denotacji instrukcji `repeat` jest poprawna.

Let Π be the set of states of the machine. Define a functional $\Phi : \Pi^{\subseteq\Pi} \to \Pi^{\subseteq\Pi}$ such that $h \in \Pi^{\subseteq\Pi}$ satisfies the recursive equation given above if and only if it is a fixpoint of Φ. Prove that Φ is continuous. Conclude (cite an appropriate theorem considered in the previous semester) that the definition of the denotation of the `repeat` instruction is correct.

Zadanie 5 (1 pkt). Rozważmy program

Problem 5 (1 p). Consider a program

```
y := 1;
while x <> 0 do
   y := y * n;
   x := x - 1;
done
```

Udowodnij, że jeśli $\pi(\mathtt{x}) \geq 0$, to

Prove that if $\pi(\mathtt{x}) \geq 0$, then

$$[\![P]\!]\pi(\mathtt{y}) = \pi(\mathtt{n})^{\pi(\mathtt{x})}.$$

Zadanie 6 (1 pkt). *Wyrażenia bezkontekstowe* mają następującą składnię abstrakcyjną:

Problem 6 (1 p). *Context-free expressions* have the following abstract syntax:

$$e \quad ::= \quad x \mid a \mid \epsilon \mid \emptyset \mid e_1 \cdot e_2 \mid e_1 + e_2 \mid \mu x.e$$

Wyrażenie $\mu x.e$ opisuje najmniejszy taki język L, że jest on równy językowi opisanemu wyrażeniem e w którym zmienna x oznacza język L. Np. wyrażenie $\mu x.(\epsilon + 0 \cdot x \cdot 1)$ opisuje język $\{0^n 1^n \mid n \in \mathbb{N}\}$. Zadaj semantykę denotacyjną tego języka. Udowodnij, że jest ona poprawna, tzn. że denotacja wyrażenia $\mu x.e$ zawsze istnieje.

Expression $\mu x.e$ describes the smallest such language L that it is equal to the language described by the expression e where x stands for the language L. *E.g.*, the espression $\mu x.(\epsilon + 0 \cdot x \cdot 1)$ describes the language $\{0^n 1^n \mid n \in \mathbb{N}\}$. Define the denotational sematics of this language. Prove that it is correct, *i.e.*, the denotation of the expression $\mu x.e$ always exists.

Zadanie 7 (1 pkt). Wzorując się na semantyce naturalnej języków programowania zdefinuj semantykę operacyjną zamkniętych wyrażeń bezkontekstowych, tj. zadaj zbiór reguł wnioskowania dla sądów postaci $e \to w$, gdzie e nie zawiera zmiennych wolnych. Sąd $e \to w$ powinien być dowodliwy wtedy i tylko wtedy, gdy $w \in [\![e]\!]$. Udowodnij równoważność semantyki denotacyjnej z poprzedniego zadania i operacyjnej wyrażeń bezkontekstowych.

Problem 7 (1 p). Following the natural semantics of programming languages define the operational semantics of closed context-free expressions, *i.e.*, give a set of inference rules for judgements of the form $e \to w$, where e contains no free variables. A judgement $e \to w$ should be provable if and only if $w \in [\![e]\!]$. Prove the equivallence of the denotational semantics from the previous problem and operational semantics of the contex-free expressions.

Zadanie 8 (1 pkt). Niech Σ będzie alfabetem. *Wyrażenia regularne* $\mathfrak{R}(\Sigma, V)$ nad alfabetem Σ ze zbiorem zmiennych V mają następującą składnią abstrakcyjną:

Problem 8 (1 p). Let Σ be an alphabet. The *regular expressions* $\mathfrak{R}(\Sigma, V)$ over the alphabet Σ and a set of variables V have the following abstract syntax:

$$e \quad ::= \quad X \mid a \mid [a_1 \ldots a_n] \mid \epsilon \mid \emptyset \mid (e_1|e_2) \mid e_1 \cdot e_2 \mid e_1^* \mid e_1^+ \mid e_1?$$

Na $(\mathcal{P}(\Sigma^*))^n$ wprowadzamy porządek $\preccurlyeq$ wzorem

We introduce an ordering $\preccurlyeq$ on $(\mathcal{P}(\Sigma^*))^n$ putting

$$(U_1, \ldots, U_n) \preccurlyeq (V_1, \ldots, V_n) \iff \forall\, i{\in}\{1, \ldots, n\}\ U_i \subseteq V_i.$$

Niech $V = \{X_i\}_{i=1}^n$ będzie zbiorem zmiennych. *Gramatyką EBNF* nazywamy układ równań postaci

Let $V = \{X_i\}_{i=1}^n$ be a set of variables. *An EBNF grammar* is a system of equalities of the form

$$\{X_i = e_i\}_{i=1}^n,$$

gdzie $e_i \in \mathfrak{R}(\Sigma, V)$, $X_i \neq X_j$ dla $i \neq j$ oraz $\bigcup_{i=1}^n \mathrm{FV}(e_i) \subseteq \{X_i\}_{i=1}^n$. Krotką języków definiowanych przez gramatykę EBNF nazywamy najmniejszą (w sensie relacji $\preccurlyeq$) krotkę

where $e_i \in \mathfrak{R}(\Sigma, V)$, $X_i \neq X_j$ for $i \neq j$ and $\bigcup_{i=1}^n \mathrm{FV}(e_i) \subseteq \{X_i\}_{i=1}^n$. The tuple of languages defined by a EBNF grammar is the smallest (with respect to the $\preccurlyeq$ relation) tuple

$$(L_1, \ldots, L_n) \in (\mathcal{P}(\Sigma^*))^n$$

taką, że

such that

$$L_i \quad = \quad [\![e_i]\!]_{[X_i/L_i]}.$$

Udowodnij poprawność powyższej definicji, tj. pokaż, że najmniejsza krotka istnieje dla każdego układu równań.

Prove the correctness of this definition, *i.e.*, show that such a tuple exists for every system of equalities.

Grupa zaawansowana

Advanced group

Zadanie 1 (2 pkt). Do zbioru instrukcji języka While dodajemy następujace instrukcje:

Problem 1 (2 p). We add the following instructions to the set of instructions of the While language:

$$c \quad ::= \quad \ldots \mid l : c \mid \texttt{goto}\ l$$

gdzie l jest zbiorem *etykiet*. Nazwijmy ten język „Goto". Zadaj jego semantykę denotacyjną.

where l is the set of *labels*. Let us call this language "Goto". Define the denotational semantics of it.

Zadanie 2 (2 pkt). Udowodnij, że wyrażenia bezkontekstowe, gramatyki bezkontekstowe i gramatyki EBNF opisują tę samą klasę języków.

Problem 2 (2 p). Prove that context-free expressions, context-free grammars and EBNF grammars describe the same class of languages.