Programowanie w Logice — zadania

Radosław Rowicki

11 kwietnia 2026

Informacje wstępne

1. Tych zadań jest dużo. **Nie trzeba robić wszystkich**. Połowa punktów wystarczy, ale z każdej sekcji poproszę przynajmniej po jednym punkcie lub solidną próbę.
2. W razie wątpliwości, nieścisłości, przemyśleń, bólu egzystencji — pisać na mejla: `radrowicki+www małpa gmail com`.
3. Nie narzucam żadnego języka programowania. Doceniam egzotykę.
4. Rozpatrujemy tylko i wyłącznie dwuwartościową logikę klasyczną.

Teoria

Dowiedz się i wyjaśnij czym się różni logika pierwszego rzędu od logiki drugiego rzędu *(2 pkt)*. Podaj przykład zdania w logice drugiego rzędu, które nie jest zdaniem w logice pierwszego rzędu *(1 pkt)*.

Rachunek zdań

Oceń prawdziwość zdań (z uzasadnieniem):

1. Bruksselka jest smaczna *(0 pkt)*
2. Zdanie
$$\big(a \Rightarrow (b \Rightarrow c)\big) \iff \big((a \land b) \Rightarrow c\big)$$
jest tautologią w logice klasycznej. *(2 pkt)*
3. Zdanie
$$\big((a \lor b) \Rightarrow c\big) \iff \big((a \Rightarrow c) \land (b \Rightarrow c)\big)$$
jest tautologią w logice klasycznej. *(2 pkt)*
4. Zdanie
$$\neg\Big(\big((p \Rightarrow r) \land (q \Rightarrow s) \land (\neg p \lor \neg s)\big) \Rightarrow (\neg p \lor \neg q)\Big)$$
jest spełnialne w logice klasycznej. *(2 pkt)*

Modele logiczne

Dla każdego zdania skonstruuj model, w którym jest ono prawdziwe (lub pokaż, że taki nie istnieje).

1. $\neg\Big(\big(f(x) \Leftrightarrow (f(y) \wedge g(x))\big) \Rightarrow g(y)\Big) \wedge g(x)$ *(2 pkt)*
2. $\forall x.\ \exists y.\ \big(f(x) \wedge \neg(\exists z.\, f(z))\big)\ \vee\ \big(\neg g(y) \Leftrightarrow g(y)\big)$ *(2 pkt)*
3. $\exists y.\ \forall x.\ \big(f(x) \wedge \neg(\exists z.\, f(z))\big)\ \vee\ \big(\neg g(y) \Leftrightarrow g(y)\big)$ *(2 pkt)*
4. $\big(\forall f.\ (\exists x.\, f(x))\big)\ \wedge\ \big(\exists f, g.\ \forall x.\ f(x) \Leftrightarrow \neg g(x)\big)$ *(2 pkt)*

Arytmetyka

Rozpatrujemy model nad zbiorem liczb naturalnych (z zerem) z zadanymi predykatami:

- SUM, taki że $\mathrm{SUM}(A, B, C)$ zachodzi wtw $A + B = C$,
- SUB, taki że $\mathrm{SUB}(A, B, C)$ zachodzi wtw $A - B = C$,
- MUL, taki że $\mathrm{MUL}(A, B, C)$ zachodzi wtw $A \cdot B = C$.

Zdefiniuj formułę w logice drugiego rzędu FAC taką, że $\mathrm{FAC}(N, F)$ zachodzi wtw $N! = F$ (silnia). Formuła nie może być rekurencyjna. Można definiować formuły pomocnicze. *(6 pkt)*

Programowanie

Napisz zadane programy (w dowolnym języku, może być pseudokod). Zasady:

1. Zakaz używania pętli.
2. Zakaz używania `goto`.
3. Zakaz wstawek assemblerowych, które majstrują przy `EIP`/`RIP`.
4. Zakaz *list comprehensions*.
5. Każda zmienna może mieć przypisaną wartość tylko jeden raz w kodzie, tzn. można korzystać tylko ze "stałych". Jeśli wybrany język ich nie wspiera, należy po prostu pilnować, by żadna zmienna nie została nadpisana.
6. Można (…trzeba) korzystać z rekurencji.
7. Nie korzystamy z gotowców w bibliotece standardowej.

Napisz program, który:

1. Policzy n pierwszych z kolei liczb pierwszych (*nie* wszystkich liczb pierwszych mniejszych od n). *(4 pkt)*
2. Podniesie n do potęgi k. Program ma działać w czasie $O(\log k)$. *(4 pkt)*
3. Policzy podłogę z pierwiastka kwadratowego n. *(4 pkt)*
4. Posortuje listę/tablicę liczb. *$O(n^2)$: 4 pkt, $O(n \log n)$: 6 pkt*

Indukcja

Zapomnijmy na razie o całej arytmetyce, jaką znamy. Wyobraźmy sobie, że żyjemy w świecie, gdzie w matematyce znane są jedynie funkcje, zbiory i logika. Jak się łatwo domyślić, nie jest nam tu łatwo. Próbując kupić coś w sklepie zaczynamy tęsknić za cudownym wynalazkiem, jakim są liczby — których w naszym świecie brakuje.

Jako herosi tej wybitnej krainy postanawiamy coś z tym zrobić i uratować ludzkość od wiecznego nieliczenia.

Od czegoś trzeba zacząć — zatem zacznijmy od niczego. Zdefiniujmy sobie **zero** lub 0 jako nic. Niestety to zbyt mało — na świecie bywają różne obiekty, których jedną z charakterystycznych cech jest to, że istnieją. Więc nie jest ich zero. Potrzebujemy zatem jakiegoś narzędzia, by je zliczać. Musimy więc utworzyć pewien zbiór, który nazwiemy majestatycznie **zbiorem liczb naturalnych** i będziemy oznaczać go $\mathbb{N}$.

Określimy teraz 5 aksjomatów, czyli powszechnych prawd przyjętych w ramach definicji naszego zbioru:

1. 0 jest liczbą naturalną.
2. Dla każdej liczby naturalnej n istnieje inna liczba naturalna, która jest jej następnikiem i oznaczymy ją $s(n)$ (btw. s jest funkcją $\mathbb{N} \to \mathbb{N}$).
3. 0 nie jest następnikiem żadnej liczby naturalnej.
4. Jeśli liczby naturalne n i m mają równe następniki, to są sobie równe.
5. **(Aksjomat indukcji)** Jeśli
   - pewna forma zdaniowa (twierdzenie) W jest prawdziwa dla 0,
   - z faktu, że W jest prawdziwe dla n, wynika, iż jest też prawdziwe dla $s(n)$,

   to W jest prawdziwe dla każdej liczby naturalnej.

Super! Mamy liczby! Możemy teraz się nimi bawić: niech $1 = s(0)$, $2 = s(1) = s(s(0))$ itd. Ale wciąż brakuje nam wielu innych ważnych rzeczy. Nie znamy dodawania, mnożenia, odejmowania i masy prawd, które w realnym świecie są oczywiste. Ze zdefiniowaniem dwóch pierwszych chętnie Ci pomogę:

Dodawanie:

$$a + 0 = a \qquad a + s(b) = s(a + b)$$

Mnożenie:

$$a \cdot 0 = 0 \qquad a \cdot s(b) = a \cdot b + a$$

Ale dalej musisz poradzić sobie sam.

Zadania

1. Udowodnij, że $a + 1 = s(a)$. *(0,5 pkt)*
2. Udowodnij, że $0 + a = a$ (to nie jest część definicji dodawania!). *(0,5 pkt)*
3. Udowodnij, że $2 + 2 = 4$. Przyjmijmy $4 = s(s(s(s(0))))$. *(0,5 pkt)*

4. Udowodnij, że $a \cdot 1 = a$. *(0,5 pkt)*

5. Udowodnij, że dodawanie jest łączne i przemienne, tj.

$$(a + b) + c = a + (b + c) \qquad \text{oraz} \qquad a + b = b + a.$$

(2 pkt)

6. Udowodnij, że mnożenie jest łączne i przemienne. *(3 pkt)*

7. Udowodnij, że $(a + b) \cdot c = a \cdot c + b \cdot c$. *(2 pkt)*

Kmina: Jak wyobrażasz sobie definicję liczb całkowitych i wymiernych w naszym systemie? Czy możliwe jest określenie liczb rzeczywistych? Jeśli tak, to w jaki sposób? Jeśli nie, to dlaczego?