Informacja o wykładzie

Wstęp do semantyki języków programowania

W dokumencie sprzed wielu lat definiującym jeden z języków programowania znalazł się następujący fragment:

4.2.3. Semantyka. Instrukcja <*robienia czegoś*> służy do <*robienia czegoś*> wartości wyrażenia . . .

W tym fragmencie między dwoma parami nawiasów < . . . > znalazło się to samo słowo w odpowiedniej formie gramatycznej. Autorzy tej definicji widzieli więc potrzebę wyjaśnienia, do czego służy instrukcja nazywająca się instrukcją <*robienia czegoś*>, ale uznali także, że mogą to wyjaśnić używając wyłącznie nazwy tej instrukcji. Aby ich usprawiedliwić można podać, że zdanie to dotyczyło bardzo prostej instrukcji, doskonale znanej specjalistom, a mianowicie instrukcji przypisania. Przytoczone zdanie miało postać: Instrukcja przypisania służy do przypisywania wartości wyrażenia jednej lub wielu zmiennym. . . Nie wydaje się jednak, aby osoba, ktora po raz pierwszy zetknęła się z instrukcją przypisania, miała szansę zrozumieć działanie tej instrukcji korzystając jedynie z podanej informacji. Zauważmy jeszcze, że sposób wykonywania instrukcji przypisania jest w tej definicji opisywany w języku naturalnym.

Definicja języka programowania jest wykorzystywana zarówno przez autorów programów tłumaczących jak i programistów piszących programy w tym języku. Jeżeli te dwie grupy osób zrozumieją ją inaczej, to poprawny zdaniem programisty program może działać niepoprawnie. Można też wyobrazić sobie sytuację, w której ten sam program działa różnie (poprawnie i niepoprawnie) na różnych komputerach. Niejednoznaczności definicji języka programowania uniemożliwiają także uzasadnianie poprawności programów. Aby uniknąć takich sytuacji i umożliwić precyzyjne weryfikowanie programów próbuje się opisywać na czym polega wykonanie poszczególnych instrukcji języka programowania stosując metody formalne. Taki precyzyjny opis wykonania instrukcji, zwłaszcza formalny, nazywa się semantyką.

Mamy kilka sposobów definiowania semantyk. Każde polecenie jest różnie interpretowane przez różne osoby. Jeżeli wydając polecenie (np. wstukując nazwę programu na klawiaturze komputera) zamierzamy coś uzyskać, np. obliczyć jakąś wartość, to nie jest istotne w jaki sposób (za pomocą jaki czynności) to polecenie zostanie wykonane. Istotne jest tylko, czy obliczyliśmy interesującą nas wartość. Takie podejście prowadzi do semantyk aksjomatycznych. Osoba wykonująca polecenie (np. pisząca program obliczający pewną wartość) zwykle wyobraża sobie kolejne czynności, które pozwolą na wykonanie tego polecenia. Analiza polecenia z tego punktu widzenia prowadzi do semantyk operacyjnych. Jeszcze inaczej polecenie wygląda z punktu widzenia osoby nadzorującej jego wykonanie. Dla tej osoby też nie są istotne czynności wykonywane podczas realizacji polecenia. Ważne są informacje o tym, co jest potrzebne do wykonania polecenia, co zostanie zużyte, gdzie znajdą urządzenia wykorzystywane podczas wykonywania polecenia, a mówiąc bardziej abstrakcyjnie, ważne są zmiany stanu czegoś, co wykonuje polecenie. Możemy wyobrażać sobie, że ten sposób analizowania wykonania polecenia prowadzi do semantyk denotacyjnych.

Poszczególne rodzaje semantyk pozwalają na analizę języków programowania z różnych punktów widzenia i są stosowane do różnych celów. Semantykę operacyjną można wykorzystać podczas pisania programu, a aksjomatyczna może służyc do sprawdzania poprawności programu. Ważna jest więc zgodność poszczególnych rodzajów semantyk.

Na wykładzie ze **wstępu do semantyki języków programowania** zapoznamy się z podstawowymi rodzajami formalnych sematyk na przykładach prostych języków programowania, poznamy metody konstruowania semantyk, techniki dowodowe stosowane podczas korzystania z semantyk. Będziemy także zajmować się problemem zgodności poszczególnych rodzajów semantyk.