Projekt 4: sprawdzanie typów i działania semantyczne

1. Wstęp.

Celem niniejszego projektu jest poprawienie Waszego parsera z ostatniego zadania domowego tak, aby wykonywał pewne działania semantyczne, w szczególności deklaracje procesów i operowanie tablicami symboli, sprawdzanie zgodnosci typowej programów i konstruowanie drzewek syntaktycznych. Zasady operowania deklaracjami i sprawdzania zgodności typów opisane są w specyfikacji języka C- -. Wasz program powinien "po cichu" akceptować poprawne dane wejściowe i zgłaszać błędy w przypadku podania na wejściu danych z błędami.

Tradycyjnie dokumentację do *lex*'a/*flex*'a znajdziecie tutaj:
`http://www.kompilatory.agh.edu.pl/pages/tk-laboratorium/flex.html`
a do *bison*'a tutaj:
`http://www.kompilatory.agh.edu.pl/pages/tk-laboratorium/bison.html`
z kolei specyfikację języka C– tutaj:
`http://www.math.us.edu.pl/~pgladki/teaching/2011-2012/tk-cminusminus.html`

2. Drzewka syntaktyczne

Drzewka syntaktyczne powinny być skonstruowane według zasady obsługiwania jednej funkcji jednocześnie. Drzewko syntaktyczne dla każdej funkcji powinno zaczynać się od węzła wejściowego, po którym następuje drzewko węzłów reprezentujących działania, które mają być podjęte przy wywołaniu danej funkcji. Każdy węzeł powinien mieć pole (opis węzła) opisujące rodzaj obliczeń odpowiadających temu węzłowi i liczbę jego potomków. Na przykład, zdanie typu "if" może tłumaczyć się na węzeł, który, poza polem opisu, ma troje potomków: węzeł "expr" opisujący warunek, który będzie oceniany i dwa węzły "stmt" reprezentujące potomków alternatyw. Węzeł "expr" może zawierać pole opisujące pewien operator przy węźle, a odpowiednia liczba potomków odpowiadać może do podformuł rozważanej formuły. Węzeł odpowiadający zmiennej może zawierać wskaźnik do pozycji w tablicy odpowiadającej tej zmiennej. Dodatkowe informacje o drzekach syntaktycznych do znalezienia w podręczniku Kenneth Louden, "Compiler construction", PWS Publishing Company, 1997.

W tym zadaniu nie będziecie robić nic pożytecznego ze skonstruowanymi przez Was drzewkami: drzewko syntaktyczne funkcji zostanie po prostu odrzucone po skonstruowaniu. W kolejnych projektach zajmiemy się generwanie kodu na podstawie drzewka, analizą przepływu danych i optymizacją. W związku z tym postarajcie się konstruować drzewka w sposób czysty i klarowny tak, aby ułatwić sobie robotę na przyszłość!

3. Wywoływanie programu.

Plik wykonywalny powinien się nazywać *compile* i powinien czytać z pliku *stdin*. Komunikaty o błędach powinny być zapisane do pliku *stderr*.