Projekt 3: LR parser dla MinimL

1. Wstęp.

Celem niniejszego zadania jest poprawienie Waszej zrozumienia parsowania LR poprzez napisanie LR parsera z obsługą błędów dla języka MinimL.

Dobrym źródłem jest klasyczna pozycja *Alfred V. Aho, Ravi Sethi, Jeffrey D. Ullman, Compilers : principles, techniques, and tools, 2nd Edition, Pearson Education, 2007.* Opis gramatyki MinimL można znaleźć tutaj:

`http://www.math.us.edu.pl/~pgladki/teaching/2011-2012/tk2-miniml.htm`

2. Zadanie

Napisać klasę LR parsera dla języka MinimL (klasa MinimLRParser w kompilatorze). Przy pisaniu tej klasy, postarajcie się napisać możliwie najlepiej działającą obsługę błędów, która będzie generowała precyzyjne, czytelne i sensowne komunikaty. Idealne rozwiązanie powinno również próbować wychodzić z błędów.

"Generowanie" komunikatu błędu nie oznacza tu oczywiście jego wyświetlenia, ale zwrócenie wyjątku *CompilerException* zawierającego komunikat. Zapoznajcie się z dokumentacją *CompilerException* aby zobaczyć, jak to należy robić.

Oczywiście napisanie dobrej obsługi błędów jest bardzo trudne, nie należy więc przeszarżować z ambicją. Powiedzmy, że mniej ważne jest to, czy Wasz parser zawsze będzie właściwie rozpoznawał błędy, a bardziej ważne jest to, jak będzie je komunikował. A priori komunikaty typu "syntax error" albo "guru meditation" nie są pożądane, należy dążyć do czegoś w stylu "brak spodziewanego endif". Napisanie sensownych komunikatów świadczy o dobrym zrozumieniu stanów parsera.

Sztuka sensownego parsowania po wykryciu błędu jest jeszcze trudniejsza od sztuki generowania dobrych komentarzy. Posłużcie się literaturą i przypomnijcie sobie główne strategie przy wychodzeniu z błędu.

Oczywiście poza obsługą błędów Wasz parser przede wszystkim powinien być w stanie generować drzewka syntaktyczne, które następnie zostaną użyte przez kompilator. Przy budowie drzewek wygodnie posłużyć się podklasami klasy MinimLParseNode.