universität wien

Ziel der Einheit

Rechner fertigstellen

- Arbeiten der letzten Einheit abschließen (vor allem ';' (Ttype::result) für Ausgabe des Wertes, 'q' (Ttype::quit) für Programmende in main implementieren und Ausgabe eines Inputprompts '>', wenn auf Eingabe gewartet wird).
- Klasse Token_stream implementieren
- Fehlerbehandlung
- Verbesserungen / Erweiterungen

universität wien

Klasse Token_stream implementieren

Verwenden Sie das vordefinierte Interface:

cp /home/Xchange/ue5/token_stream.h .

Implementieren Sie die Klasse in token_stream.cpp.

Benötigt werden folgende Implementierungen:

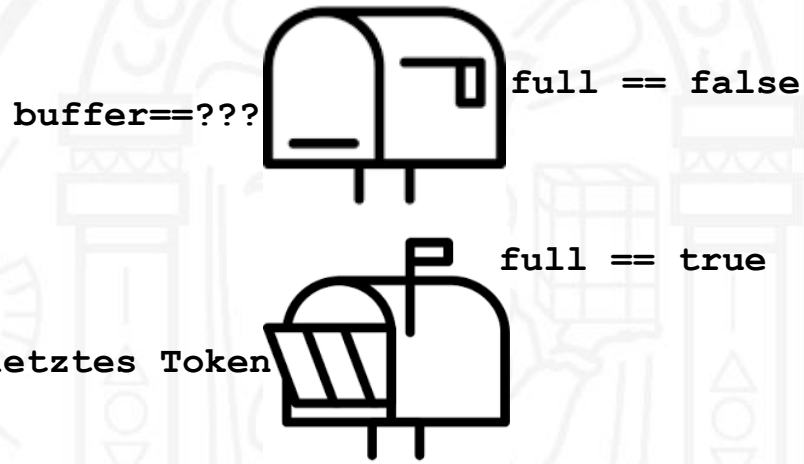
- Konstruktoren für Token (Token(), Token(Ttype), Token(Ttype, double)
- Die Methode putback(Token)
- Die Methode get()

void putback(Token)

Hat die Aufgabe, ein Token, das gelesen aber nicht benötigt wurde, zwischenzuspeichern, so dass es als nächstes Token wieder mit get gelesen werden kann.

Dafür gibt es zwei Instanzvariablen:

- `Token buffer //gepufferter Wert`
- `bool full //Status des Puffers`



Falls schon ein Token gespeichert ist (full == true), dann ist eine Exception zu werfen. Andernfalls wird das als Parameter empfangene Token in buffer gespeichert und der Status full auf true gesetzt.

Token get()

Ist ein Token zwischengespeichert (full == true), dann setze full auf false und retourniere das gespeicherte Token.

Andernfalls wird ein Zeichen aus cin gelesen:

- Ist das Zeichen eine Ziffer ('0' – '9'), dann wird das Zeichen in den Inputstream zurückgestellt (cin.putback(c)), eine Variable vom Typ double gelesen und es wird ein Token vom Typ number mit dem eingelesenen Wert erstellt und retourniert.
- Ist das gelesene Zeichen keine Ziffer, aber eines der gültigen Zeichen (+,-,*, etc.) dann wird aus dem gelesenen Zeichen ein Token vom entsprechenden Typ erzeugt und retourniert.
- Entspricht das gelesene Zeichen weder einer Ziffer, noch einem gültigen Zeichen, so ist eine Exception zu werfen.

universität wien

Fehlerbehandlung

Unser Programm endet, sobald ein Fehler auftritt. Das ist nicht wünschenswert. Besser wäre es, alle Zeichen zu überlesen, bis man wieder irgendwo sinnvoll weitermachen kann (hier z.B. beim nächsten ';').

Es ist klar, dass main nicht von cin einfach Zeichen überlesen sollte (Eventuell wird zum Beispiel irgendwann einmal eine Änderung durchgeführt, dass nicht ';' sondern ein anderes Zeichen für den "result"-Befehl verwendet wird). Auch so lange Tokens zu holen, bis man den "result"-Befehl findet, ist nicht sinnvoll, da ungültige Zeichen in der Eingabe (z.B.: '@') zu immer weiteren Exceptions führen würden.

Der richtige Ort, die Zeichen zu überlesen, ist in der Klasse Token_stream, die sich ja um das Einlesen der Zeichen kümmert. Diese wird um eine Methode **`void ignore(Ttype)`**, die alle Zeichen inklusive des als Parameter übergebenen Stoppsignals überliest. (Prüfen des Puffers von putback nicht vergessen!)

universität wien

Verbesserungen

Umfangreiches Testen des Programms und Beheben eventueller Fehler

Sinnvolle Kommentare im Programm einfügen

cin komplett aus main entfernen

- Dazu kann man Token_stream zu einem "echten" Input-Stream machen, indem man **`operator>>`** überlädt und eine Konversion von Token_stream zu bool implementiert.
- In main kann man dann eine Schleife **`while (ts>>t)`** verwenden analog zur üblichen Schleife mit cin.

Erweiterungen

Negative Zahlen

- Änderung der Grammatik:
  primary = number | '(' expression ')' | ('+' | '-') primary

Variablen / Konstante / vordefinierte Werte (wie e oder π)

Weitere Operatoren (z.B. Quadrat, Quadratwurzel, Fakultät, ++, --, ...)