Testat 4: Zwischencode

Erzeugen Sie mittels der AST-Klassen Zwischencode. Es ist egal ob Sie den Zwischencode aus einem Syntaxbaum oder direkt aus dem Parser heraus erzeugen.

- Aufbau wie Testat 2 und Testat 3!
- Testprogramm (pl0-studi.cpp) ist vorhanden

Listing: (pl0-studi.cpp)

```
#include <stdio.h>
#include <string>
using namespace std;

#include "ast.cpp"
#include "ram.cpp"
ast AST; // You may set reference variable in Your parser!

int yylex();
#include "y.tab.c"
#include "lex.yy.c"



int main(int argc, char * argv[]) {
  int n = yyparse();
  if (n == 0) {
    AST.print();
    AST.interpret();
  }
  return n;
}
```

- Semantische Fehler werden durch Ausführen des Macros YYERROR verarbeitet:

```
%%
  input: 'a' | 'b' {YYERROR;}
  ;
%%
```

- Schnittstelle globaler Variablen zum Scanner:

  Yacc / Bison:

```
%union {
    char * txt;
    // ...
}
```

  Lex / Flex

```
%{
int comm_level = 0;
#define PROCESS(id) {\
  yylval.txt = strdup(yytext);\
  return id;\
}
%}
...
```

```
%%
BEGIN                          PROCESS(t_begin)
...
```

Abgegeben wird eine Datei mit Namen pl0-parser.y oder pl0-parser.cpp.