Lab 1. Part III. Cool overview.

Mipt (Ilab), 24.10.2018

```
$ perl pa1-grading.pl

$ ./lexer grading/all_else_true.cl.cool > out.log

$ vimdiff out.log grading/all_else_true.cl.cool.out

$ gdb --args ./lexer grading/all_else_true.cl.cool
!! utilities.cc
```

++ packages:
Perl, Sed, Awk, Tar, etc.

cool-manual.pdf :

1. Introduction
2. Getting Started
3. **Classes**
4. **Types**
5. **Attributes**
6. **Methods**
7. **Expressions**
8. **Basic Classes**
9. Main Class
10. **Lexical Structure**

PA1.pdf

cool-toor.pdf :

Понадобится на этапе парсера.

lextest.cc

```
int main(int argc, char** argv) {
    int token;
    handle_flags(argc,argv);
    while (optind < argc) {
        fin = fopen(argv[optind], "r");
        // sm: the 'coolc' compiler's file-handling loop resets
        // this counter, so let's make the stand-alone lexer
        // do the same thing
        curr_lineno = 1;
        // Scan and print all tokens.
        cout << "#name \"" << argv[optind] << "\"" << endl;
        while ((token = cool_yylex()) != 0) {
         dump_cool_token(cout, curr_lineno, token, cool_yylval);
        }
        fclose(fin);
        optind++;
    }
    exit(0);
}
```

utilites.cc

```
// dump the token in format readable by the sceond phase token
lexer
void dump_cool_token(ostream& out, int lineno, int token,
YYSTYPE yylval)
{
    out << "#" << lineno << " " << cool_token_to_string(token);
    switch (token) {
    case (STR_CONST):
     out << " \"";
     print_escaped_string(out, cool_yylval.symbol->get_string());
     out << "\"";
#ifdef CHECK_TABLES
     stringtable.lookup_string(cool_yylval.symbol->get_string());
#endif
     break;
...
    }
    out << endl;
}
```

utilites.cc

```
char *cool_token_to_string(int tok)
{
  switch (tok) {
  case 0:             return("EOF");        break;
  case (CLASS):       return("CLASS");      break;
...
  case (ISVOID):      return("ISVOID");     break;
...
  case '}': return("'}'"); break;
  default:  return("<Invalid Token>");
  }
}
```

Cells.cl (клеточный автомат)

```
class Main {
    cells : CellularAutomaton;

    main() : SELF_TYPE {
        {
            cells <- (new CellularAutomaton).init("         X         ");
            cells.print();
            (let countdown : Int <- 20 in
                while 0 < countdown loop
                    {
                        cells.evolve();
                        cells.print();
                        countdown <- countdown - 1;
                    }
                pool
            );
            self;
        }
    };
};
```

- Попеременно показывать cool исходники

- Залезть в репозиторий.
- Запустить пример, показать как работает.

- как flex реагирует на конфликты имен ?
- - flex -d (debug) -T (trace)