IN2030

Løsningsforslag

Uke 35 2023

Oppgave 1

```
private String expandLeadingTabs(String s) {
    String newS = "";
    int n = 0;

    for (int i = 0;  i < s.length();  i++) {
        char c = s.charAt(i);
        if (c == ' ') {
            newS += " ";
            n++;
        } else if (c == '\t') {
            int nReplace = 4 - (n % 4);
            for (int j = 0; j < nReplace; j++) {
                newS += " ";
            }
            n += nReplace;
        } else {
            newS += s.substring(i);
            break;
        }
    }
    return newS;
}
```

Oppgave 2

Oppgave 2a

+ – * / ( ) e-o-f number

Oppgave 2b

(Denne koden finnes også i ~inf2100/e/e1/).

Main.java

```
class Main {
    public static void main(String arg[]) {
        Scanner s = new Scanner(arg[0]);
        while (true) {
            System.out.println("Token: " + s.curToken().showInfo());
            if (s.curToken().kind == TokenKind.eofToken) break;
            s.readNextToken();
        }
    }
}
```

Scanner.java

```
import java.io.*;
import java.util.*;

class Scanner {
    private LineNumberReader sourceFile = null;
    private ArrayList<Token> curLineTokens = new ArrayList<>();

    public Scanner(String fileName) {
        try {
            sourceFile = new LineNumberReader(
                            new InputStreamReader(
                                new FileInputStream(fileName),
                                "UTF-8"));
        } catch (IOException e) { }

        readNextLine();
    }

    public Token curToken() {
        return curLineTokens.get(0);
    }

    public void readNextToken() {
        if (! curLineTokens.isEmpty())
            curLineTokens.remove(0);
    }

    private void readNextLine() {
        String line = null;
        try {
            line = sourceFile.readLine();
        } catch (IOException e) { }

        // Find all the tokens:
        int pos = 0;
        while (pos < line.length()) {
            char c = line.charAt(pos++);

            if (Character.isWhitespace(c)) {
                // Ignore spaces.
            } else if (isDigit(c)) {
                Token t = new Token(TokenKind.integerToken);
                t.integerLit = c - '0';
                curLineTokens.add(t);
            } else if (c == '*') {
                curLineTokens.add(new Token(TokenKind.astToken));
            } else if (c == '(') {
                curLineTokens.add(new Token(TokenKind.leftParToken));
            } else if (c == '-') {
                curLineTokens.add(new Token(TokenKind.minusToken));
            } else if (c == '+') {
                curLineTokens.add(new Token(TokenKind.plusToken));
            } else if (c == ')') {
                curLineTokens.add(new Token(TokenKind.rightParToken));
            } else if (c == '/') {
                curLineTokens.add(new Token(TokenKind.slashToken));
            }
        }
        curLineTokens.add(new Token(TokenKind.eofToken));
    }

    private boolean isDigit(char c) {
        return '0'<=c && c<='9';
    }
}
```

TokenKind.java

```
enum TokenKind {
    integerToken("integer literal"),

    astToken("*"),
    leftParToken("("),
    minusToken("-"),
    plusToken("+"),
    rightParToken(")"),
    slashToken("/"),

    eofToken("E-o-f");

    private String image;

    TokenKind(String im) {
        image = im;
    }

    public String toString() {
        return image;
    }
}
```

Token.java

```
class Token {
    TokenKind kind;
    long integerLit;

    Token(TokenKind k) {
        kind = k;
    }

    String showInfo() {
        String t = kind + " token";
        if (kind == TokenKind.integerToken)
            t += ": " + integerLit;
        return t;
    }
}
```