Nějaké ty hinty pro psaní jednotného kódu

Obecné zásady:

- používat jeden jazyk při psaní kódu (zvolte si který chcete, ale dodržujte jej jak v názvech proměnných, tak i ve funkcích a komentářích)
- vždy používejte výstižné identifikátory proměnných i funkcí (aby bylo poznat k čemu slouží)
- kód vhodně odsazujte tabulátorem do bloků tak, aby bylo zcela jasné, které příkazy platí do kterého bloku (podmínky, cykly, struktury apod.)
- vyhýbejte se používání zkratek alespoň v názvech funkcí, to platí především pro zkratky, u nichž by nebylo na první pohled jasné, k čemu slouží
- kód vhodně komentujte (stručné popisy funkcí, případně některých složitějších konstrukcí)

Práce se soubory:

- soubory pojmenovávejte co nejpřesněji tak, aby bylo na první pohled jasné co obsahují a to pouze malými písmeny, případně s podtržítkem (např. symbol_table.h / stable.h apod.)
- každý soubor by měl obsahovat hlavičku obsahující informace o projektu, datu poslední modifikace a informace o obsahu souboru (krátký popis implementovaných fičur)
- dodržujte prosím UTF-8 kódování u všech zdrojových souborů (u souborů dokumentace asi netřeba)

Identifikátory proměnných, funkcí apod.

- veškeré konstanty a stejně tak globální proměnné by měly být psány velkým písmem, aby se snadno odlišily od běžných proměnných (např. SYNTAX_ERROR apod.)
- identifikátory proměnných a funkcí prosím uvádějte ve tvaru camelCase (např. symbolTable, getToken() apod.)
- pro vnitřní iterátory a čítače cyklů užívejte zavedené názvy proměnných (např. „i“ nebo „j“ pro iterační cyklus apod.)
- začátek těla funkčních bloků uvozujte složenou závorkou již na úrovni deklarace / klíčového slova (viz ukázka)

Neberte to jako žádné dogma, které se musí přesně do puntíku dodržovat, ale řekl bych, že nám to usnadní práci. Připomínky vítány :)

Ukázka kódu aneb Jak by to třebas mohlo vypadat)

```c
/**
 * @file    example.c
 * @brief   IFJ project, team 17
 *
 *          ...
 *
 * @date    2012/10/06
 */

#include <stdio.h>

/**
 * @info    The main function is responsible for ...
 * @param   int - argument count
 * @param   char** - argument vector
 * @return 0 EXIT_SUCCESS, 1 EXIT_FAILURE
 */
int main(int argc, char* argv[]){

    if(token == RIGHT_VINCULUM){

        while((token == getToken()) != LEX_ERROR){
            /*
                ...
            */
        }
    }
    else if(token == LEFT_VINCULUM){

        for(int i = 0; i < 255; i++){
            /*
                ...
            */
        }
    }
    else{
        switch(token){
            case WHILE:
                /*
                    ...
                */

                break;
            default:
                break;
        }
    }

    return 0;
}
```