

System pro automatickou detekci plagiátorství v kurzech programování

Jiří Baroš

2005



Univerzita Tomáše Bati ve Zlíně
Fakulta technologická

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: Jiří BAROŠ
Studijní program: B3902 Inženýrská informatika
Studijní obor: Informační technologie

Téma práce: Systém pro automatickou detekci plagiátorství
v kurzech programování

Zásady pro vypracování:

Vytvořte modul výukového systému www.vyuka.ft.utb.cz, který umožní automatické vyhodnocování originality odevzdávaných úkolů v kurzech programování v jazycích Pascal a C:

1. Systém bude fungovat jako modul do e-learningového systému Moodle, kde studenti odevzdávají své práce. Modul bude odhalovat plagiáty, či nedokonale změněné zdrojové texty, které do systému již někdo odevzdal.
2. Modul bude mít 2 různá uživatelská rozhraní – jedno pro učitele, druhé pro studenty.
 - Rozhraní pro učitele bude umožňovat textové zadání úkolu a upload souborů řešení z předchozích běhů kurzu. Dále bude obsahovat políčka pro definici maximální povolené míry shody u jednotlivých příznaků.
 - Rozhraní pro studenty bude obsahovat formulář pro zaslání souboru s úkolem na server, který jej zpracuje programem pro zjišťování korelace odevzdávaného souboru se soubory, které již byly odevzdány v minulosti.
3. Algoritmus pro výpočet korelace bude ve zdrojových souborech zjišťovat shodu např. v těchto příznacích:
 - shoda struktury
 - shoda posloupnosti klíčových slov.

Rozsah práce:

Rozsah příloh:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

Dokumentace MySQL – <http://dev.mysql.com/doc/>

Dokumentace PHP – <http://php.net/docs.php>

Dokumentace systému Moodle – <http://moodle.org/doc/>

Manuál a help programu Matlab

Vedoucí bakalářské práce:

Ing. Tomáš Dulík

Institut řízení procesů a aplikované informatiky

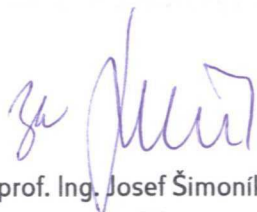
Datum zadání bakalářské práce:

21. února 2005

Termín odevzdání bakalářské práce:

10. června 2005

Ve Zlíně dne 16. února 2005



prof. Ing. Josef Šimoník, CSc.
děkan




prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

Abstrakt

Tato práce se zabývá problémem automatické detekce plagiátorství v kurzech programování, čili vyhledáváním podobných souborů se zdrojovými kódy. Popisuje možnosti detekce, možné řešení tohoto problému, a integraci automatické detekce do online výukového systému Moodle.

Abstract

This bachelor thesis deals with automatic plagiarism detection in programming courses, finding similarities in submitted assignments. Describes possibilities and methods in plagiarism detection, and integration into e-learning system Moodle.

Děkuji vedoucímu práce Ing. Tomáši Dulíkovi za odborné vedení, připomínky a návrhy.

Greerův třetí zákon:

„Počítačový program dělá jen to, co mu řeknete, nikdy však nedělá to, co byste chtěli, aby udělal.“

Prohlašuji, že jsem na celé bakalářské práci pracoval samostatně a použitou literaturu jsem citoval.

Ve Zlíně, 09. 06. 2005

.....

jméno diplomanta

OBSAH

OBSAH	6
ÚVOD.....	8
I. TEORETICKÁ ČÁST	9
1 SYSTÉM PRO DETEKCI PLAGIÁTŮ V KURZECH PROGRAMOVÁNÍ.....	10
1.1 PROBLÉM PLAGIÁTORSTVÍ	10
1.1.1 Zdrojový kód programu.....	10
1.1.2 Co je plagiát?.....	12
1.1.3 Plagiátorství a spolupráce.....	13
1.1.4 Jak vypadá plagiát?	13
1.1.5 Metody hledání plagiátů.....	14
1.1.6 Systémy pro hledání plagiátů	16
1.2 SYSTÉM YAP	17
1.2.1 Jak YAP funguje	17
1.2.2 Tokenizer.....	18
1.2.3 Algoritmus Running Karp-Rabin Greedy String Tiling.....	19
1.3 UTILITY UNIXU	19
1.3.1 BASH	19
1.3.2 Sed.....	20
1.3.3 Tr	21
1.3.4 Awk	22
1.3.5 Ctags.....	22
1.3.6 Regulární výrazy	22
E-LEARNINGOVÝ SYSTÉM MOODLE	23
1.3.7 Popis Moodle	23
1.3.8 Historie Moodle	25
1.3.9 Moodle česky	27
1.3.10 Vlastnosti Moodle	27
1.3.11 Použití Moodle	32
II. PRAKTICKÁ ČÁST	34
2 DETEKCE PLAGIÁTŮ PRO SYSTÉM MOODLE	35
2.1 STANDARDNÍ MODUL ASSIGNMENT (ÚKOL)	35
2.2 MODUL ASSIGNMENT-PD4M	35
2.2.1 Rozhraní studenta.....	38
2.2.2 Rozhraní učitele	41
SEZNAM POUŽITÉ LITERATURY.....	49
SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK	51
SEZNAM OBRÁZKŮ	53
SEZNAM TABULEK.....	54
SEZNAM PŘÍLOH.....	55

ÚVOD

Součástí vzdělávacích kurzů je zadávání úkolů studentům, jejichž cílem je jednak samostatné procvičení studenta v daném problému, posouzení zvládnutí látky a ohodnocení studenta.

Zadávání úkolů sebou ovšem nese problém plagiátorství, ať už jde o zkopírování celého úkolu, nebo využití myšlenky někoho jiného.

Jen málokde je toto jednání jednodušší než u úkolů vypracovaných a odevzdaných pomocí počítače.

Kurzy programování jsou přesně tento případ, studenti odevzdávají k posouzení zdrojové kódy programů, které jsou ve většině případů tvořeny čistým textem, tento text učitel manuálně kontroluje, kromě kontroly funkčnosti a správnosti programu si ovšem musí také všimnout originality zdrojového kódu.

Při tomto přístupu plagiátorovi může stačit ve zkopírovaném zdrojovém kódu změnit formální úpravu, jako například styl odsazování, názvy identifikátorů atd.. Toto může učitele kontrolujícího desítky odevzdaných natolik zmást, že bude odevzdaný zdrojový kód považovat za originální.

Jelikož je zdrojový text programu napsán podle pravidel programovacího jazyka, lze ho analyzovat pomocí počítače podobně jako to dělá kompilátor (program pro překlad zdrojového textu do podoby spustitelného programu), a použít některou z metod pro zjišťování podobnosti mezi texty.

I. TEORETICKÁ ČÁST

1 SYSTÉM PRO DETEKCI PLAGIÁTŮ V KURZECH PROGRAMOVÁNÍ

1.1 Problém plagiátorství

1.1.1 Zdrojový kód programu

Zdrojový kód (někdy též zdrojový text) programu je symbolický zápis algoritmu programu, v syntaxi určitého programovacího jazyka. V naprosté většině případů jde o obyčejný text, ve kterém se operace zapisují podle pravidel daného programovacího jazyka.

Zdrojový text se překládá pomocí **kompilátoru** do binárního tvaru (soubory EXE, ELF...), případně může jít o tzv. **interpretovaný kód**¹.

Zdrojové texty které nás budou zajímat jsou psány v tzv. vyšších jazycích kde se elementární operace procesoru abstrahují do přehlednějších zápisů.

¹ Interpretovaný kód se nepřekládá do binárního tvaru (instrukce procesoru), ale rovnou se vykonává. Např. jazyky Java, Perl, Lisp, TCL.

Příklad zdrojového kódu v jazyce Pascal

```
program ukazka;

{toto je komentar a na funkci programu nemá zadny vliv}

uses Crt;

{deklarace celociselné promenne}
Var promenna:integer;

{uzivatelem nedefinovaná funkce}
Procedure NapisNeco;
Begin
  Writeln('ahoj svete');
End;

{toto je hlavní blok a provede se po spuštění programu}
begin
  {přirazení hodnoty do promenne}
  promenna:=0;

  {následující blok se bude provádět dokud platí podmínka promenna<10}
  repeat

    NapisNeco;

    writeln('promenna ma hodnotu:',promenna);

    promenna:=promenna+1;

  until promenna<10;

  {konec programu}
end.
```

Zdrojový kód programu může obsahovat:

- Nepovinné direktivy (program)
- Použití knihovny (uses)
- Deklaraci proměnných (var), nebo konstant (const), typů (type)
- Deklaraci vlastních procedur nebo funkcí (procedure, function)
- Programové konstrukce
 - Podmínky (if, else)
 - Cykly (for, while, do, repeat)
- Volání funkcí („NapisNeco;“, „Writeln“...)
- Přřazování hodnot („promenna:=1“)
- Aritmetické operace („promenna:=promenna+1“)

- Logické operace („promenna:=promenna or (not bin_hodnota)“)

Identifikátor je název proměnné, funkce, procedury („promenna“, „NapisNeco“, „writeln“...)

Příklad zdrojového kódu v jazyce C (stejný program)

```
#include <stdio.h>

/*toto je komentar a na funkci programu nemá zadny vliv*/

/* deklarace celociselne promenne*/
int promenna=0;

/*uživatelem nadefinovana funkce*/
void NapisNeco(){

    printf("ahoj svete\n");

}

/*toto je hlavni funkce a provede se po spuštění programu*/
int main(){

/*nasledujici blok se bude provádět dokud plati podmínka promenna<10*/
do{

    NapisNeco();

    printf("promenna ma hodnotu:%d\n",promenna);

    promenna++;

}while(promenna<10);

}
```

Syntaxe jazyka C je příbuzná syntaxi jazyka Pascal proto nebude více rozebírána, autor předpokládá že čitatel se v daném směru orientuje.

1.1.2 Co je plagiát?

Alan Parker a James Hamblen, autoři práce Computer Algorithms for Plagiarism Detection (1989), definují softwarový plagiát jako „*program který byl vytvořen na základě jiného programu s malým počtem změn.*“

Pojem „plagiarism“ v encyklopedii Wikipedia²:

„Plagiarism refers to the use of another's ideas, information, language, or writing, when done without proper acknowledgment of the original source.“

„Plagiát je věrná napodobenina díla, u níž je jako původce díla - autor - uveden napodobitel tzv. plagiátor.“

Plagiátorstvím se zabývá například také stránka <http://www.plagiarism.org>.

1.1.3 Plagiátorství a spolupráce

Při zadávání stejného úkolu celé skupině studentů v kurzu se ovšem samozřejmě stane že studenti mezi sebou o problému diskutují a následně použijí podobný nebo stejný algoritmus. Toto je běžné a poučka „nedělat něco co už někdo vymyslel“ je obhájitelná, nicméně v kurzech programování jde o to zvládnout daný programovací jazyk, jeho syntaxi a sémantiku, algoritmizaci v něm.

Jak tedy odlišit zkopírovaný kód od toho který vznikl spoluprací? Lze částečně využít toho , že velká část problémů lze v programovacích jazycích řešit vícero způsoby, použitím jiných funkcí, jiných konstrukcí atd..

1.1.4 Jak vypadá plagiát?

Cílem je vždy nalézt dvojice textů které se nejvíc podobají a seřadit podle míry podobnosti.

Typický případ plagiátu je zdrojový kód kde plagiátor změnil „vzhled“ kódu tím že přejmenoval proměnné a jiné identifikátory, případně udělal další „optické“ úpravy. V takovémto kódu se ovšem nezměnil algoritmus, čili to jak zdrojovému kódu rozumí kompilátor, struktura programu zůstává stejná.

Jako příklad je uveden zdrojový text programu, který vznikl úpravou příkladu v kapitole 1.1.1 - Příklad zdrojového kódu v jazyce C. Komentáře jsou vynechány.

```
#include <stdio.h>
```

² Wikipedia – Otevřená internetová encyklopedie, <http://wikipedia.org/>

```
int var=0;
void funkce() {

    printf("hello world\n");

}

int main()
{

    do
    {

        funkce();

        printf("var has value:%d\n",var);

        var=var+1;

    }while(var<10);

}
```

Porovnejte s příkladem v kapitole 1.1.1 - Příklad zdrojového kódu v jazyce C.

V plagiátu jsou změněny názvy identifikátorů, čili „promenna“ se nyní jmenuje „var“, funkce „NapisNeco“ se jmenuje „funkce“, další změna je v umístění začátků bloků („{“) na nový řádek. Jediná skutečná změna programu je změna „promenna++;“ na „var=var+1“, toto je ale v jazyce C ekvivalentní operace a překladač jim rozumí stejně. Příklad komplexnější změny by byla například záměna cyklu `while` za cyklus `for`.

1.1.5 Metody hledání plagiátů

Metoda počítání atributů (ACS, attribute counting systems).

Počítá se výskyt určitých elementů, následující parametry používá systém Accuse (Grier, „*A tool that detects plagiarism in Pascal systems*“, 1981):

- Počet unikátních operátorů
- Počet unikátních operandů (identifikátorů)
- Celkový počet operátorů
- Celkový počet operandů
- Počet řádků kódu (komentáře a prázdné řádky jsou vynechány)
- Počet proměnných deklarovaných a použitých
- Celkový počet řídicích příkazů

Metoda porovnání struktury (structure metric systems)

Kdybych pracovali pouze s elementy které mají význam pro funkčnost programu, dostaneme pro zdrojový kód v kapitole 1.1.4 následující strukturu:

tabulka 1 – Funkční struktura zdrojového kódu (tokenizace)

Přiřazení	<code>(int var=0;)</code>
Funkce 1	<code>(int main())</code>
vykonej	<code>(do)</code>
Funkce 2	<code>(funkce();)</code>
Vypiš	<code>(printf("hello world\n");)</code>
Vypiš	<code>(printf("var has value:%d\n",var);)</code>
Přiřazení	<code>(var=var+1;)</code>
Konej dokud	<code>(while)</code>
porovnání	<code>((var<10);)</code>

Všimněte si že obsah funkce „funkce()“ je obsažen v posloupnosti až když je funkce zavolána. Kdybychom totéž provedli s příkladem v kapitole 1.1.1 - Příklad zdrojového kódu v jazyce C, dostali bychom stejnou strukturu.

Pro tuto činnost lze velmi dobře využít možností počítače. Popsání struktury programu může být podobné jako když kompilátor překládá zdrojový kód do podoby spustitelného programu. Odtud vyplývá fakt že zkompilevané programy originálního a upraveného zdrojového kódu budou minimálně velmi podobné.

Vytvoření struktury programu (tabulka 1) bude dále označována jako **tokenizace**.

1.1.6 Systémy pro hledání plagiátů

MOSS

Moss (Measure Of Software Similarity,) je automatický systém pro určování podobnosti mezi zdrojovými kódy jazyků C, C++, Java, Pascal, Ada, ML, List a Scheme. Jeho hlavní použití je právě v kurzech programování.

Moss funguje jako internetová služba, pomocí skriptu odešlete na server kde systém běží soubory zdrojových kódů ke zkontrolování. Jakmile Moss kontrolu dokončí odešle vám zpět výsledky. Výsledky jsou v podobě HTML souborů kde jsou uvedeny podobnosti mezi zdrojovými kódy, Moss také v podobných souborech zvýrazní části které se jeví jako individuální (nezkopírované). Moss také automaticky přeskakuje části kódu které jsou součástí knihoven a jsou vždy stejné.

Jak systém Moss pracuje autoři nezveřejnili, nicméně pracuje velmi efektivně.

Více informací v [10].

YAP

Metodu porovnání struktury (viz 1.1.5 - Metoda porovnání struktury) mezi zdrojovými kódy používá systém YAP (viz 1.2).

Je to metoda založená právě na přeložení zdrojového kódu na popis struktury kódu, která reprezentuje algoritmus zkoumaného programu, a vzájemné porovnání struktur. Toto by mělo vést k odhalení zdrojových kódů které používají minimálně velmi podobnou posloupnost operací.

1.2 Systém YAP

„Počítače usnadňují plagiátorství, ale zároveň napomáhají jeho odhalování“, říká Michael J. Wise z Department of Computer Science University of Sydney v [2], autor systému YAP, který je sadou programů pro vyhledávání plagiátů ve zdrojových kódech. Obsahuje podporu pro programovací jazyky Pascal, Lisp a C.

Systém YAP je akademickým projektem tudíž je jeho použití v našich podmínkách možné.

1.2.1 Jak YAP funguje

První část analýzy zdrojového kódu je tzv. tokenizace (viz. 1.1.5 - Metoda porovnání struktury), neboli převedení textu na posloupnost funkčních elementů zdrojového kódu:

- Odstranění komentářů a řetězcových konstant (textů)
- Všechny znaky jsou změněny na malé (lower case, A > a)
- Všechny synonyma jsou převedena na společnou formu³
- Pokud je to možné, funkce nebo procedury jsou rozepsány v pořadí v jakém jsou v programu volány, funkce které již byly rozepsány se znova nevypisují ale označí se jako opětovné volání funkce která již byla volána.
- Všechny elementy které nejsou ve slovníku známých elementů jsou odstraněny

Po tokenizaci máme k dispozici posloupnost, operací které obsahuje zdrojový kód k řešení daného problému. Tato posloupnost prakticky vypadá jako řada čísel nebo slov, přičemž každé číslo nebo slovo vyjadřuje jeden funkční element zdrojového kódu.

Další fáze je již vzájemné porovnávání tokenů (posloupností elementů).

Zde existují tři verze YAPu:

³ V některých programovacích jazycích může mít jedna operace nebo funkce více názvů, např. v jazyce C funkce `strncmp` se nahradí `strcmp`. Případně se za synonymum považuje podobná operace

- YAP1: používá různé běžné utility UNIXu, ovládací skript je napsán v jazyce shellu (BASH, KSH), pro porovnávání se používá utilita `sdiff`. Toto není příliš efektivní řešení. YAP1 je popsán v [2].
- YAP2: Je napsán v jazyce Perl a používá speciálně napsaný program `heckel`, fungující na bázi algoritmu Heckel's algorithm (*Paul Heckel, „A Technique for Isolating Differences Between Files“*).
- YAP3: Používá algoritmus Running Karp-Rabin, Greedy String Tiling (1.2.3, [3],[4]), podrobnější informace o YAP3 v [5].

1.2.2 Tokenizer

Jazyk C

Skript je napsán v jazyce shellu (kornshell).

Syntaxe je:

```
tokenizers/tokenize_C [-2] <c-soubor> <hlavní funkce>
```

Přepínač `-2` určuje zda se má generovat soubor tokenů pro YAP verze 2, pokud přepínač `-2` není uveden generuje se soubor tokenů pro YAP verze 3⁴.

Blokově byla funkce tokenizeru popsána v 1.2.1, uvedu jen příklady některých operací s využitím utilit UNIXu (viz 1.3).

```
sed -e '/^#include/d' -e 's/"[^"]*"//g' -e 's/\'.'///g' < $FILE
```

Ze souboru určeného proměnnou `$FILE` se odstraní všechny řádky `#include`, všechny řetězcové konstanty (sed viz 1.3.2). Výstup příkazu se předá dále pomocí roury (pipe, „|“) na vstup programu `awk` (viz 1.3.4):

```
awk -f $BIN/tokenize_C.comments.awk
```

⁴ YAP2 používá soubory tokenů s textovým vyjádřením elementů (například „if – while – TEST – if – if – =“), zatímco tokeny pro YAP3 jsou tvořeny čísly které reprezentují elementy (definice elementů je v souboru `tokenizers/tokenize_C.awk` pro jazyk C a v `tokenizers/tokenize_PASCAL.awk`)

Odstranění komentářů ve zdrojovém kódu (instrukce jsou uloženy v souboru `tokenize_C.comments.awk`).

```
/usr/ucb/tr 'A-Z' 'a-z' > oo
```

Změna všech znaků na lower-case a uložení výstupu do souboru „oo“.

Další operace:

- Ctags – vypsaní seznamu identifikátorů ze zdrojového textu
- `tokenize_C.sed` – nahrazení některých formulací (např. porovnání „= =“, „<=“, „>“, „!=“ se nahradí obecně „TEST“)
- `tokenize_C.synonyms` – převod synonym na společnou formu, viz poznámka 3.
- Cexpand – rozepsání volaných uživatelských funkcí/procedur
- `tokenize_C.awk` – finální výpis tokenů pro zdrojový kód

1.2.3 Algoritmus Running Karp-Rabin Greedy String Tiling

Metoda Greedy String Tiling je navržena pro určení stupně podobnosti páru řetězců, její výhodou je že je imunní vůči záměně pořadí (transpozici), podřetězců.

1.3 Utility UNIXu

V této kapitole jsou popsány utility které jsou použity v tokenizeru.

1.3.1 BASH

Jako většina shellů (prostředí příkazové řádky) dostupné v Linuxu není bash (Bourne Again Shell) pouze výbornou příkazovou řádkou, ale zároveň také plnohodnotným skriptovacím jazykem. Skriptování v shellu umožňuje zautomatizovat spoustu úloh, které by jinak vyžadovaly psaní spousty příkazů na příkazovou řádku. Mnoho programů v jednotlivých distribucích Linuxu či Unixu jsou pouze skripty příkazové řádky.

Bash patří mezi tzv. interpretované jazyky, to znamená že napsaný program není přeložen do strojového kódu, místo toho interpreter jazyka čte instrukce ze zdrojového kódu a podle obsahu tyto instrukce vykonává. V případě bash spouští programy, které jste napsali do zdrojového kódu. Bohužel, protože interpreter čte každé instrukce jednotlivě a jejich

zpracování není tím pádem úplně nejrychlejší (program je částečně překládán za běhu), jsou tedy skriptovací jazyky výrazně pomalejší než kompilované programy. Hlavní výhoda skriptovacích jazyků spočívá v jejich přenositelnosti na jiné platformy či operační systémy, jejich jednoduchosti a také velikosti.. Příklady dalších interpretovaných jazyků (skriptovacích) jsou Perl, Lisp, Prolog a Tcl .

Skripty v bash jsou jakousi obdobou .bat souborů v DOSu, ale jelikož operační systémy UNIX jsou založeny na ovládání pomocí příkazů, a obsahují velké množství neinteraktivních utilit, lze pomocí sekvencí volání těchto příkazů zautomatizovat opravdu velké množství činností, kvůli kterým bychom jinak museli psát a překládat speciální program v některém z běžných jazyků (C, C++, Pascal).

```
#!/bin/bash
if [ "${1##*.}" = "tar" ]
then
echo Vypada to ze soubor zadany argumentem je archiv TAR.
else
echo Soubor na na první pohled nezda byt archivem TAR.
```

Více [14].

1.3.2 Sed

Sed je dávkový, neinteraktivní, řádkový editor textu. Nepoužívá se tedy jako běžný editor textu (notepad), ale slouží k dávkovému zpracování souboru.

```
sed [-n] -f soubor_příkazů [seznam_souborů]
sed [-n] -e příkazy [seznam_souborů]
```

Přepínač `-n` zamezí sedu cokoli vypisovat, vypisuje se pouze pokud je v souboru instrukcí `print`. Jsou dvě možnosti jak zapsat příkazy které se mají nad textem provést:

- Instrukce pro zpracování uložíme do zvláštního souboru a použijeme přepínač `-f <soubor_příkazů>`
- příkazy zapíšeme jako argument příkazu `sed` za přepínač `-e`

Seznam souborů obsahuje soubory které chceme zpracovat.

Příkaz se skládá z:

```
[adresa[,adresa]] instrukce [argumenty]
```

Adresy nemusíte uvádět sed potom provede instrukci na každou řádku..

Sed pracuje takto:

Pro všechny řádky:

- Načte řádek ze souboru
- Pro všechny příkazy:
 - Vezme první příkaz a jestli adresa platí na soubor provede instrukci

Adresu může být:

- číslo řádku.
- speciální znak ^ - začátek řádku, \$-konec posledního souboru ...
- Regulární výraz, stejný jako u awk. (viz kapitola 1.3.4 a 1.3.6). Je ohraničen lomítky

Instrukce

- d(delete) – mazání řádku
- n(new) – načtení nového řádku
- a(append) – připojení jednoho nebo více řádku k aktuálnímu
- s(substitute) – nahrazení vzorku jiným textem
- další

```
$ sed -e '/^#include/d' ./zdroj.c
```

Tento příkaz odstraní ze souboru zdroj.c všechny řádky začínající #include

Více informací v [15].

1.3.3 Tr

Tr je utilita sloužící k překladu či smazání určitých znaků. Překládá se text ze standardního vstupu (stdin), a výsledek se vypisuje na standardní výstup (stdout).

Použití:

```
$ /usr/bin/tr [jake znaky překládat] [na jake znaky je prekladat]
```

Následujícím příkazem změníme v textu všechny velké znaky (upper case) na malé (lower case)

```
$ /usr/ucb/tr 'A-Z' 'a-z'
```

1.3.4 Awk

Awk je nazýván „jazykem textových manipulací“, je to opravdu přesné vyjádření k čemu awk slouží. Je to jazyk podobný shellu, který je specializovaný pro práci s textovými soubory, nebo výstupy jiných programů.

```
$ awk '/linux/ { print $0 }' soubor.txt
```

Pomocí tohoto příkazu vypíšeme ze souboru soubor.txt jen ty řádky které obsahují slovo „linux“ (princip známého programu grep).

```
$ cat soubor.txt | awk '/linux/ { print $0 }'
```

Tento, naprosto stejně fungující příkaz ukazuje že lze zpracovat i výstup jiného programu.

Více informací v [16].

1.3.5 Ctags

Ctags je utilita která vyextrahuje ze zdrojového kódu všechny identifikátory (názvy proměnných, funkcí..).

1.3.6 Regulární výrazy

Pavel Satrapa v [17] píše: „*Unix bez regulárních výrazů je jako sex bez partnera*“. Regulární výrazy jsou opravdu velmi silným nástrojem.

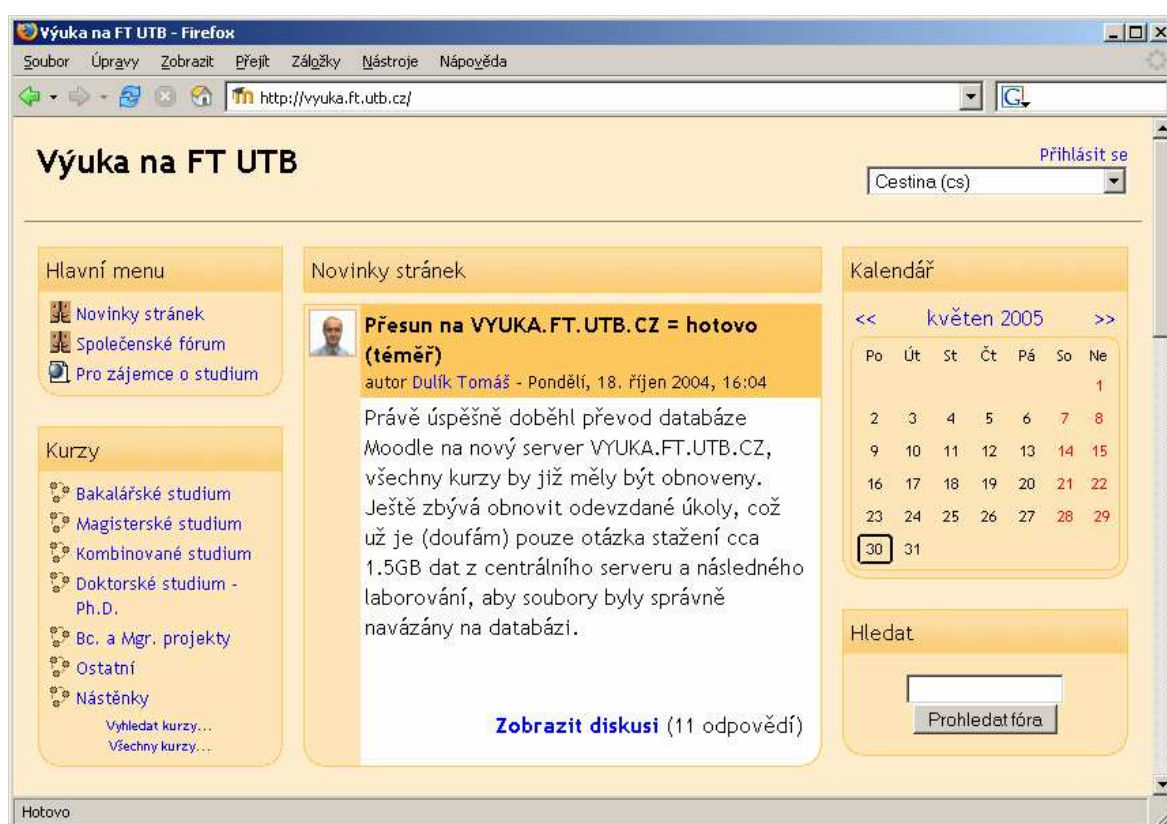
Regulární výrazy (také se označují regex) jsou cestou k popsání textu pomocí odpovídajícího vzoru. Lze je použít k ověření validity dat, vyjímání částí textu z většího celku, nebo nahrazení textu jiným textem.

Syntaxi regulárního výrazu určuje jazyk který pro popis použijeme. Regulární výrazy jsou zahrnuty ve velkém množství programovacím jazyků, skriptovacích jazyků, v editorech textu, databázích a jiných aplikacích.

Více [1], [17].

E-learningový systém Moodle

Moodle je zkratka Modular Object-Oriented Dynamic Learning Environment - Modulární objektově orientované dynamické prostředí pro výuku

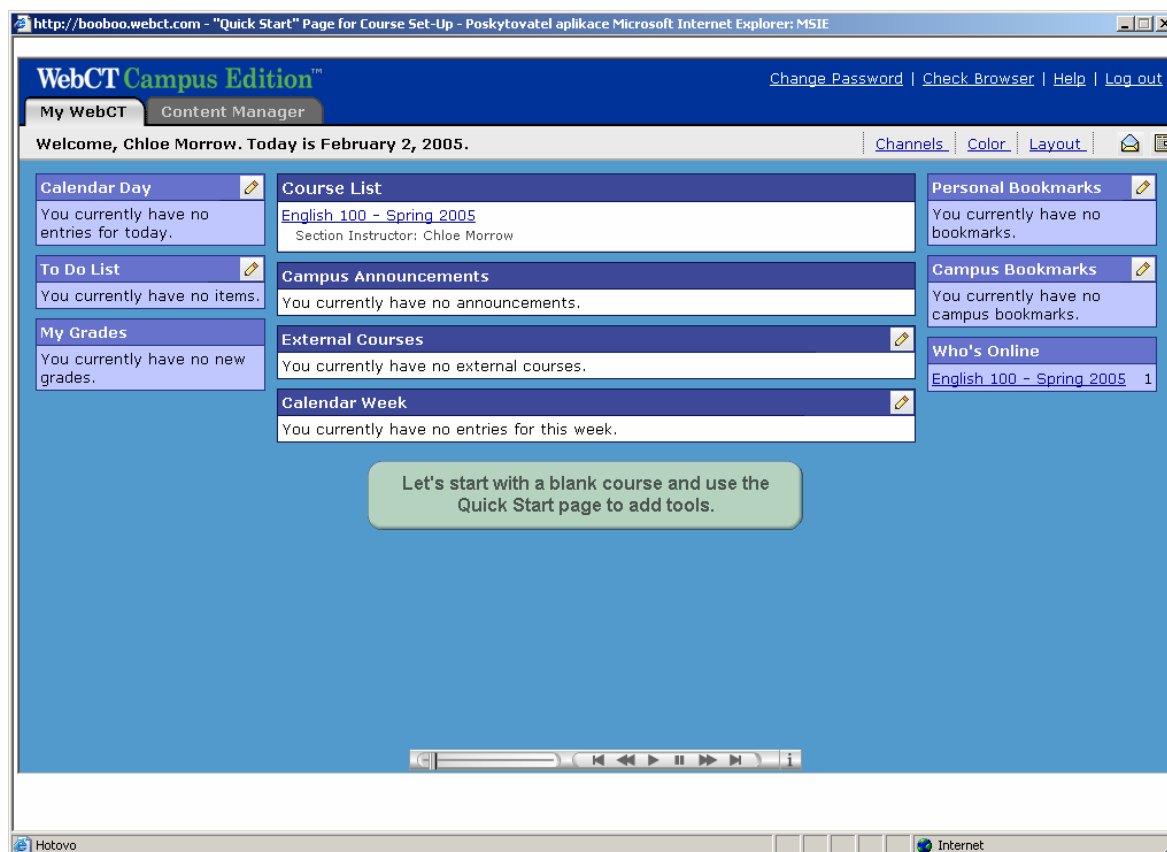


obrázek 1 – Úvodní stránka systému Moodle, používaného na IŘPI – www.vyuka.ft.utb.cz

1.3.7 Popis Moodle

Moodle je softwarový balíček pro tvorbu výukových systémů a elektronických kurzů přístupných přes internet. Jedná se o neustále se vyvíjející projekt, navržený na základě sociálně konstruktivistického přístupu k vzdělávání (viz dále).

Podobných produktů existuje více ať už komerčních (WebCT, BlackBoard, LearningSpace nebo Yahoo!Groups), nebo volně šiřitelných (OpenCourse, CampusGPL, Claroline classroom online, ClassWeb, nebo PHPHomework).



obrázek 2 – systém WebCT – www.webct.com

Moodle je poskytován zdarma jako Open Source software (spadající pod obecnou veřejnou licenci GNU). To v zásadě znamená, že je chráněn autorskými právy, ale poskytuje přitom uživatelům značnou svobodu. Moodle můžete kopírovat, používat i upravovat, pokud souhlasíte s tím, že: budete tento zdroj poskytovat ostatním; nebudete měnit ani odstraňovat původní údaje o licencích a autorských právech, a uplatníte stejné licenční podmínky i u jakýchkoliv odvozených produktů⁵.

⁵ Úplné znění licence: <http://www.gnu.org/licenses/gpl.txt>

Moodle je podobně jako další Open Source projekty vyvíjen decentralizovanou komunitou vyvojářů a spravován pomocí CVS⁶.

1.3.8 Historie Moodle

Jelikož první verze Moodle byla vypuštěna už 20. srpna 2002, dnes již tento systém netrpí „dětskými nemocemi“ ale jde o ověřený a stále se zdokonalující systém.

Vývoj koordinuje Martin Dougiamas z Austrálie, který píše ([11],[12]):

„Zabývám se tím problémem už řadu let. Začalo to v devadesátých letech, kdy jsem byl správcem sítě na Curtin University of Technology, a tedy i správcem tamní instalace výukového systému WebCT. A právě četné frustrace, které mi to přivodilo, mi nasadily brouka do hlavy - to se přece určitě dá dělat lépe! (Ne, tabuli, křídou ani Blackboard jsem tím nemyslel :-)

Znám spoustu lidí ve školách a menších institucích (ale i velkých), kteří by rádi využívali internet efektivněji, ale pro nepřehlednost a množství technologií a pedagogických přístupů nevědí, kde začít. Vždy jsem doufal ve vznik bezplatné alternativy, která by těmto lidem umožnila uplatnit jejich znalosti na síti.

Přesvědčení, že výuka na internetu skýtá velké, ale zatím nevyužité možnosti, mě vedlo k magisterskému a posléze doktorandskému studiu v oboru výchovy a vzdělávání, což mi umožnilo propojit mou dosavadní profesi v oblasti výpočetní techniky s poznatky z pedagogiky. Nejsilněji mě oslovila teorie poznání nazývaná sociální konstruktivismus - která nejenže pojímá studium jako společenskou aktivitu, ale zaměřuje se na získávání dovedností a vědomostí při aktivním vytváření artefaktů (např. textů) určených ke čtení nebo jinému použití dalším lidem.

Je pro mne velmi důležité, aby práce s tímto systémem byla co nejsnazší - usiluji o co největší intuitivnost ovládání.

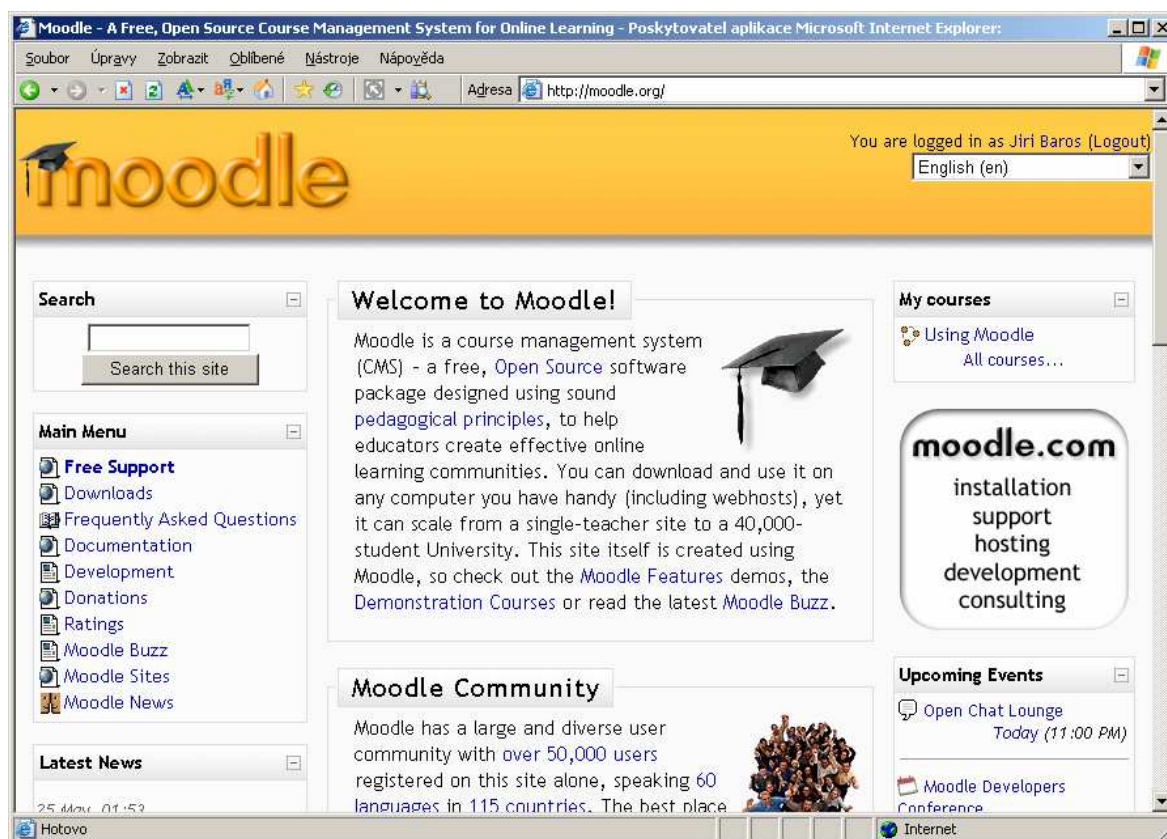
⁶ CVS - Concurrent Versions System, systém pro správu rozsáhlých projektů viz [13]

Jsem odhodlán na Moodlu pracovat dál a zachovat ho otevřený a bezplatný. Hluboce věřím ve význam vzdělání bez omezení a ve výuku s podporou nových technologií, a Moodle je způsob, jak mohu k naplnění těchto tezí nejlépe přispět. “

(www.moodle.cz)

Úspěch Moodle podnítl neustálé zdokonalování systému a také vývoj nových modulů.

Domovská stránka projektu Moodle <http://www.moodle.org>, je důležitým zdrojem informací jak pro uživatele, tak pro vývojáře a také místem pro diskuze.



obrázek 3 – Domovská stránka projektu Moodle, www.moodle.org

Podle vývojářů Moodle bude vždy zdarma, ale pro zákazníky kteří potřebují správu, rozšířenou podporu, konzultantské služby existuje společnost Moodle.com s placenou verzí (<http://moodle.com/>).

Více [11],[12].

1.3.9 Moodle česky

Samozřejmě existuje česká lokalizace o kterou se stará David Mudrák z Katedry informačních technologií a technické výchovy Pedagogické fakulty Univerzity Karlovy v Praze. Domovská stránka české lokalizace je <http://www.moodle.cz>.

1.3.10 Vlastnosti Moodle

Základní vlastnosti

- Podporuje sociálně konstruktivistickou pedagogiku (spolupráce, aktivita, kritická sebereflexe aj.)
- Je vhodný pro plně distanční internetovou výuku i jako doplněk kontaktní výuky
- Jednoduché, efektivní, široce kompatibilní, technicky nenáročné a intuitivní uživatelské rozhraní
- Snadná instalace na téměř všechny platformy, které podporují PHP. Vyžaduje pouze jednu databázi (a tu může sdílet)
- Plně nezávislý na konkrétní databázi; podporuje všechny hlavní typy databází (kromě úvodního vytvoření tabulek)
- Seznam kurzů nabízí popis každého kurzu i informaci, zda do něj mají přístup návštěvníci
- Kurzy lze třídit do kategorií, kategorie lze prohledávat - každý server s Moodleem může podporovat tisíce kurzů
- Velký důraz na zabezpečení: data ze všech formulářů jsou kontrolována, cookies jsou šifrovány atd.
- Většinu oblastí pro vkládání textu (zdroje, příspěvky do fór, záznamy do deníku atd.) lze editovat pomocí vestavěného WYSIWYG editoru HTML

Správa systému

- Systém spravuje administrátor, který je určen během instalace

- Doplnkový modul Vzhled umožňuje administrátorovi nastavit barvy, písma a rozložení stránek tak, aby vyhovovaly místním potřebám
- K stávajícím instalacím Moodlu lze přidávat doplňkové moduly činností
- Doplňkové moduly jazyků umožňují plnou lokalizaci do jakéhokoliv jazyka. Jazykové balíčky lze upravovat pomocí vestavěného webového editoru. V současné době existují balíčky pro více než 34 jazyků
- Kód je přehledně napsán v jazyce PHP a je distribuován na základě licence GPL, takže jej lze snadno upravovat podle potřeb uživatelů

Správa uživatelů

- Cílem je omezit nutnost zásahů administrátora na minimum a přitom zachovat vysoký standard zabezpečení
- Díky doplňkovým modulům podporuje řadu ověřovacích mechanismů, které umožňují snadnou integraci do stávajících systémů
- Standardní e-mailová metoda: studenti si mohou vytvářet své vlastní účty. Uvedou při tom e-mailovou adresu, která se ověřuje potvrzením po obdržení e-mailu
- Metoda LDAP: přihlašování se kontroluje proti serveru LDAP. Administrátor může upřesnit, která pole při tom budou použita
- IMAP, POP3, NNTP: přihlašování se kontroluje proti e-mailovému nebo newsovému serveru. Podpora certifikátů SSL a TLS
- Externí databáze: jako zdroj pro autentizaci může být použita jakákoliv databáze, která obsahuje alespoň dvě pole
- Každá osoba potřebuje pro celý systém pouze jeden účet; pro různé účely lze účtu přiřadit různá práva
- Administrátor řídí zakládání kurzů; učitelem kurzu může stanovit libovolného uživatele
- Administrátor může stanovit tvůrce kurzů; ten je pak oprávněn vytvářet kurzy a určovat pro ně učitele
- Učitelé lze odebrat práva na editování kurzů (například u externích učitelů)

- Učitel může pro každý kurz stanovit "klíč k zápisu", aby do něj měli přístup pouze oprávnění studenti. Tento klíč jim pak sdělí osobně, soukromým e-mailem apod.
- V případě potřeby mohou učitelé zapsat studenty do kurzu také ručně
- Učitelé mohou studenta také ručně odhlásit. Jinak je student odhlášen automaticky, pokud po určitou dobu, kterou nastaví administrátor, nevyvíjí žádnou činnost
- Studenti jsou vedeni k tomu, aby si v systému vytvořili svůj osobní profil obsahující fotografii a charakteristiku. Pokud si to přejí, mohou zakázat veřejné zobrazování své e-mailové adresy
- Každý uživatel si může sám zvolit jazyk uživatelského rozhraní Moodle.

Správa kurzů

- Každý učitel s právem editace má plnou kontrolu nad nastavením kurzu, včetně práva omezovat ostatní učitele
- Volba mezi týdenním, tématickým nebo diskusním uspořádáním kurzu
- Široká nabídka možných činností v kurzu: fóra, deníky, testy, materiály, hlasování, dotazníky, úkoly, chat, workshop
- Na domovské stránce kurzu se mohou zobrazovat změny provedené od posledního přihlášení - posiluje se tím pocit sounáležitosti
- Většinu oblastí pro vkládání textu (materiály, příspěvky do fór, záznamy do deníku atd.) lze editovat pomocí vestavěného WYSIWYG editoru HTML
- Všechna hodnocení z fór, deníků, testů a úkolů mohou být zobrazena na jedné stránce (případně uložena jako soubor pro zpracování v tabulkovém procesoru)
- Rozsáhlé možnosti sledování a zaznamenávání činnosti uživatelů - podrobný záznam a grafy činnosti každého studenta v libovolném modulu (poslední přístup, počet čtení) a také přehledná "historie" studenta v kurzu na jediné stránce, tj. záznam o všech jeho činnostech včetně zápisů do deníku, přispívání do fór atd.
- Vestavěný e-mailový klient - kopie příspěvků do fóra, zpětná vazba atd. mohou být zaslány jako zpráva ve formátu HTML nebo jako prostý text

- Vlastní škály - učitelé si mohou definovat vlastní škály pro hodnocení fór, úkolů a deníků
- Pomocí funkce Zálohování je možné celý kurz sbalit do jediného souboru ve formátu ZIP. Z tohoto souboru jej pak lze zase obnovit na libovolném serveru provozujícím Moodle

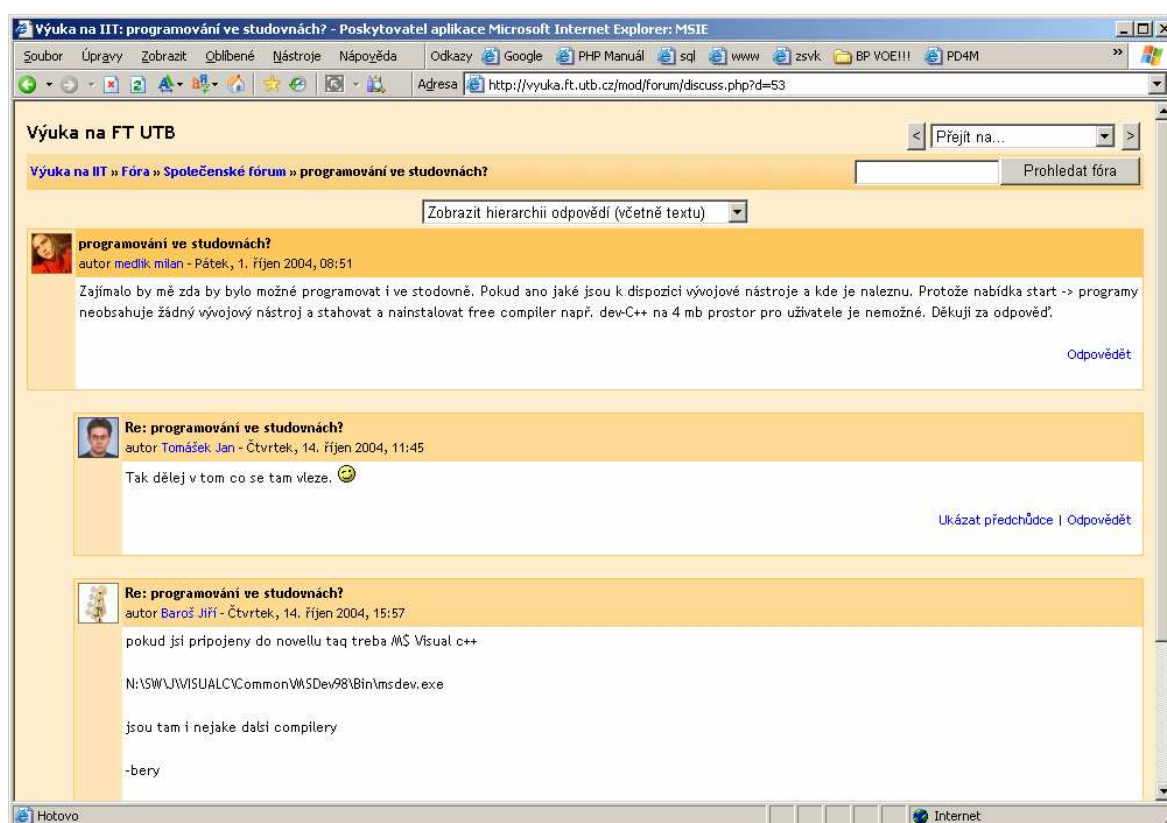
Modul Úkoly

- U úkolů lze stanovit termín odevzdání a maximální počet bodů
- Studenti mohou úkoly nahrát na server (jako soubor v libovolném formátu); každý odevzdaný úkol je při tom označen časovým razítkem
- Opožděné odevzdání úkolu je přípustné; učiteli se však zřetelně zobrazí, s jakým zpožděním byl úkol odevzdán
- Hodnocení úkolu i s komentářem lze vyplnit pro celou třídu na jediné stránce prostřednictvím jediného formuláře
- Hodnocení je studentovi přidáno na stránku s odevzdaným úkolem a zároveň je mu e-mailem zasláno upozornění
- Učitel si může zvolit, jestli je úkol po ohodnocení možné odevzdat znovu (k novému ohodnocení)

Modul Fórum

- K dispozici jsou různé typy fór, např. učitelské, aktuální zprávy z kurzu, veřejné fórum nebo fórum umožňující každému uživateli založit pouze jedno téma diskuse
- U všech příspěvků se zobrazují fotografie jejich autorů
- Různé typy zobrazení fór: lineárně řazené příspěvky (vzestupně či sestupně podle data vložení), hierarchicky řazené příspěvky nebo hierarchicky řazené názvy příspěvků
- Uživatelé mohou pro každé fórum určit, zda jim mají být nové příspěvky zasílány e-mailem. Učitel může provést nucené přihlášení všech účastníků

- Učitel může zakázat odpovídat na příspěvky, například v případě fór sloužících jen jako oznámení
- Jednotlivé diskuse může učitel snadno přemísťovat z jednoho fóra do druhého
- Přiložené obrázky se zobrazují přímo v textu příspěvku
- Je-li používáno hodnocení diskusních příspěvků, lze je omezit jen na příspěvky vložené v určitém časovém rozmezí



obrázek 4 – Systém Moodle – modul Fórum (www.vyuka.ft.utb.cz)

Další moduly

- Modul Chat
- Modul Hlasování
- Modul Deník
- Modul Test
- Modul Studijní materiály

- Modul Dotazníky
- Modul Workshop (Umožňuje vzájemné hodnocení odevzdaných dokumentů všemi účastníky kurzu)

Pro bližší seznámení se systémem Moodle doporučuji stránku [12] – Manuál Moodle

<http://moodle.cz/moodle/doc/> - Příručka učitele.

1.3.11 Použití Moodle

Moodle lze použít na jakémkoliv počítači s fungujícím PHP, podporuje řadu typů databází (především MySQL).

Požadavky

- PHP 4.1.0 nebo vyšší
 - Knihovna GD s podporou formátů JPEG a PNG
 - Podpora sessions
 - Upload souborů
 - Vypnutý „safe mode“
- Databázový server MySQL nebo PostgreSQL (ale lze i jiné)

Instalace v systému Linux

Po stažení a rozbalení příslušných balíčků z <http://moodle.org/>, je nutno tyto zkopírovat do adresáře webového serveru (např. htdocs), nebo do libovolného podadresáře.

Kromě tohoto adresáře kde je samotný systém Moodle je ještě potřeba vytvořit datový adresář kde se budou ukládat například nahrané soubory, studijní materiály, odevzdané úkoly atd.. Je vhodnější aby tento adresář z důvodu bezpečnosti nebyl v adresáři webového serveru. Samozřejmě je třeba zajistit aby proces webového serveru měl právo zápisu do tohoto adresáře (obvykle uživatel apache).

Dále musíme vytvořit databázi v databázovém serveru, a vytvořit uživatelský účet pro Moodle:

```
$ mysql -u root -p

> CREATE DATABASE moodle;
> GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,DROP,INDEX,ALTER ON
moodle.* TO moodleuser@localhost IDENTIFIED BY 'nejakeheslo';

> quit

$ mysqladmin -p reload
```

Tento postup je platný pokud databázový server je MySQL a běží na stejném počítači jako webový server.

II. PRAKTICKÁ ČÁST

2 DETEKCE PLAGIÁTŮ PRO SYSTÉM MOODLE

Systém je zpracován jako modul do systému Moodle (0), a je odvozen od stávajícího modulu Assignment (Úkol), který je součástí systému Moodle, viz 1.3.10 - Modul Úkoly.

Pro orientaci v této části je dobré mít alespoň částečný přehled v používání systému Moodle, protože popis ovládání by zvětšil rozsah této práce na dvojnásobek. Doporučuji [11], [12] Příručka učitele, Příručka vývojáře.

2.1 Standardní modul Assignment (Úkol)

Skripty tohoto modulu jsou uloženy v *[adresář s instalací Moodle]/mod/assignment*.

Tento adresář obsahuje následující soubory:

- *backuplib.php* – provádí zálohu a obnovení dat modulu
- *config.html* – HTML formulář, pro nastavení modulu
- *icon.gif* – ikona která se zobrazí v kurzu u zadaného úkolu
- *index.php* – zobrazí všechny úkoly v kurzu
- *lib.php* – funkce pro obsluhu operací s úkoly
- *mod.html* – formulář pro nastavení úkolu
- *restorelib.php* – provádí zálohu a obnovení dat modulu
- *submissions.php* – zobrazí učiteli odevzdané úkoly
- *upload.php* – obsluha nahrávání (odevzdávání) úkolů na server
- *version.php* – definuje verzi modulu a požadovanou verzi systému Moodle
- *view.php* – zobrazí úkol

2.2 Modul Assignment-PD4M

Toto je modifikovaný standardní modul Úkol, s implementovanými funkcemi pro hledání plagiátů v kurzech programování s využitím systému YAP (viz. 1.2).

Upravený modul obsahuje všechny soubory jako původní a navíc soubory *pd4m.php*, *pd4m.make.php*:

pd4m.php

Tento soubor obsahuje všechny zásadní funkce pro hledání plagiátů:

- `pd4m_get_similarities($assignment, $user)`
vrátí informace o podobnostech úkolu `$assignment` pro studenta určeného argumentem `$user`.
- `pd4m_print_similarities($assignment, $user, $printbr=true)`
provede výpis informací o podobnosti (volá funkci `pd4m_get_similarities`).
- `pd4m_check_tokens_created($submission)`
zkontroluje jestli existuje pro odevzdaný úkol soubor tokenů.
- `pd4m_delete_tokens($submission)`
smaže soubor tokenů pro odevzdaného úkolu.
- `pd4m_create_tokens($submission, $assignment, $newfile_name)`
vytvoří soubor tokenů pro odevzdaný úkol, volá se v souboru *upload.php* při odevzdávání úkolu.
- `pd4m_print_makealltokens_button($assignment)`
tlačítko „Vytvořit soubory tokenů pro všechny odevzdané úkoly“ v hodnocení odevzdaných úkolů.
- `pd4m_print_makesummary_button($assignment)`
tlačítko „Vytvořit Spustit hledání plagiátů“ v hodnocení odevzdaných úkolů.
- `MakeTokens($assignment)`
Vytvoří soubory tokenů pro všechny odevzdané soubory.
- `ExecYap($assignment)`
Provede hledání podobností.
- `LoadYapSummary($assignment)`
Načte výsledky hledání podobností.

V souboru je pro správnou funkčnost také nastavit tyto proměnné:

- `$PD4MCFG->newpath="/pub/yap/tmp_tokens";`
Adresář kde se budou ukládat soubory tokenů odevzdaných úkolů, pro snadnou identifikaci se ukládají soubory do podadresáře jehož název tvoří ID úkolu, a název

souboru tokenů je ID studenta (například soubor `/pub/yap/tmp_tokens/5/3.tokens` bude patřit studentovi s ID 3 a jde o úkol 5).

- `$PD4MCFG->tokenizer_c="/pub/yap/tokenize_C";`
Skript který provádí tokenizaci zdrojového textu v jazyce C.
- `$PD4MCFG->tokenizer_pas="/pub/yap/tokenizers/tokenize_PASCAL2";`
Skript který provádí tokenizaci zdrojového textu v jazyce Pascal.
- `$PD4MCFG->simchecker="/pub/yap/yap3/check2";`
Skript pro určení míry shody pro všechny dvojice soubory tokenů, výsledek je uložen do souboru `yap.summary` v adresáři s tokeny pro daný úkol.

Skripty YAPu

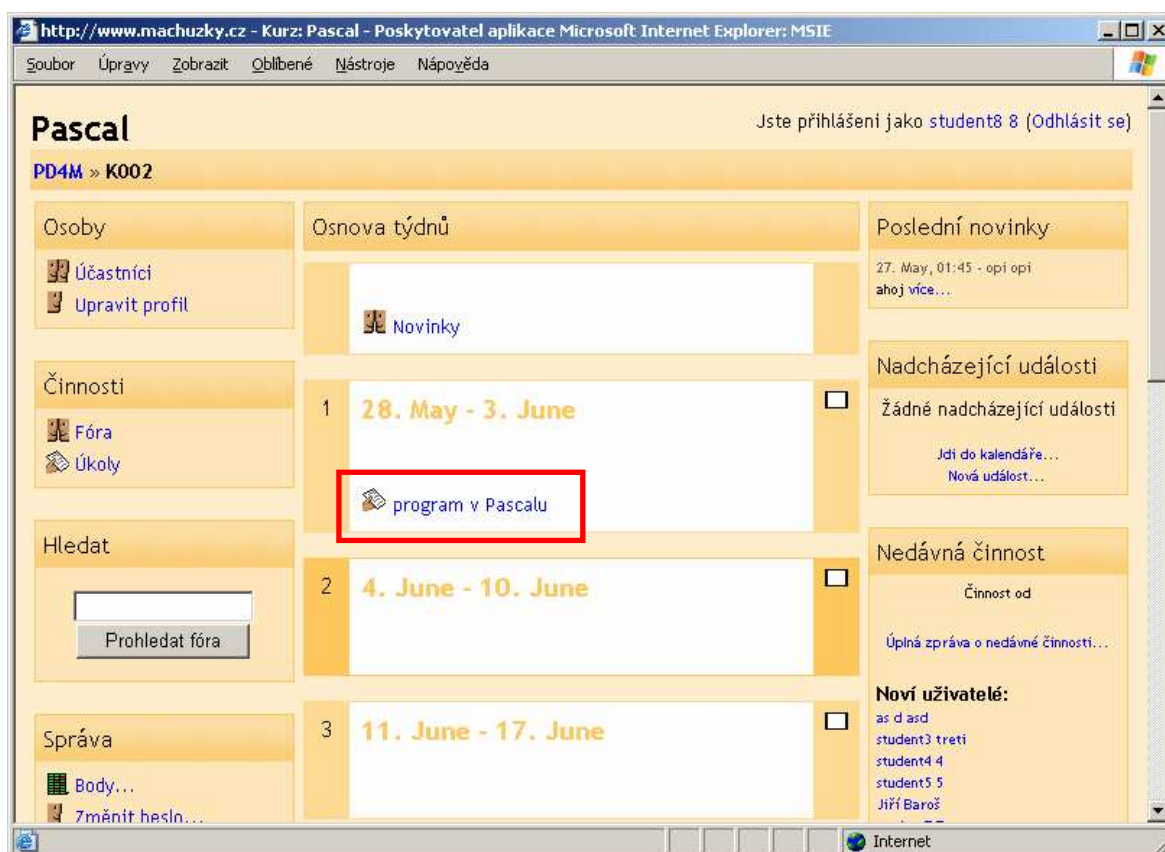
Tyto skripty používá upravený modul Assignment.

- *tokenizers/tokenize_C* – skript pro vytvoření souboru tokenů zdrojového kódu jazyka C
- *tokenizers/tokenize_PASCAL2* – skript pro vytvoření souboru tokenů zdrojového kódu jazyka Pascal, (tokenizery viz. 1.2.2)
- *yap3/check2* – provádí porovnání souborů tokenů, každý s každým (viz. 1.2.3)

2.2.1 Rozhraní studenta

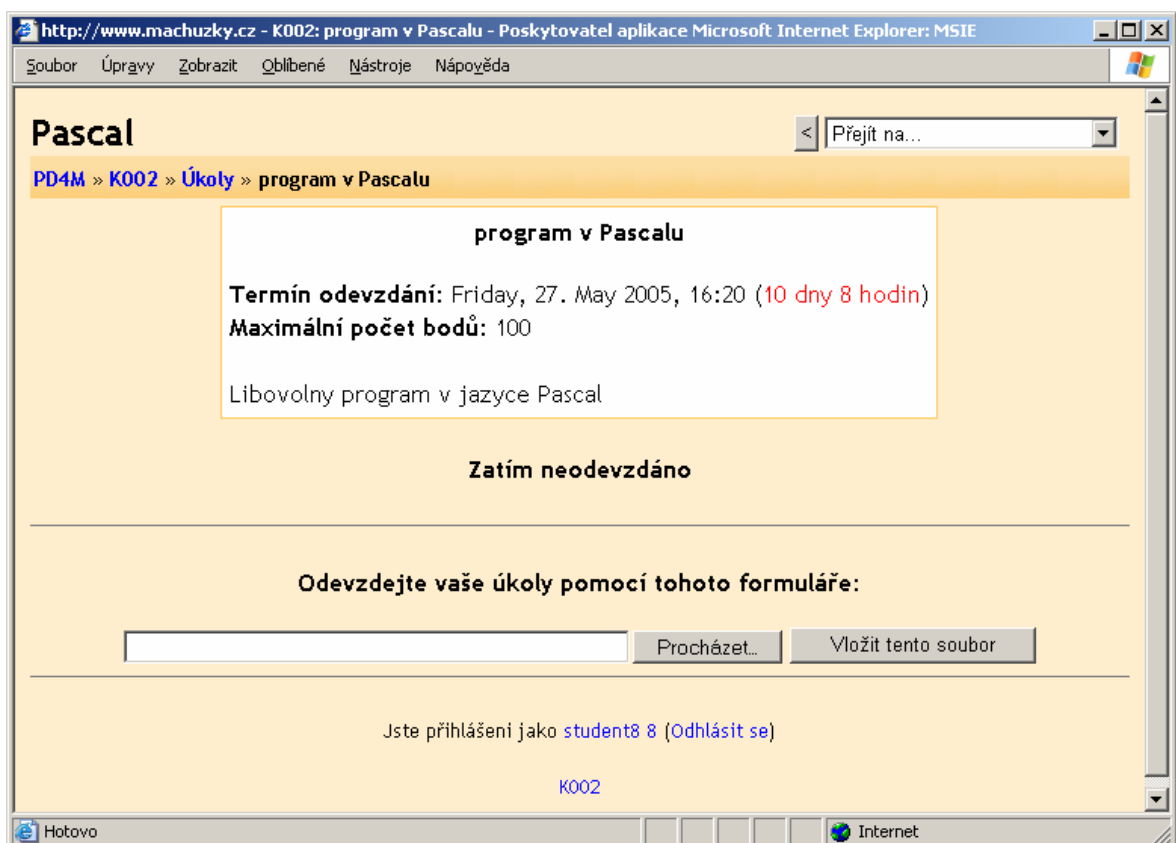
Odevzdání úkolu:

Student po zapsání do kurzu vidí osnovu týdnů a činnosti v daných týdnech. Na následujícím obrázku je zvýrazněn zadaný úkol.



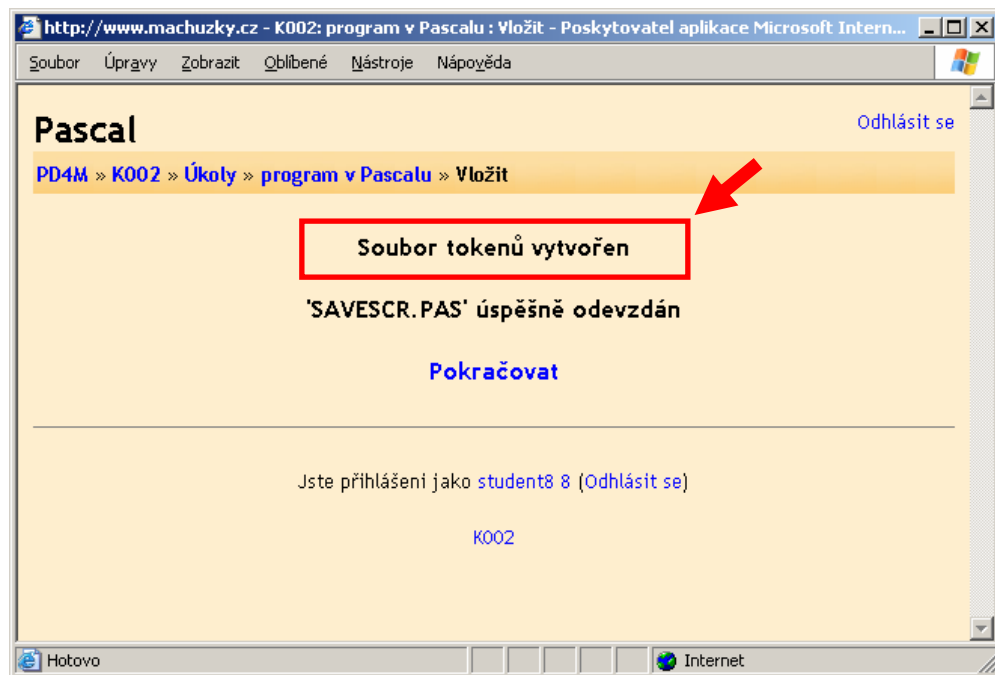
obrázek 5 - úkol v kurzu

Pokud student klepne na odkaz v osnově kurzu, vypíše se informace o úkolu (zadání) a formulář pro odevzdání.



obrázek 6 – úkol

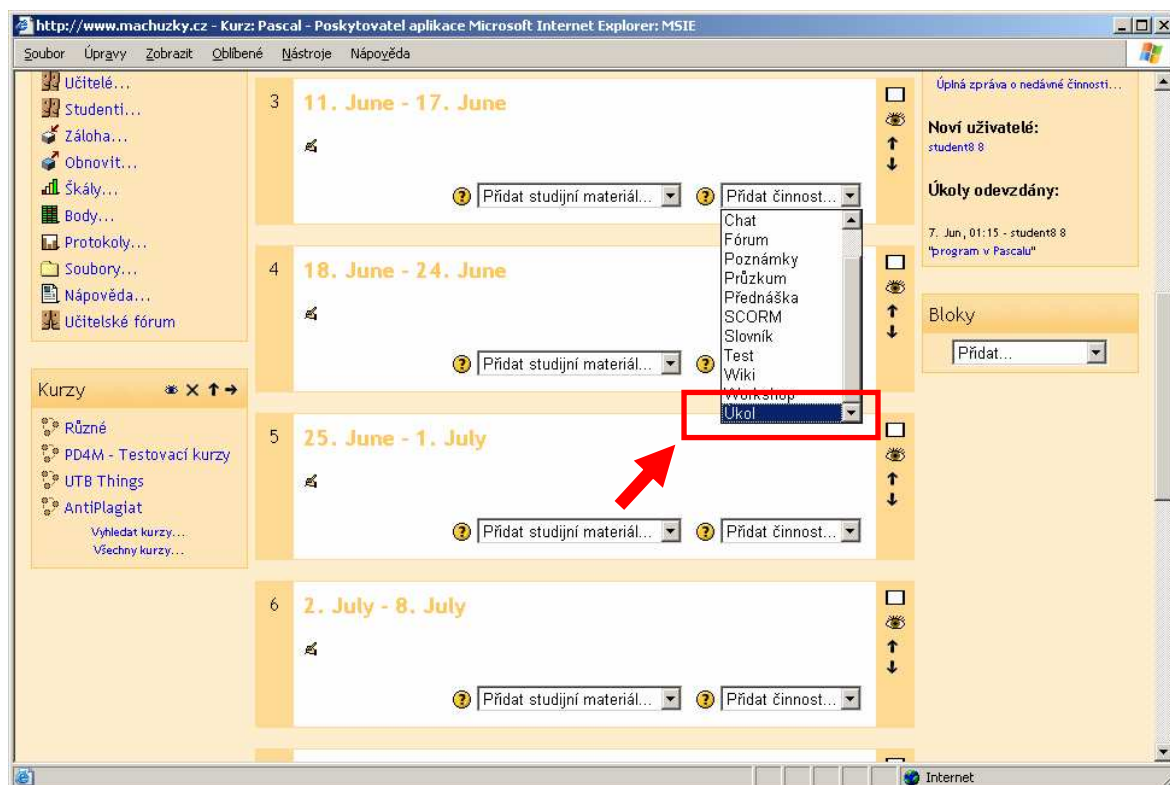
Při odevzdání úkolu studentem se po provedení uploadu ihned provede tokenizace (1.2.2), Upload odevzdávaného souboru zpracovává skript *upload.php*, po zpracování uploadu se zavolá funkce `pd4m_create_tokens` (*pd4m.php*) která vytvoří soubor tokenů pomocí tokenizeru určeného programovacím jazykem úkolu.



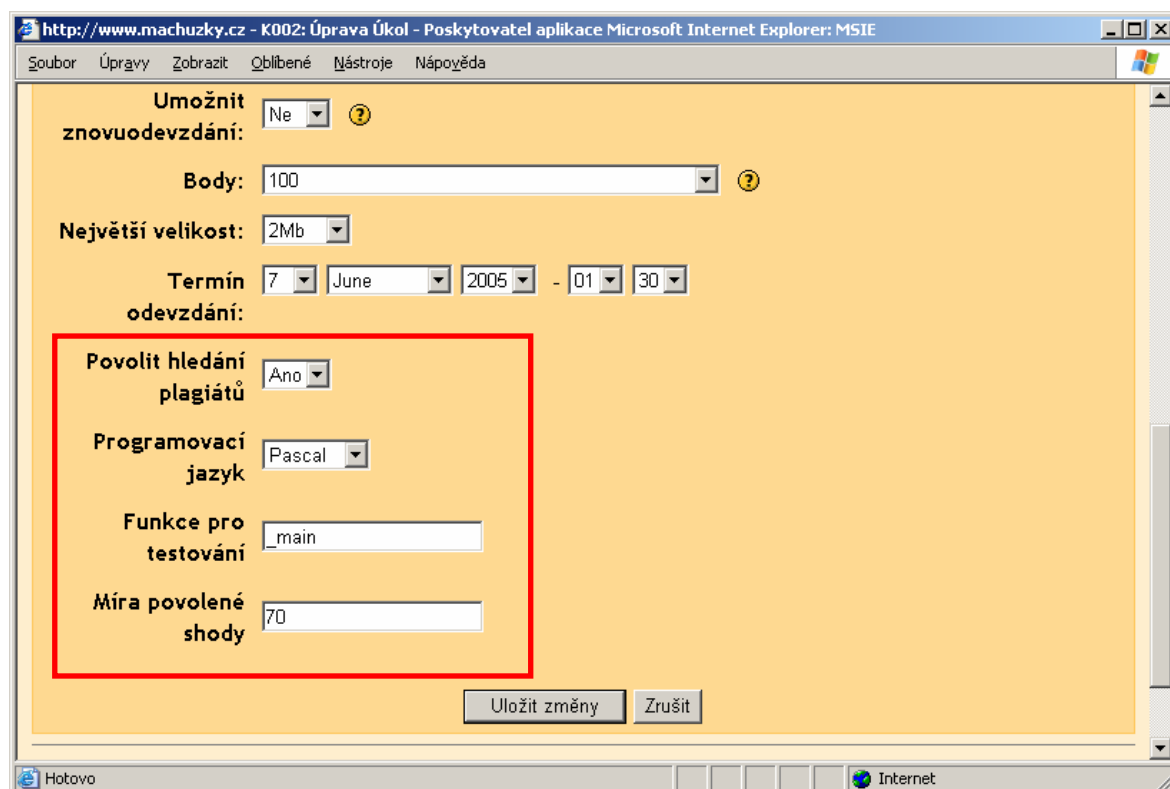
obrázek 7 – úkol odevzdán

Soubory tokenů se ukládají do adresáře určeného v souboru *pd4m.php* proměnnou \$PD4MCFG->newpath.

2.2.2 Rozhraní učitele



obrázek 8 – přidání úkolu do osnovy



obrázek 9 – nastavení úkolu

Červeně zarámovaná část formuláře je nastavení týkající se hledání plagiátů:

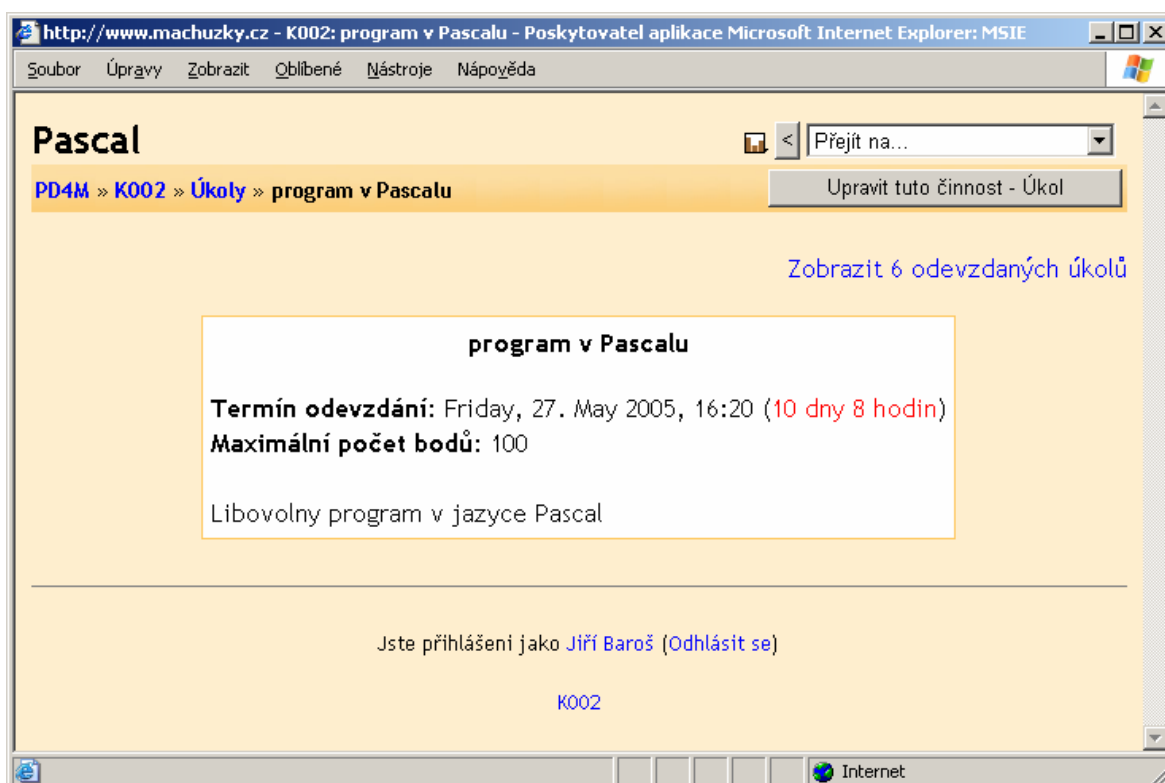
- Povolit hledání plagiátů – povolení detekčních funkcí modulu (tokenizace při uploadu, zobrazení tlačítek pro zahájení detekce).
- Programovací jazyk – pro nastavení správného tokenizeru.
- Funkce pro testování – učitel zadá kterou funkci (blok) z odevzdaného zdrojového testu chce porovnávat, úkol studenta je například jen napsat jednu funkci do učitelem připravené šablony.

tabulka 2 – Struktura tabulky `assignment`

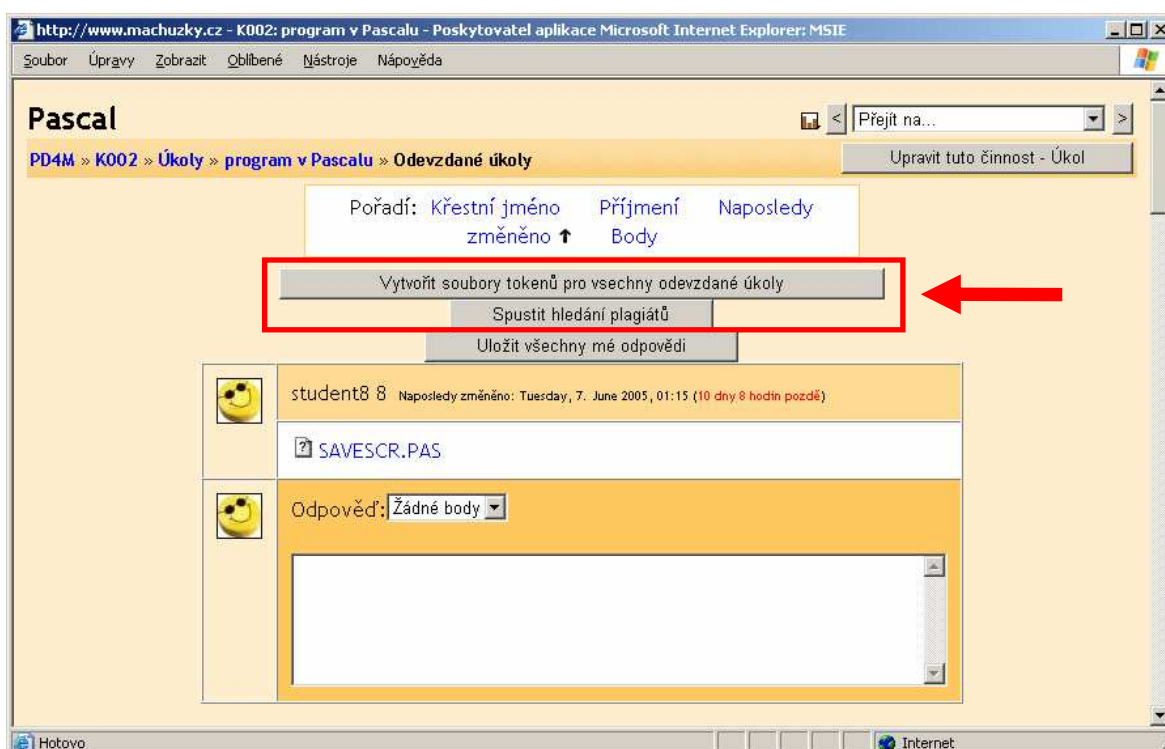
Sloupec	Typ	Vlastnosti	Nulový	Výchozí	Extra
<code>id</code>	<code>int(10)</code>	UNSIGNED	Ne		<code>auto_increment</code>
<code>course</code>	<code>int(10)</code>	UNSIGNED	Ne	0	
<code>name</code>	<code>varchar(255)</code>		Ne		
<code>description</code>	<code>text</code>		Ne		
<code>format</code>	<code>tinyint(2)</code>	UNSIGNED	Ne	0	
<code>resubmit</code>	<code>tinyint(2)</code>	UNSIGNED	Ne	0	
<code>type</code>	<code>int(10)</code>	UNSIGNED	Ne	1	
<code>maxbytes</code>	<code>int(10)</code>	UNSIGNED	Ne	100000	
<code>timedue</code>	<code>int(10)</code>	UNSIGNED	Ne	0	
<code>grade</code>	<code>int(10)</code>		Ne	0	
<code>timemodified</code>	<code>int(10)</code>	UNSIGNED	Ne	0	
<code>pd4menable</code>	<code>int(1)</code>		Ne	0	
<code>proglang</code>	<code>varchar(20)</code>		Ne	c	
<code>functiontotest</code>	<code>varchar(250)</code>		Ne	main	
<code>similaritylimit</code>	<code>int(11)</code>		Ne	50	

Informace o úkolu se v systému Moodle ukládají v databázi do tabulky `assignment`. Její strukturu popisuje tabulka 2, poslední čtyři řádky jsou informace o nastavení týkajícího se hledání plagiátů:

- `pd4menable` – povolení funkcí hledání plagiátů.
- `proglang` – programovací jazyk úkolu.
- `functiontotest` – název funkce která se bude ve zdrojovém kódu prověřovat.
- `similaritylimit` – míra povolené shody mezi dvěma odevzdanými úkoly.



obrázek 10 – úkol - rozhraní učitele

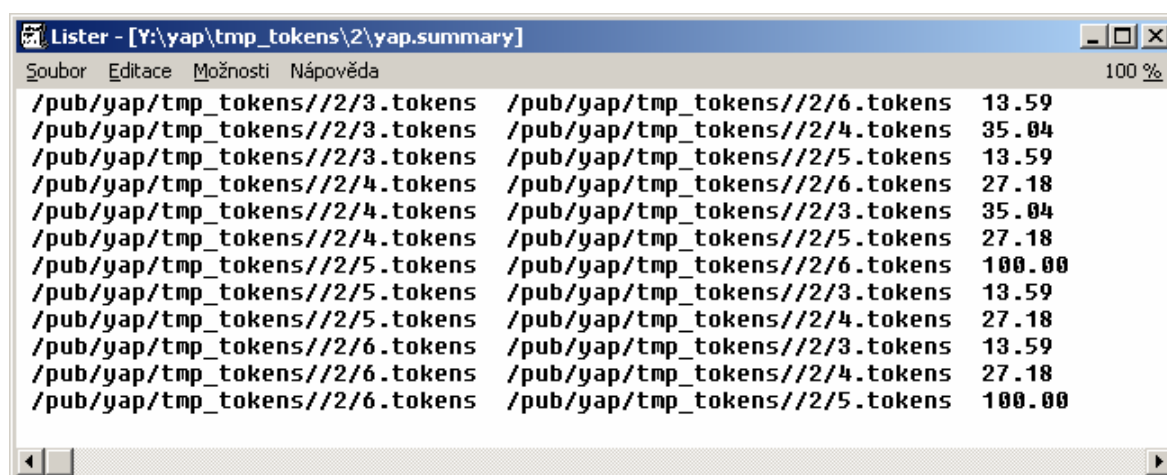


obrázek 11 – hodnocení odevzdaných úkolů

Na stránce kde učitel hodnotí odevzdané úkoly má možnost tlačítkem “Spustit hledání plagiátů” provést kontrolu všech odevzdaných souborů.

Klepnutím na toto tlačítko se spustí skript *pd4m.make.php*, který zavolá funkce definované v *pd4m.php*: `ExecYap()` a `LoadYapSummary`.

Funkce `ExecYap()` spustí porovnání souborů tokenů pro daný úkol.



/pub/yap/tmp_tokens//2/3.tokens	/pub/yap/tmp_tokens//2/6.tokens	13.59
/pub/yap/tmp_tokens//2/3.tokens	/pub/yap/tmp_tokens//2/4.tokens	35.04
/pub/yap/tmp_tokens//2/3.tokens	/pub/yap/tmp_tokens//2/5.tokens	13.59
/pub/yap/tmp_tokens//2/4.tokens	/pub/yap/tmp_tokens//2/6.tokens	27.18
/pub/yap/tmp_tokens//2/4.tokens	/pub/yap/tmp_tokens//2/3.tokens	35.04
/pub/yap/tmp_tokens//2/4.tokens	/pub/yap/tmp_tokens//2/5.tokens	27.18
/pub/yap/tmp_tokens//2/5.tokens	/pub/yap/tmp_tokens//2/6.tokens	100.00
/pub/yap/tmp_tokens//2/5.tokens	/pub/yap/tmp_tokens//2/3.tokens	13.59
/pub/yap/tmp_tokens//2/5.tokens	/pub/yap/tmp_tokens//2/4.tokens	27.18
/pub/yap/tmp_tokens//2/6.tokens	/pub/yap/tmp_tokens//2/3.tokens	13.59
/pub/yap/tmp_tokens//2/6.tokens	/pub/yap/tmp_tokens//2/4.tokens	27.18
/pub/yap/tmp_tokens//2/6.tokens	/pub/yap/tmp_tokens//2/5.tokens	100.00

obrázek 12 – výsledky hledání plagiátů

Porovnává se každý soubor s každým, první dva sloupce jsou soubory tokenů které se porovnávají a třetí sloupec je míra shody mezi soubory v procentech, výsledek se uloží do adresáře se soubory tokenů.

Funkce `LoadYapSummary()` provede načtení souboru obsahujícího výsledky porovnání (obrázek 1) pomocí funkcí PHP pro práci s řetězci. Z výsledného souboru se vyextrahují názvy jednotlivých souborů, které určují id studenta jak je veden v databázi Moodle, a míra shody mezi soubory.

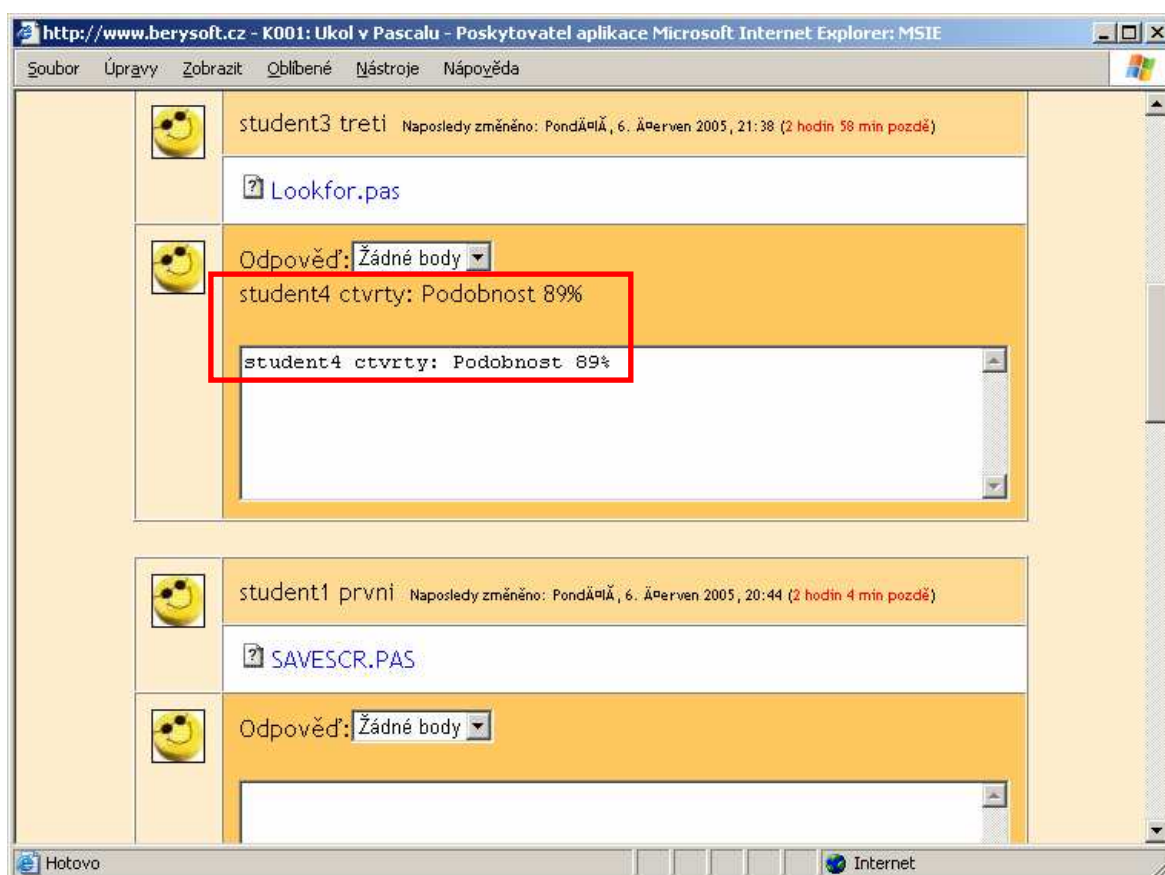
Pokud míra shody přesahuje limit který učitel zadal v nastavení úkolu, uloží se v databázi do tabulky `assignment_submissions_similarity` záznam o podobnosti dvou úkolů.

tabulka 3 - Struktura tabulky `assignment_submissions_similarity`

Sloupec	Typ	Vlastnosti	Nulový	Výchozí	Extra
<code>id</code>	<code>int(10)</code>	UNSIGNED	Ne		<code>auto_increment</code>
<code>id_pri</code>	<code>int(10)</code>		Ne	0	
<code>id_sec</code>	<code>int(10)</code>		Ne	0	
<code>similarity</code>	<code>float</code>		Ne	0	
<code>assignment</code>	<code>int(10)</code>	UNSIGNED	Ne	0	

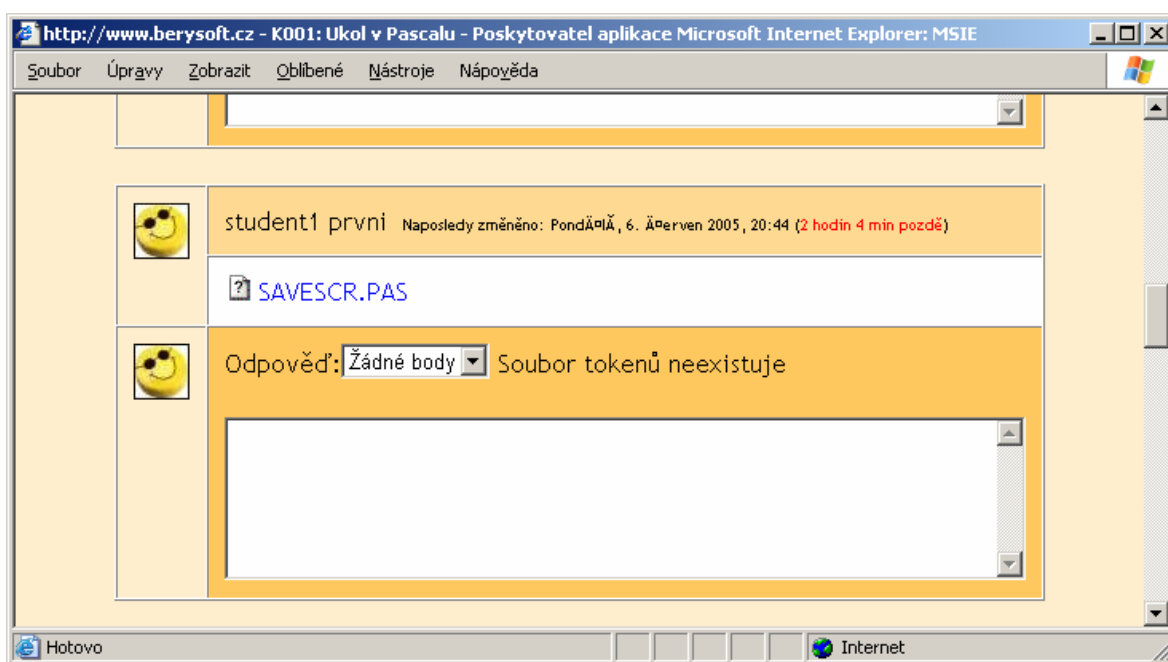
Sloupce `id_pri` a `id_sec` určují id studentů jejichž úkoly se shodují nad stanovenou míru. `similarity` je míra podobnosti v procentech a `assignment` je id úkolu.

Nyní se na stránce s hodnocením odevzdaných úkolů zobrazí u inkriminovaných podobných souborů informace o podobnosti z tabulky `assignment_submissions_similarity`. Zobrazí se jméno studenta s podobným úkolem a míra shody. Pokud je soubor podobný na více jiných souborů, zobrazí se všechny. Informace o plagiátu se zároveň doplní do textového pole kde učitel píše poznámku k odevzdanému úkolu, Tyto operace obstarávají funkce `pd4m_print_similarities` a `pd4m_get_similarities` ze souboru *pd4m.php*.



obrázek 13 – zobrazení informace o plagiátu

Pro každý odevzdaný úkol je provedena kontrola zdali existuje soubor tokenů funkcí `pd4m_check_tokens_created`. Pokud tokeny pro soubor neexistují úkol nebyl zahrnut do kontroly podobnosti a učitel je na toto upozorněn.



obrázek 14 – upozornění na neexistenci souboru tokenů

ZÁVĚR

Celý systém navržen tak aby zjednodušil a zefektivnil práci učitele při hodnocení úkolů v kurzech programování.

Jako efektivnější vhodnější pro dané použití, se jeví princip porovnání struktury zdrojového kódu. Tato metoda opravdu odhalí zdrojové texty u kterých je algoritmus stejný.

Jeden z předpokladů pro úspěšnou detekci je minimální složitost zadaného úkolu, u velmi jednoduchých úloh, kde není mnoho možností jak navrhnout algoritmus řešení může systém označit za plagiát program který plagiátem není. Detekce plagiátů vždy funguje jen s určitou konečnou úspěšností, tato úspěšnost je dobrá pokud zdrojový kód obsahuje minimálně cca. 70 – 100 elementů (tokenů).

Systém Moodle je velmi komplexní a kvalitní systém pro podporu výuky, zkušenosti s tímto systémem z výuky na našem institutu jsou velmi dobré, odevzdávání úkolů pomocí tohoto systému je jedna z hlavních výhod tohoto systému pro studenty i učitele. Student má přehled o tom co již odevzdal a jaké má hodnocení, a učitel má pohromadě všechny odevzdané úkoly. Ve srovnání s klasickou metodou posílání úkolů elektronickou poštou je toto opravdu efektivnější a spolehlivější.

Integrace automatické detekce plagiátů přímo do systému Moodle dále zefektivňuje práci učitele, který by jinak musel používat externí nástroje.

Možnosti vylepšení

Systém není ještě ve finální verzi. Důležité budou poznatky z reálného nasazení v kurzech programování.

Použitá metoda porovnání struktury zdrojového kódu je sice efektivní, ale systém by mohl obsahovat i jiné metody, aplikovat je všechny a poté váhově vyhodnocovat výsledek porovnání.

V současné verzi je nutné detekci spustit manuálně. Detekce by se například mohla spouštět automaticky po termínu odevzdání úkolu.

SEZNAM POUŽITÉ LITERATURY

Publikace:

- [1] Stubblebine, Tony *Regular Expression Pocket Reference*, O'Reilly, August 2003,
ISBN : 0-596-00415-X

Elektronické monografie:

- [2] Wise, Michael J.: *YAP: Detection of Similarities in Student Programs*,
<http://www.bio.cam.ac.uk/~mw263/ftp/doc/yap.ps>
- [3] Wise, Michael J.: *Running Karp-Rabin and Greedy String Tiling*, technical report
number 463, 1993, <http://www.it.usyd.edu.au/research/tr/tr463.pdf>
- [4] Wise, Michael J.: *String similarity via Greedy String Tiling and Running Karp-Rabin
matching*, 1993, http://www.bio.cam.ac.uk/~mw263/ftp/doc/RKR_GST.ps
- [5] Wise, Michael J.: *YAP3: Improved detection of similarities in computer program and
other texts*, <http://www.bio.cam.ac.uk/~mw263/ftp/doc/yap3.ps>
- [6] Wise, Michael J.: *Software for Detecting Suspected Plagiarism: Comparing Structure
and Attribute-Counting Systems*,
http://www.bio.cam.ac.uk/~mw263/ftp/doc/yap_vs_acm.ps
- [7] Clough, Paul: *Plagiarism in natural and programming languages: an overview of
current tools and technologies*, 2000, Department of Computer Science,
University of Sheffield,
http://www.dcs.shef.ac.uk/~cloughie/plagiarism/HTML_Version/index.html

Internetové stránky

- [8] Wise, Michael J.: *Plagiarism Detection – YAP*,
<http://www.bio.cam.ac.uk/~mw263/YAP.html>
- [9] Centre for Interactive Systems Engineering, Plagiarism Prevention and Detection,
<http://cise.lsbu.ac.uk/resources.html>
- [10] Systém Moss, <http://www.cs.berkeley.edu/~aiken/moss.html>
- [11] Moodle: <http://moodle.org>

[12] Moodle – česká lokalizace: <http://moodle.cz>

[13] CVS - <https://www.cvshome.org/>

Elektronické seriály

[14] Žáček, Kamil, *Úvod do skriptování v BASH*

<http://docs.linux.cz/programming/interpreted/bashdoc-1.4/index.html>

[15] Hubička, Jan: *Školičky, SED*, <http://docs.linux.cz/misc/skolicky/skolicka21.html>

[16] Hubička, Jan: *Školičky, AWK*, <http://docs.linux.cz/misc/skolicky/skolicka20.html>

[17] Satrapa, Pavel: *Regulární výrazy*, <http://www.root.cz/clanky/regularni-vyrazy-1/>

Elektronické dokumentace

[18] PHP Manual, <http://www.php.net/docs>

[19] MySQL Documentation, <http://dev.mysql.com/doc/>

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

ACS	Attribute Counting System – Systém pro porovnávání textu pomocí počítání atributů
Awk	Utilita pro zpracování textu, jazyk textových manipulací.
bash	Bourne Again Shell – standardní interpret příkazů pro systémy UNIX, Linux
CVS	Concurrent Versions System – systém pro správu vývoje velkých projektů.
ELF	Executable Linux File – formát binárního spustitelného souboru pro OS Linux.
EXE	Spustitelný soubor pro DOS nebo Windows.
GPL	General Public Licence – volná licence.
HTML	Hyper-Text Markup Language – jazyk pro popis dokumentu.
IMAP	Internet Message Access Protocol – protokol pro prohlížení elektronické pošty uložené na serveru.
JPEG	Joint Photographic Experts Group - formát bitmapové grafiky.
LDAP	Lightweight Directory Access Protocol – protokol pro sdílení seznamu uživatelských účtů.
MySQL	Volně šiřitelný databázový server.
NNTP	Network News Transport Protocol – protokol pro prohlížení diskusních serverů.
PHP	PHP: Hypertext Preprocessor, nebo Personal Home Page – skriptovací jazyk pro tvorbu interaktivních webových aplikací.
PNG	Portable Network Graphics – formát bitmapové grafiky.
POP3	Post Office Protocol 3 – protokol pro stahování elektronické pošty z poštovního serveru.
RKR-GST	Running Karp-Rabin and Greedy String Tiling – algoritmus pro výpočet podobnosti dvou řetězců.
sed	Utilita pro dávkové zpracování textu.

SSL	Secure Socket Layer – šifrovací metoda pro zabezpečení síťové komunikace na aplikační úrovni
TAR	Tape Archive – formát pro archivaci souborů.
TLS	Transport Layer Security – zabezpečení síťové komunikace šifrováním
WYSIWYG	What You See Is What You Get – dokument při editaci vypadá tak jak bude vypadat v prohlížeči
ZIP	Formát komprese souborů.
Zkr.	Vysvětlení.

SEZNAM OBRÁZKŮ

obrázek 1 – Úvodní stránka systému Moodle, používaného na IŘPI – www.vyuka.ft.utb.cz.....	23
obrázek 2 – systém WebCT – www.webct.com	24
obrázek 3 – Domovská stránka projektu Moodle, www.moodle.org	26
obrázek 4 – Systém Moodle – modul Fórum (www.vyuka.ft.utb.cz).....	31
obrázek 5 - úkol v kurzu	38
obrázek 6 – úkol.....	39
obrázek 7 – úkol odevzdán	40
obrázek 8 – přidání úkolu do osnovy	41
obrázek 9 – nastavení úkolu.....	41
obrázek 10 – úkol - rozhraní učitele	43
obrázek 11 – hodnocení odevzdaných úkolů	43
obrázek 12 – výsledky hledání plagiátů	44
obrázek 13 – zobrazení informace o plagiátu	46
obrázek 14 – upozornění na neexistenci souboru tokenů	47

SEZNAM TABULEK

tabulka 1 – Funkční struktura zdrojového kódu (tokenizace).....	15
tabulka 2 – Struktura tabulky assignment	42
tabulka 3 - Struktura tabulky assignment_submissions_similarity	45

SEZNAM PŘÍLOH

P I CD-ROM – Modul Assignment-PD4M, instalační balíček Moodle