**Vysoká škola polytechnická Jihlava**

Katedra technických studií   
Programovací jazyky a překladače

**Překladač z jazyka daného gramatikou do mezikódu ve formě čtveřic**semestrální práce

Autor práce: Tomáš Cink  
Jihlava 2019

Obsah

[Zadání 3](#_Toc9796813)

[1 Analýza problému 5](#_Toc9796814)

[2 Teoretický a technický návrh řešení 6](#_Toc9796815)

[2.1 Lex 6](#_Toc9796816)

[2.2 Yacc 6](#_Toc9796817)

[2.3 Makefile 9](#_Toc9796818)

[3 Testování 11](#_Toc9796819)

[4 Závěr 18](#_Toc9796820)

Seznam obrázků

[Obrázek 1 - Gramatika (zdroj: zadání semestrální práce) 3](#_Toc10049218)

[Obrázek 2 - Lexikální analýza (vlastní) 6](#_Toc10049219)

[Obrázek 3 - Ukázka ze souboru sp.y (vlastní) 7](#_Toc10049220)

[Obrázek 4 - Práce s parametrem (vlastní) 8](#_Toc10049221)

[Obrázek 5 - Výpis čtveřic s pozicemi a symboly (vlastní) 9](#_Toc10049222)

[Obrázek 6 - Makefile (vlastní) 10](#_Toc10049223)

[Obrázek 7 - Výpis nápovědy (vlastní) 11](#_Toc10049224)

[Obrázek 8 - Testovací soubor 1 (vlastní) 11](#_Toc10049225)

[Obrázek 9 - Test č.1 mezikód (vlastní) 12](#_Toc10049226)

[Obrázek 10 - Testovací soubor 2 (vlastní) 13](#_Toc10049227)

[Obrázek 11 - Test č.2 mezikód (vlastní) 13](#_Toc10049228)

[Obrázek 12 - Test 2 základní trasování (vlastní) 14](#_Toc10049229)

[Obrázek 13 - Testovací soubor 3 (vlastní) 15](#_Toc10049230)

[Obrázek 14 - Úplné trasování testu 3 (vlastní) 16](#_Toc10049231)

[Obrázek 15 - Test č. 3 mezikód (vlastní) 17](#_Toc10049232)

[Obrázek 16 - Testování možnosti -d s posláním souboru 18](#_Toc10049233)

[Obrázek 17 - Testování syntaxe pomocí zadání do terminálu 18](#_Toc10049234)

Seznam tabulek

[Tabulka 1 - Syntaxe a sémantika (zdroj: zadání semestrální práce) 3](#_Toc9796821)

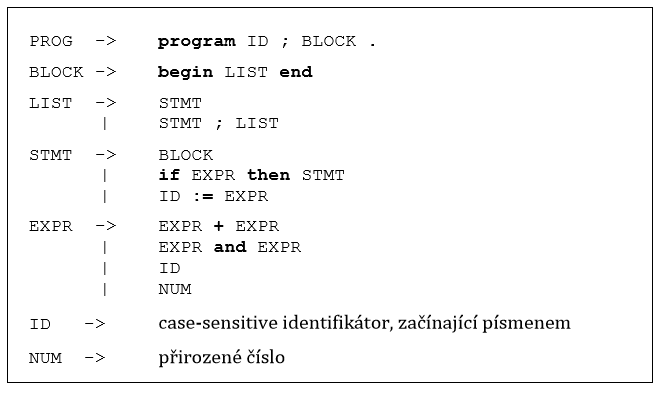
[Tabulka 2 - Parametry (zdroj: zadání semestrální práce) 4](#_Toc9796822)

Seznam použitých zkratek

YACC Yet another compiler compiler

# Zadání

Navrhněte, vytvořte, otestujte a zdokumentujte překladač programovacího jazyka, generovaného následující gramatikou:



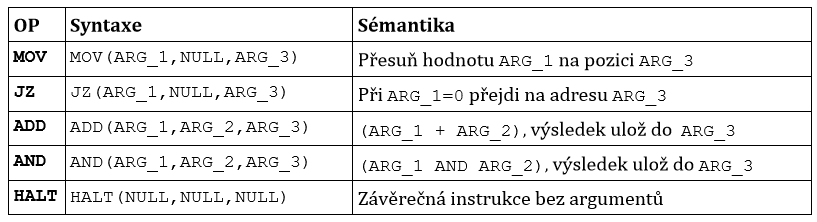
Obrázek 1 - Gramatika (zdroj: zadání semestrální práce)

a generujícího intermediální kód (mezikód) ve formě čtveřic (quadruples) s tímto obecným formátem:



a následující syntaxí a sémantikou:

Tabulka 1 - Syntaxe a sémantika (zdroj: zadání semestrální práce)



**Poznámky ke specifikaci jazyka a způsobu implementace:**

1. Všechny „neviditelné“ mezery (white spaces) jsou ignorovány
2. Jazyk povoluje samostatně stojící celořádkové komentáře, předznamenané středníkem:

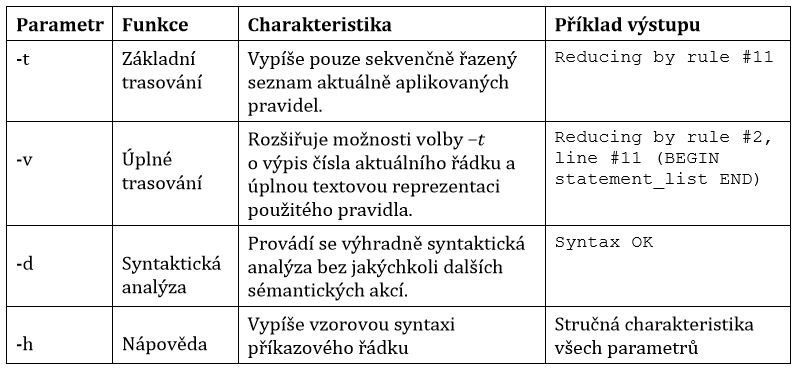
Kód 1

; Toto je příklad komentáře

Kód 2

1. Výsledný překladač bude realizován pomocí standardních generátorů (např. typu lex a yacc) ve formě dvou souborů, lexikálního a syntaktického analyzátoru. Překlad projektu zajistěte prostřednictvím utility make. Ke zdrojovým kódům dodejte nejméně 3 testovací příklady.
2. Překladač pracuje se standardním vstupem a výstupem (stdin, stdout)
3. Zajistěte, aby překladač po nalezení jakékoli syntaktické chyby nepřerušil svoji práci, ale zpracoval i zbytek vstupního kódu. Na konci vypíše, pokud možno co nevýstižnější informaci o chybách, které se v průběhu překladu vyskytly.
4. Spustitelný soubor, realizující Váš překladač, bude v závislosti na parametrech příkazového řádku pracovat v následujících režimech:

Tabulka 2 - Parametry (zdroj: zadání semestrální práce)



# Analýza problému

Hlavním krokem k analýze je rozvrhnout tokeny potřebné v gramatice. V zadání jsou zvýrazněné a jsou to slova *program, begin, end, if, then, and* a symboly *„;“* pro zaznačení komentáře, *„.“* pro označení konce programu, *„+“* ke sčítání dvou expression a přiřazení pomocí *„:=“*. Dále lze zadat libovolnou proměnnou značenou v *ID* anebo numerické hodnoty značené *NUM*. Tyto tokeny jsou vytvořeny v lexikální analýze.

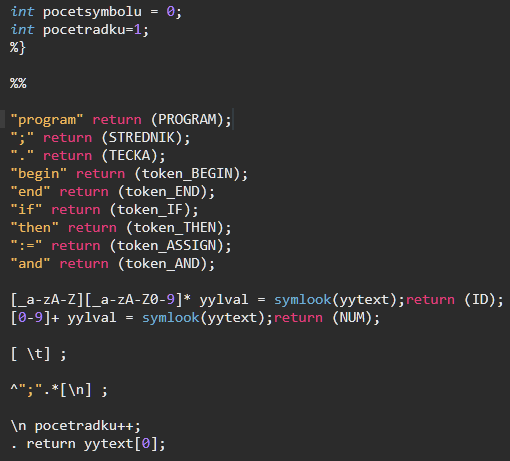
Pro syntaktickou analýzu lze využít zadanou gramatiku, kde jsou zvolené neterminály *PROG*, *BLOCK*, *LIST*, *STMT*, *EXPR,* ve kterých se dále využívá terminálů *ID* a *NUM* a též tokenů z lexikální analýzy.

# Teoretický a technický návrh řešení

V této části budou popsány definice z lexikální a syntaktické analýzy

## Lex

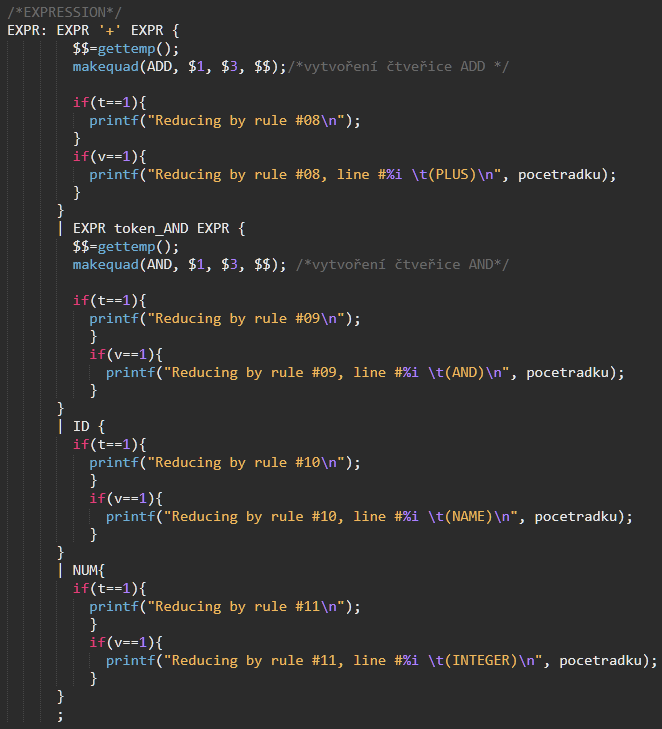
Tato část má obsažené deklarace knihoven pro přístup ke vstupním výstupním operacím a pomocným deklaracím ze souboru *sp.h*. Také jsou zde definovány deklarace pro tokeny, proměnné, číselné hodnoty, ignorování mezer a ošetření komentářů.



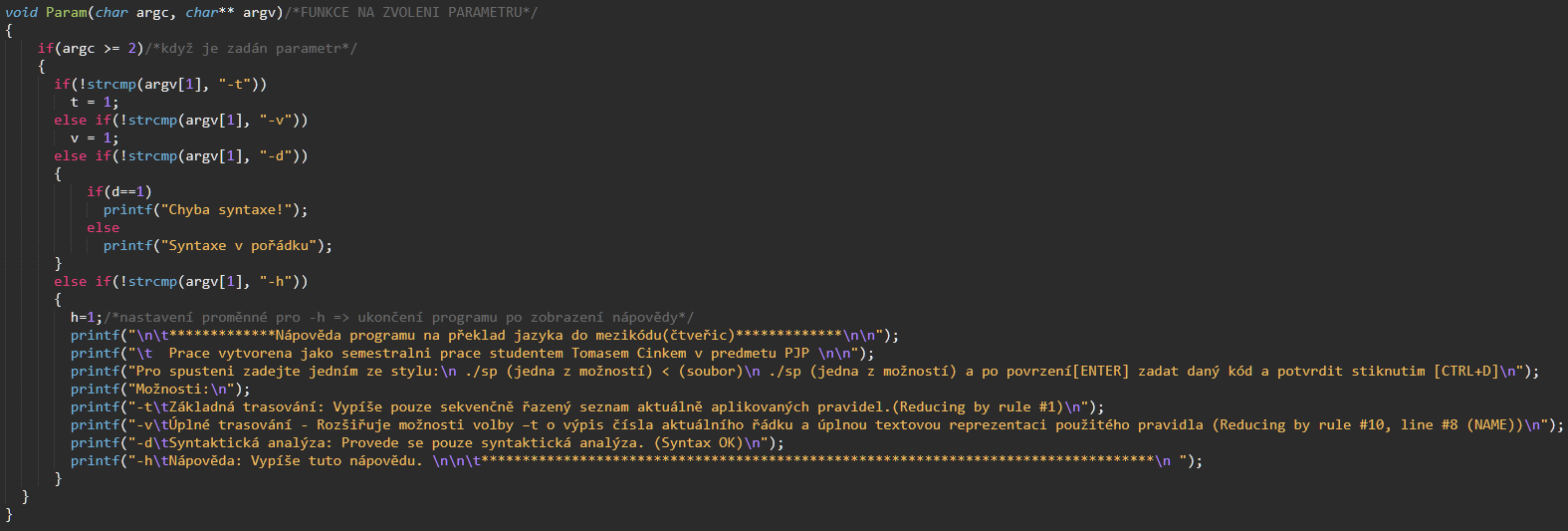
Obrázek 2 - Lexikální analýza (vlastní)

## Yacc

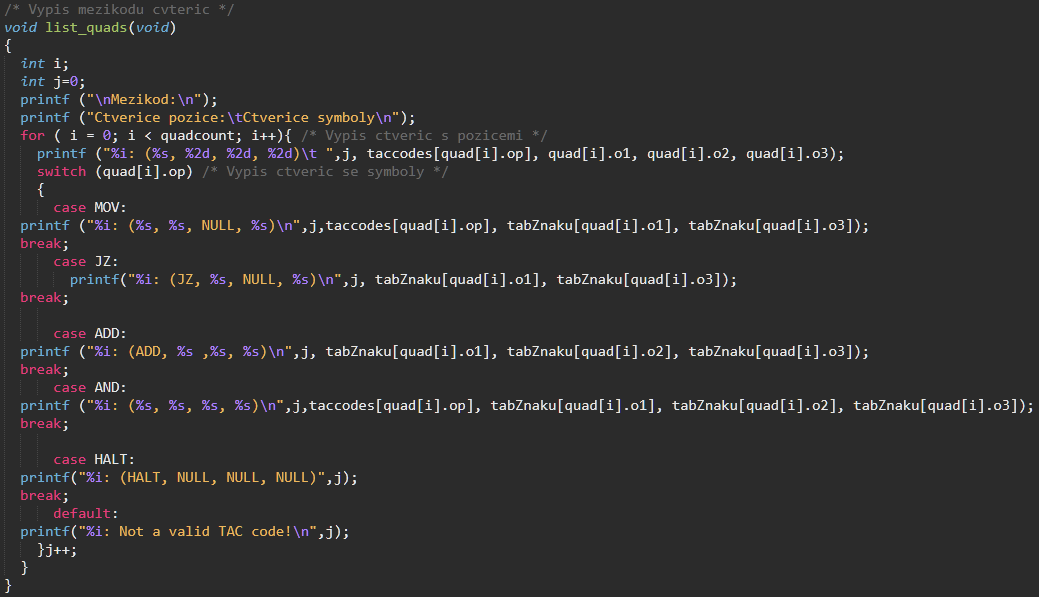
V tomto souboru jsou definovány potřebné knihovny, pomocné proměnné, gramatiku definovanou v zadání, zachycení chyb, ošetření parametrů pro vstup a výpis mezikódu a tabulky symbolů.



Obrázek 3 - Ukázka ze souboru sp.y (vlastní)



Obrázek 4 - Práce s parametrem (vlastní)



Obrázek 5 - Výpis čtveřic s pozicemi a symboly (vlastní)

## Makefile

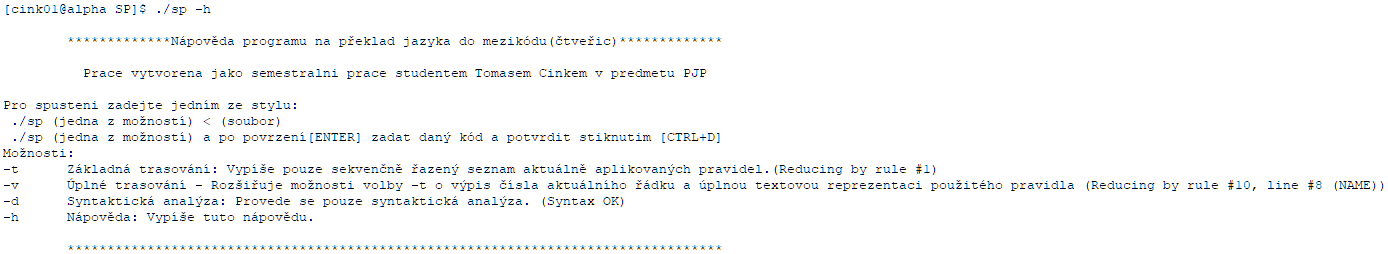
Pro zjednodušení kompilace spustitelného souboru slouží soubor *Makefile*, který provede všechny potřebné příkazy za nás. Dále obsahuje přidané příkazy jako *make help*. Který zobrazí nápovědu ke spuštění programu.



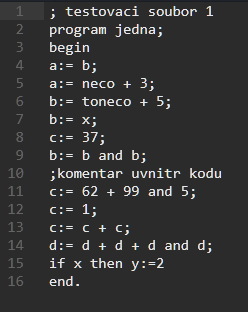
Obrázek 6 - Makefile (vlastní)

# Testování

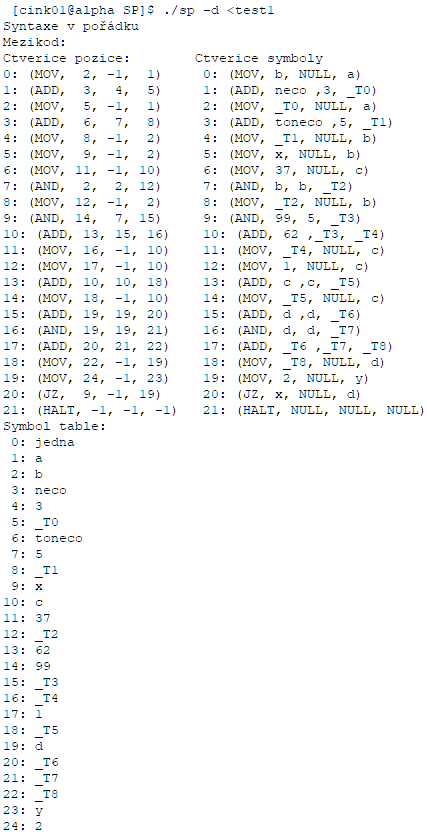
Pro testování jsou vytvořeny tři testovací soubory, které budou využiti k testování funkčnosti a ukázce práce s parametry.



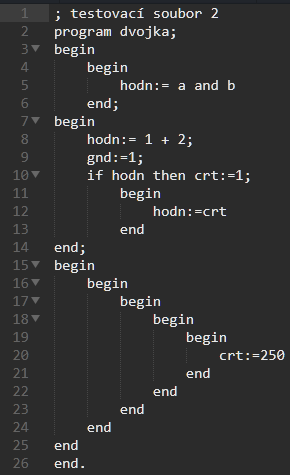
Obrázek 7 - Výpis nápovědy (vlastní)



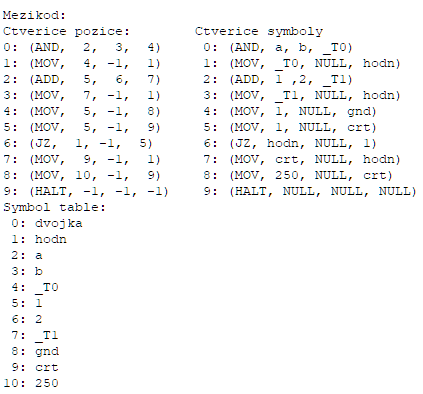
Obrázek 8 - Testovací soubor 1 (vlastní)



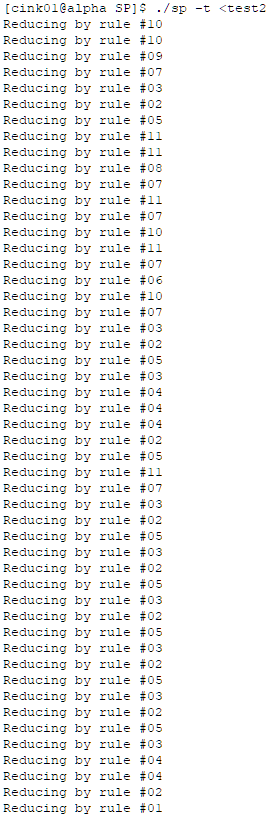
Obrázek 9 - Test č.1 mezikód (vlastní)



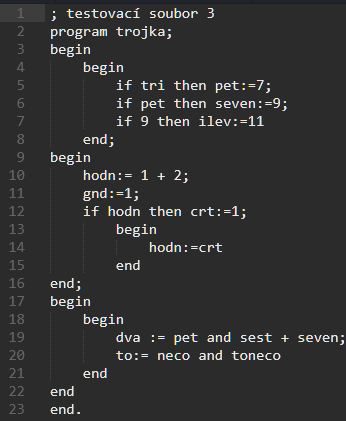
Obrázek 10 - Testovací soubor 2 (vlastní)



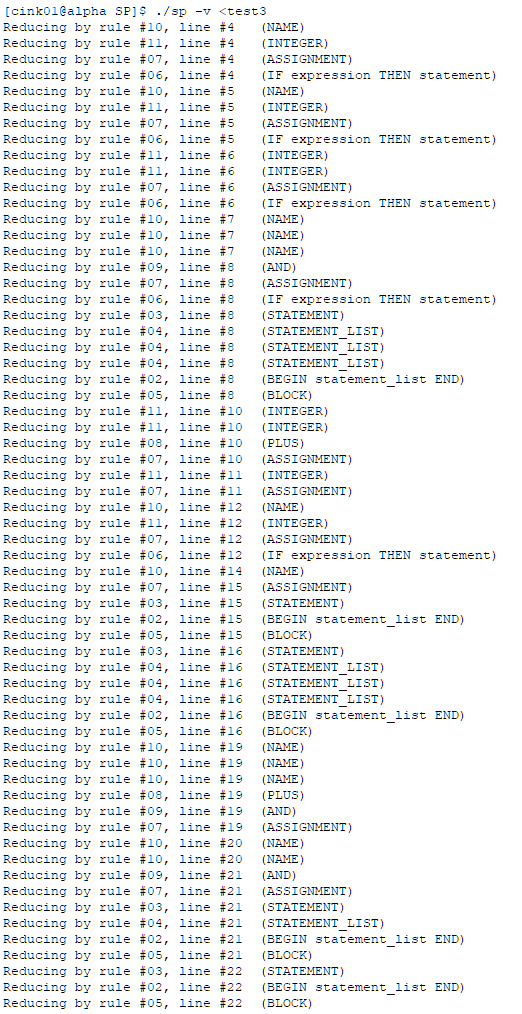
Obrázek 11 - Test č.2 mezikód (vlastní)



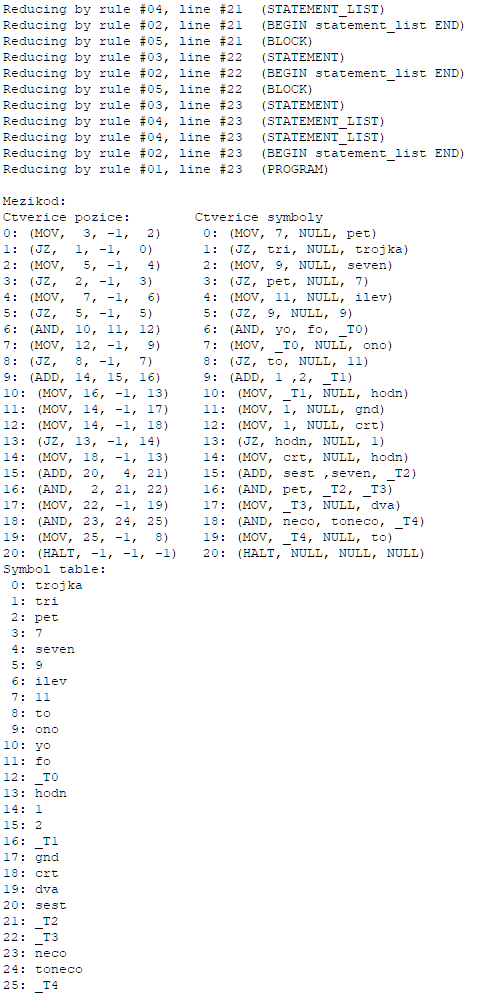
Obrázek 12 - Test 2 základní trasování (vlastní)



Obrázek 13 - Testovací soubor 3 (vlastní)



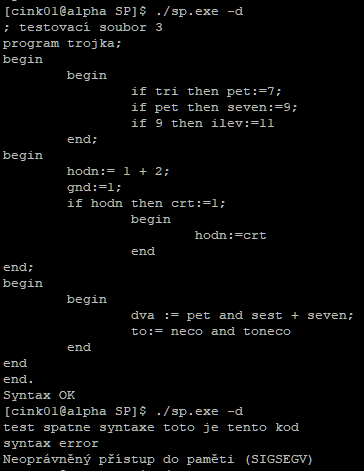
Obrázek 14 - Úplné trasování testu 3 (vlastní)



Obrázek 15 - Test č. 3 mezikód (vlastní)



Obrázek - Testování možnosti -d s posláním souboru



Obrázek - Testování syntaxe pomocí zadání do terminálu

# Závěr

Díky poslednímu cvičení bylo jednodušší vytvořit tuto práci, jelikož se jednalo o zpracovávání mezikódu. Pomocí *Makefile* je vygenerován soubor sp.exe, kterému lze poslat soubor nastavit parametr nebo vložit kód po potvrzení ručně. Na výstupu se objeví dva druhy výpisu čtveřic v prvním se zobrazují pozice v tabulce symbolů a v druhé samotné symboly, aby byla vidět i tabulka symbolů na porovnání, je zobrazena pod čtveřicemi. Podle vzorů v zadání tento úkol není stoprocentně stejný. *TAC* kód *JZ* se zobrazuje na jiném řádku a také ve vzoru vypadá, že vypisuje pozici a místo symbol na posledním místě čtveřice, což je ojedinělý výskyt tohoto prohození. Z tohoto důvodu jsou vytvářeny tyto dvojité výpisy s pozicemi a symboly. Při vytvoření se objeví několik warning upozornění, která nemají na výsledný překlad a generaci vliv.