Dokumentace projektu z předmětů IFJ a IAL

Implementace překladače imperativního jazyka IFJ19

Tým 057, varianta I

11. 12. 2019

Martin Koči (xkocim05)
Magdaléna Ondrušková (xondru14)
Michal Koval (xkoval17)
Zuzana Hradilova (xhradi16)

Obsah

1 Úvod

Cílem projektu bylo vytvořit překladač ze zdrojového jazyka IFJ19, založeného na programovacím jazyce Python, do jazyka IFJcode19. Program byl implementován v jazyce C.

2 Implementace

Projekt je rozdělen do několika částí, které byli implemetovány samostatně.

2.1 Lexikální analýza

3 Práce v týmu

Na projektu jsme pracovali jsme ve čtyřčlenném týmu. Po zveřejnění zadaní jsme se sešli na 1. týmové schůzce, ujasnili si, co všechno bude potřeba udělat a rozdělili si úkoly.
Na většině částech projektu jsme pracovali samostatně nebo ve dvojicích, ale často jsme se potkávali a společně konzultovali řešení. Komunikace probíhala také přes Discord a Messenger.
Zdrojové soubory jsme sdíleli pomocí GitHubu, což nám umožnilo jejich rychlé sdílení a složení výsledného projektu.

Martin Koči	Vedoucí týmu, syntaktická analýza, sémantická analýza
Magdaléna Ondrušková	Syntaktická analýza, zpracování výrazů, testování
Zuzana Hradilová	Tabulka symbolu, testování, dokumentace
Michal Koval	Lexikální analýza, generování kódu

Tabulka 1: Rozdělení práce v týmu

4 Diagram konečného automatu pro lexikální analýzu

5 LL Tabulka

	eof	eol	def	id	(	:	indent	=	return	pass	while	if	else	,	)	none	float	string	int	dedent	expr
<prog>	1		1	1						1	1	1									1
<st-list>	3		2	2						2	2	2									2
<stat>			4	5						7	8	9									6
<params>				11											10						
<nested-st-list>				32						32	32	32									32
<assign>	eof	eol	def	id	(	:	indent	=	return	pass	while	if	else	,	)	none	float	string	int	dedent	expr
<next-param>	eof	eol	def	id	(	:	indent	=	return	pass	while	if	else	,	)	none	float	string	int	dedent	expr
<arg-params>	eof	eol	def	id	(	:	indent	=	return	pass	while	if	else	,	)	none	float	string	int	dedent	expr
<value>	eof	eol	def	id	(	:	indent	=	return	pass	while	if	else	,	)	none	float	string	int	dedent	expr
<arg-next-params>	eof	eol	def	id	(	:	indent	=	return	pass	while	if	else	,	)	none	float	string	int	dedent	expr
<nested-stat>	eof	eol	def	id	(	:	indent	=	return	pass	while	if	else	,	)	none	float	string	int	dedent	expr
<func-nested-st-list>	eof	eol	def	id	(	:	indent	=	return	pass	while	if	else	,	)	none	float	string	int	dedent	expr
<func-nested-stat>	eof	eol	def	id	(	:	indent	=	return	pass	while	if	else	,	)	none	float	string	int	dedent	expr
<def-id>	eof	eol	def	id	(	:	indent	=	return	pass	while	if	else	,	)	none	float	string	int	dedent	expr
<after-id>	eof	eol	def	id	(	:	indent	=	return	pass	while	if	else	,	)	none	float	string	int	dedent	expr
<next-func-nested-st-list>	eof	eol	def	id	(	:	indent	=	return	pass	while	if	else	,	)	none	float	string	int	dedent	expr
<nex-nested-st-list>	eof	eol	def	id	(	:	indent	=	return	pass	while	if	else	,	)	none	float	string	int	dedent	expr
<after-return>	eof	eol	def	id	(	:	indent	=	return	pass	while	if	else	,	)	none	float	string	int	dedent	expr
<eof-or-eol>	eof	eol	def	id	(	:	indent	=	return	pass	while	if	else	,	)	none	float	string	int	dedent	expr

Tabulka 2: LL tabulka