Università di Cagliari

CdL in Informatica

Architettura degli Elaboratori

Realizzazione di un programma su Mic1 per la conversione di una espressione dal formato infisso al formato postfisso (RPN)

Relazione

Progetto Are – A.A. 2015 / 2016 – Antonello Meloni – matr. 42062

Descrizione del Progetto

Scopo del presente progetto è la realizzazione di un programma in linguaggio IJVM per il MIC1 che, mediante l'utilizzo di un semplice algoritmo, realizzi la conversione di una espressione dal formato infisso a quello postfisso.

Poichè non vi erano vincoli sull'algoritmo da utilizzare si è preferito utilizzare quello presentato a lezione e descritto nelle slide.

Tale algoritmo presenta il vantaggio di non richiedere risorse difficili da gestire su Mic1: infatti ha richiesto esclusivamente l'utilizzo dello stack, di alcune costanti e di qualche variabile di appoggio.

Sono state definite le seguenti costanti:

Nome	Valore Esadecimale	Carattere ASCII corrispondente	Utilizzo
ZERO	0x30	"0"	Verifica cifre numeriche
NOVE	0x39	"9"	
UGUALE	0x3d	"="	Inizio e fine espressione
PAP	0x28	"("	Individuazione operatori
PCH	0x29	")"	
PER	0x2a	"*"	
PIU	0x2b	"+"	
MENO	0x2d	"-"	
DIV	0x2f	"/"	
SP	0x20	" "	Inserimento spazi nell'espressione postfissa

Le variabili utilizzate sono:

Nome	Contenuto	Utilizzo
ult	0 ~ 9, + - * / =	Contiene l'ultimo carattere letto dalla console di input e viene utilizzato durante i confronti
sp	0, -1	Verifica della necessità di inserire o meno uno spazio nell'espressione in uscita.

Funzionamento del programma

Il programma è strutturato in due cicli, uno che viene ripetuto per ogni carattere presente in input e l'altro quando viene trovato il carattere di fine espressione.

Il primo analizza il carattere in ingresso, identifica se si tratta di una cifra numerica o di un operatore e salta alla parte di programma necessaria a seconda dei casi.

Il secondo vuota lo stack, una volta che è finito l'input.

Non sono stati utilizzati metodi e questo rende il listato del programma un pochino "spaghetti code", ma sono stati inseriti numerosi commenti allo scopo di rendere meno oscuro il significato di ogni salto.

L'algoritmo utilizzato è un riadattamento dell'idea di Dijkstra e opera lasciando passare direttamente sull'out le cifre numeriche e utilizzando la seguente tabella decisionale nel caso in cui ve un operatore.

		Ultimo carattere analizzato (variabile ult)						
		"="	+	-	*	/	(	)
Ultimo carattere sullo stack	"="	4	1	1	1	1	1	5
	+	2	2	2	1	1	1	2
	-	2	2	2	1	1	1	2
	*	2	2	2	2	2	1	2
	/	2	2	2	2	2	1	2
	(	5	1	1	1	1	1	3

I numeri in tabella rappresentano i seguenti comportamenti:

1. Il carattere viene posto sullo stack;
2. Il carattere in cima allo stack viene scritto sull'output e si cicla per ripetere il controllo;
3. Il carattere viene eliminato insieme a quello presente in cima allo stack;
4. Fine programma: l'output corrisponde alla versione scritta in RPN dell'espressione in input;
5. Condizione di errore nella sintassi dell'espressione.

L'input avviene attraverso la console di input.

Possono essere immessi numeri anche a più cifre (0 - 9), gli operatori matematici per le quattro operazioni e le parentesi tonde. L'immissione di altri caratteri non causa inconvenienti in quanto gli stessi vengono semplicemente ignorati.

Una limitazione nello scrivere l'espressione esiste e causa output errati: non è possibile effettuare correzioni sul testo dell'espressione mentre la si digita da tastiera in quanto, benchè visualizzato correttamente, il testo passato al programma contiene anche i caratteri eliminati ed una gestione dei caratteri immessi da tastiera per l'editing del testo esulerebbe dagli scopi di questo progetto.

Ulteriore limitazione è l'immissione obbligatoria a fine espressione del segno "=", senza la quale il programma prosegue in loop infinito.

Lo stesso carattere viene inserito automaticamente sullo stack all'inizio del programma per assicurare che lo stesso termini correttamente una volta elaborato tutto l'input.

L'input può essere fornito prima di avviare il programma o durante l'esecuzione dello stesso. In questo secondo caso è possibile vedere la costruzione dell'output un carattere alla volta, fino all'inserimento del carattere "=", che termina il programma.

Di seguito si riporta una schermata del programma.