Programmazione e Algoritmi 1

A.A. 2022/23 — Soluzioni prova del 26/6/2023

prof. Gianluca Amato

Per gli esercizi 1, 3, 4 e 6 vedere i file Java allegati.

Esercizio 2

Il programma consiste di un ciclo di $n-1$ passi, dove n è la lunghezza del parametro a. Tutte le operazioni eseguite dentro il ciclo (e anche fuori dal ciclo) sono operazioni elementari di accesso ad un elemento di un array, confronto tra due interi e simili, che richiedono tempo costante. La complessità computazionale del metodo è dunque $O(n)$. Notera che, poiché il numero di iterazioni è sempre $n-1$, indipendentemente dal contenuto dell'array a, non esiste differenza tra caso ottimo e caso pessimo.

Esercizio 5

Il problema è che il risultato dovrebbe essere -200, ma -200 non è rappresentabile in una variabile di tipo `byte`, il cui intervallo di valori ammissibili va da -128 a 127. Si verifica quindi un *overflow* e il risultato sarà sicuramente errato.

Per capire esattamente cosa succede, eseguiamo manualmente il programma, ma tenendo conto della rappresentazione binaria. Determiniamo la rappresentazione di -100 nel tipo di dato `byte`. Prima di tutto convertiamo 100 in binario. Usando il metodo delle divisioni successive, otteniamo $100_{10} = 1100100_2$. Poiché -100 è negativo, per ottenere la sua rappresentazione dobbiamo:

- estendere il numero binario che abbiamo ottenuto fino ad occupare il nuneri di bit del typo `byte` (ovvero 8 bit);
- calcolare il complemento a 2.

L'estensione di 1100100_2 su 8 bit è 01100100_2. Il suo complemento a 1 si ottiene rimpiazzando gli zeri in uno e viceversa, ovvero 10011011_2. Il complemento a 2 si ottiene sommando 1 al complemento a 1, ottenendo così 10011100.

Dunque, dopo la prima istruzione, la variabile `b` contiene la sequenza di bit 10011100. Questa viene sommata con se stessa, ottenendo 10011000 che però ha 9 bit. Il bit più a sinistra viene rimosso, visto che il tipo `byte` ha spazio solo per 8 bit, ottenendo 00111000. Che numero rappresenza questa sequenza di bit ? Poiché il bit più a sinistra è 0, allora si tratta di un numero positivo. Basta convertitolio in decimale con il solito sistema. Notiamo che sono ad 1 i bit in posizione 3, 4, e 5, quindi $0011100_2 = 2^3 + 2^4 + 2^5 = 8 + 16 + 32 = 56$, che è il risultato mostrato dal programma.

Esercizio 7

riga programma						valore variabili	note
3						args={}	main({})
4						args={} s= “abc”	
5						args={} s= “abc” t=?	
	17					x=“abc”	metodo(“abc”)
	18					x=“abc” res=“”	
	19					x=“abc” res=“” i=0	
	20					x=“abc” res=“” i=0	
		9				c=“a” n=0	repeat(“a”, 0)
		10				c=“a” n=0 res=“”	
		11				c=“a” n=0 res=“” i=0	
		14				c=“a” n=0 res=“”	return “”
	20					x=“abc” res=“” i=0	
	19					x=“abc” res=“” i=1	
	20					x=“abc” res=“” i=1	
		9				c=“b” n=1	repeat(“b”, 1)
		10				c=“b” n=1 res=“”	
		11				c=“b” n=1 res=“” i=0	
		12				c=“b” n=1 res=“b” i=0	
		11				c=“b” n=1 res=“b” i=1	
		14				c=“b” n=1 res=“b”	return “b”
	20					x=“abc” res=“b” i=1	
	19					x=“abc” res=“b” i=2	
	20					x=“abc” res=“b” i=2	
		9				c=“c” n=2	repeat(“c”, 2)
		10				c=“c” n=2 res=“”	
		11				c=“c” n=2 res=“” i=0	
		12				c=“c” n=2 res=“c” i=0	
		11				c=“c” n=2 res=“c” i=1	
		12				c=“c” n=2 res=“cc” i=2	
		11				c=“c” n=2 res=“cc” i=2	
		14				c=“c” n=2 res=“cc”	return “cc”
	20					x=“abc” res=“bcc” i=2	
	19					x=“abc” res=“bcc” i=3	
	21					x=“abc” res=“bcc”	return “bcc”
6						args={} s=“abc” t=“bcc”	print “bcc”