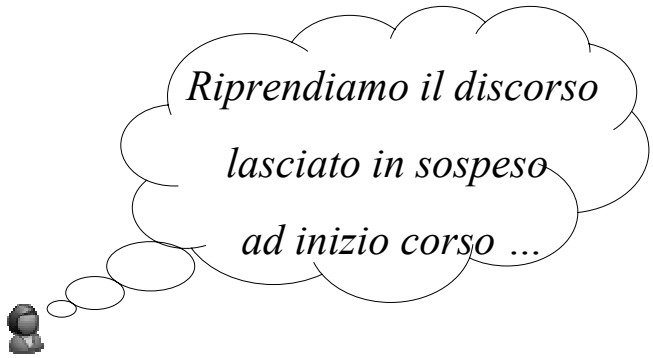
Architettura dell'elaboratore



Prodotto tra due numeri

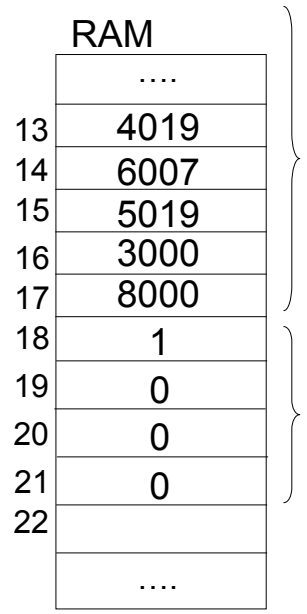
A * B = B + …+ B + B ….

RAM

0	2000
1	7015
2	4020
3	2000
4	7015
5	4021
6	4019
7	5020
8	1018
9	4020
10	7015
11	5021

12	19	
13	4019	
14	6007	
15	5019	
16	3000	
17	8000	
18	1	costante per fare il decremeto
19	0	risultato
20	0	A
21	0	B
22		
	….	

Istruzioni o dati?

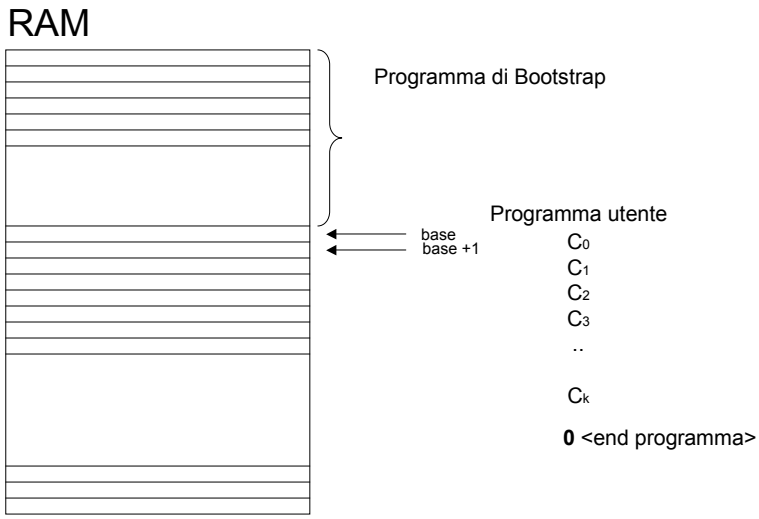


È compito del programmatore garantire che **non** si interpretino come istruzioni le celle che contengono dei dati e viceversa

Programmi automodificanti

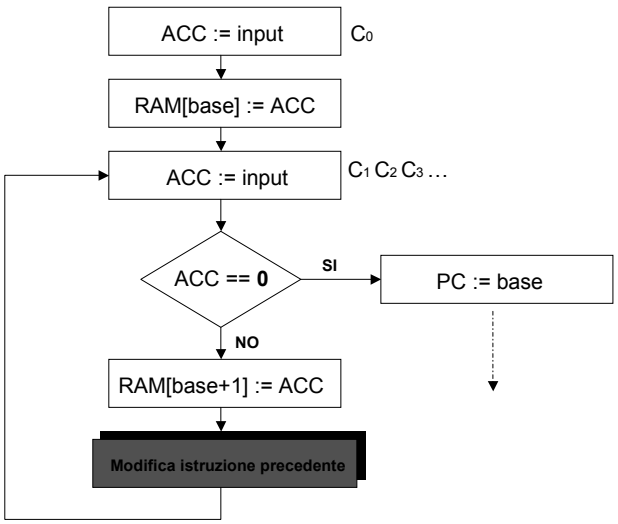
- Sono programmi che prima inseriscono un dato in una cella e poi lo interpretano come istruzione
- Sono difficili e oggi non si usano più

Il bootstrap

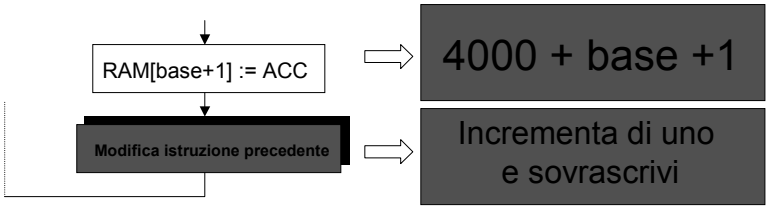


Facciamo l'ipotesi che il programma utente abbia **almeno una istruzione**

Il bootstrap: algoritmo



Il bootstrap: algoritmo



Nel seguito facciamo l'ipotesi che **base sia 20**

Il bootstrap

0	2000
1	4020
2	2000
3	7020
4	4021
5	5004
6	9
7	4004
8	6002
9	1
	….

ACC:= input
RAM[20] := ACC
ACC := input
if (ACC==0) then PC:=20
RAM[21] := ACC
ACC := RAM[4]
ACC := ACC + RAM[9]
RAM[4] := ACC
PC := 2

Attenzione! l'istruzione RAM[4] viene modificata in 4022

Il bootstrap

0	2000
1	4020
2	2000
3	7020
4	4022
5	5004
6	9
7	4004
8	6002
9	1
	….

Ogni volta che si legge una nuovo input l'istruzione RAM[4] viene **incrementata**.
Così facendo,gli input C_0, C_1, … C_k,vengono memorizzati in celle successive, a partire dall'indirizzo base (= 20)

Il bootstrap

	….
20	C_0
21	C_1
22	C_2
23	C_3
24	C_4
25	
26	….
27	
28	C_k
29	
	….

Dopo l'esecuzione di 7020 si passa ad eseguire l'istruzione contenuta in RAM[20], cioè la prima istruzione del programma utente appena caricato in RAM

Il bootstrap

0	2000
1	4020
2	2000
3	7020
4	40xx
5	5004
6	9
7	4004
8	6002
9	1
	….

Problema: quando termina il caricamento del programma utente il codide del bootstrap è **diverso** da quello originale

Possiamo “resettare” l’istruzione in RAM[4] riportandola al valore iniziale **4021**

Il bootstrap

0	2000
1	4020
2	2000
3	7011
4	40xx
5	5004
6	9
7	4004
8	6002
9	1
10	4021
11	5010
12	4004
13	6020

if (ACC==0) then PC:=11

ACC := RAM[10]

RAM[4] := ACC

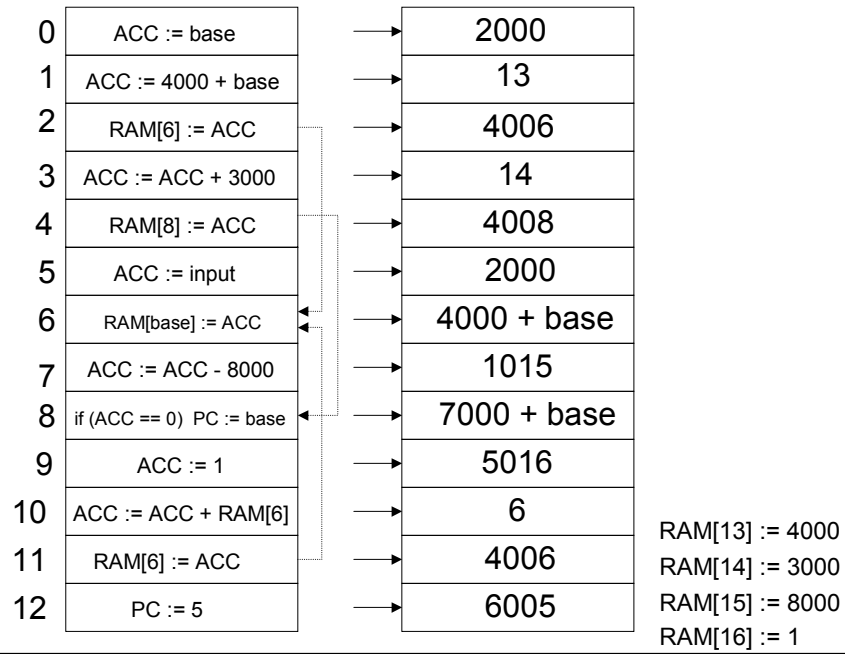
PC := 20

Il bootstrap (2)

Modifichiamo il programma di bootstrap considerando variabile anche il valore di **base**

Inoltre, il programma utente termina quando C_k**=8000**

Il bootstrap (2)



Estensione procedurale

Si definiscono dei sottoprogrammi che vengono richiamati da altri programmi per portare a termine parte del compito

Estensione procedurale

RAM[981] = 7995	if (ACC==0) PC:=995
RAM[982] = 4998	RAM[998]:=ACC
RAM[983] = 5999	ACC:=RAM[999]
RAM[984] = 7995	if (ACC==0) PC:=995
RAM[985] = 4997	RAM[997]:=ACC
RAM[986] = 5998	ACC:=RAM[998]
RAM[987] = 1996	ACC:=ACC-RAM[996]
RAM[988] = 7994	if (ACC==0) PC:=994
RAM[989] = 4998	RAM[998]:= ACC
RAM[990] = 5999	ACC:=RAM[999]
RAM[991] = 997	ACC:=ACC+RAM[997]
RAM[992] = 4997	RAM[997] := ACC
RAM[993] = 6986	PC:=986
RAM[994] = 5997	ACC:=RAM[997]
RAM[995] = 6000+i	PC:= i
RAM[996] = 1	costante 1
RAM[997] = 0	valore irrilevante
RAM[998] = 0	valore irrilevante
RAM[999] = 0	valore primo operando

Prodotto tra RAM[999] e il contenuto del registro ACC, risultato in ACC

Estensione procedurale

RAM[0] = 2000	ACC:= input
RAM[1] = 4999	RAM[999] := ACC
RAM[2] = 2000	ACC := input
RAM[3] = 6981	PC:= 981
RAM[4] = 3000	output := ACC
RAM[5] = 8000	end

Si dovrà avere RAM[995] = 6004

Le macchine convenzionali moderne …