

 POLITECNICO DI MILANO

Dipartimento di
Elettronica e Informazione

Quinta esercitazione

Riccardo Cattaneo



POLITECNICO
DI MILANO

11/11/2013



Quinta esercitazione: agenda

- (20') Es1: somma virgola fissa e mobile
- (20') Selezione esercizi da prova di esame 1
- (20') Selezione esercizi da prova di esame 2
- (10') Pausa
- (60') Es2: Spotted:InfoB

Es1: Somma in virgola mobile

$$22,375 - 32,5$$

22	0	↑	0,375	0	
11	1		0,750	1	
5	1		0,500	1	
2	0		0	END	↓
1	1				
0	END				

32	0	↑	0,5	1	↓
16	0		0	END	
8	0				
4	0				
2	0				
1	1				
0	END				

$$10110,011$$

← NORMALIZZO

$$+ 1,0110011 \times 2^4$$

$$S = 0$$

$$e = 4 + 127 = 131$$

$$m = 0110011$$

PER VIA DEL 2^a ↓

$$1,0110011 \times 2^4$$

↓ PASSIAMO A STESSO EXP (QUELLO DEL MAGGIORE)

$$0,10110011 \times 2^5$$

$$100000,1$$

← NORMALIZZO

$$- 1,000001 \times 2^5$$

$$S = 1$$

$$e = 132$$

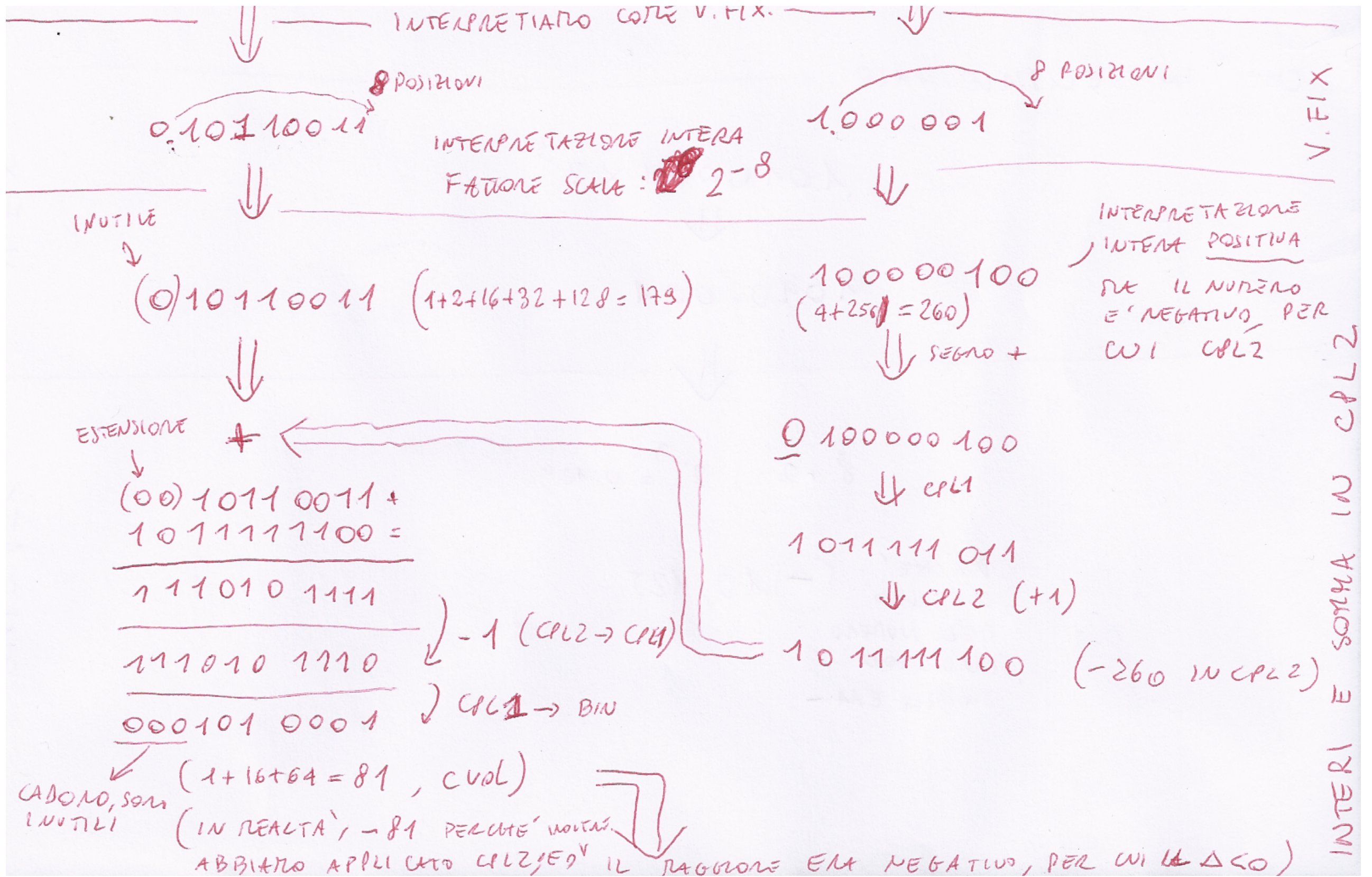
$$m = 000001$$

$$1,000001 \times 2^5$$

//

V. ROBINE

Es1: Somma in virgola mobile



Es1: Somme in virgola mobile

AGGIUNDO RIPRENDEBAMO IL FATTORE DI SCALA (2^{-8})

$$1010001 \times 2^{-8}$$



$$0.01010001 \leftarrow 8$$

V. FIX

QUESTO E' IL RISULTATO DELLA DIFFERENZA. IL NUMERO 1, IN MAGGIORE TRA I DUE ORIGINALI E' IL SECONDO (2^5 , L'ALTRO Ossia IL PRIMO-MA 2^4) ED HA SEGNO NEGATIVO. ORA,

$$0.01010001 \times 2^5$$

NON E' NORMALIZZATO; NORMALIZZATO PER TORNARE NEL FORMATO CORRETTO

$$0.01010001 \times 2^5 \Rightarrow 1.010001 \times 2^3 \quad (5-2)$$

(1. m $\times 2^{e-127}$)

V. MOBILE

QUINDI IL RISULTATO E'

$$S = 1 \quad (\text{NEGATIVO})$$

$$E = 3 + 127 = 130$$

$$M = 010001$$

Es1: Somma in virgola mobile

CHE IN DECIMALE VALE

$$1.010001 \times 2^3$$

⇓

$$1010.001$$

⇓

$$8 + 2, 2^{-3} = 0.125$$

PERCHÉ
IL SEGNO
DEL NUMERO
IN VIRGOLA
MOBILE È -

BIN 2 DEC

BIN 1 X

Quinta esercitazione: agenda

- (20') Es1: somma virgola fissa e mobile
- (20') Selezione esercizi da prova di esame 1
- (20') Selezione esercizi da prova di esame 2
- (10') Pausa
- (60') Es2: Spotted:InfoB

Selezione esercizi da prova di esame 1

<i>A</i>	<i>B</i>	<i>C</i>	<i>NOT B</i>	<i>A AND NOT B</i>	<i>NOT (A AND NOT B)</i>	<i>NOT B OR C</i>	<i>NOT (A AND NOT B) AND (NOT B OR C)</i>
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

Selezione esercizi da prova di esame 1

<i>A</i>	<i>B</i>	<i>C</i>	<i>NOT B</i>	<i>A AND NOT B</i>	<i>NOT (A AND NOT B)</i>	<i>NOT B OR C</i>	<i>NOT (A AND NOT B) AND (NOT B OR C)</i>
0	0	0	1	0	1	1	1
0	0	1	1	0	1	1	1
0	1	0	0	0	1	0	0
0	1	1	0	0	1	1	1
1	0	0	1	1	0	1	0
1	0	1	1	1	0	1	0
1	1	0	0	0	1	0	0
1	1	1	0	0	1	1	1

Quinta esercitazione: agenda

- (20') Es1: somma virgola fissa e mobile
- (20') Selezione esercizi da prova di esame 1
- (20') Selezione esercizi da prova di esame 2
- (10') Pausa
- (30') Selezione esercizi da prova di esame 3
- (40') Es2: Spotted:InfoB

Selezione esercizi da prova di esame 1

Si consideri ora la condizione, scritta in linguaggio C, in cui **x** e **y** siano due variabili int:

!((x>3) && !(y>5)) && (!(y>5) || (x<2))

ottenuta dalla prima formula sostituendo la variabile A con **x>3**, la variabile B con **y>5**, la variabile C con **x<2**

Si risponda alle seguenti domande:

- L'espressione e' vera o falsa quando x=1 e y=7? (giustificare la risposta)
- Se y>5, per quali valori di x l'espressione e' vera? (giustificare la risposta)

Selezione esercizi da prova di esame 1

!((x>3) && !(y>5)) && (!(y>5) || (x<2))

Con A==(x>3), B==(y>5), C==(x<2)

Risposta 1 - hp: x=1, y=7

!((1>3) && !(7>5)) && (!(7>5) || (1<2))

!(F && !(T)) && (! (T) || (T))

! (F && F) && (F || T)

!F && T

T && T

T

Selezione esercizi da prova di esame 1

!((x>3) && !(y>5)) && (!(y>5) || (x<2))

Con A==(x>3), B==(y>5), C==(x<2)

Risposta 2 - hp: y>5

!((x>3) && !(T)) && (!(T) || (x<2))

!((x>3) && F) && (F || (x<2))

! (F) && (F || (x<2))

T && x<2

vera iif x<2

Quinta esercitazione: agenda

- (20') Es1: somma virgola fissa e mobile
- (20') Selezione esercizi da prova di esame 1
- (20') Selezione esercizi da prova di esame 2
- (10') Pausa
- (60') Es2: Spotted:InfoB

Selezione esercizi da prova di esame 2

Il Triangolo di Tartaglia è una disposizione geometrica a forma di triangolo dei coefficienti binomiali. Un esempio di triangolo di Tartaglia di 5 righe è riportato qui sotto:

```
  1
 1 1
1 2 1
1 3 3 1
1 4 6 4 1
```

Il triangolo di Tartaglia può essere rappresentato in C con una matrice triangolare come mostrato qui sotto

1				
1	1			
1	2	1		
1	3	3	1	
1	4	6	4	1

Date le seguenti definizioni:

```
#define N 10
int tartaglia[N][N];
```

si scriva il frammento di programma che, per ogni riga r (con $0 \leq r < N$) della matrice `tartaglia`, verifichi che la somma degli elementi di tale riga sia pari a 2^r e stampi a schermo un messaggio contenente l'esito della verifica.

Quinta esercitazione: agenda

- (20') Es1: somma virgola fissa e mobile
- (20') Selezione esercizi da prova di esame 1
- (20') Selezione esercizi da prova di esame 2
- (10') Pausa
- (60') Es2: Spotted:InfoB

Es2. Spotted:InfoB

Si ipotizzi di volere realizzare un software per la gestione di un sistema alla “**Spotted:Polimi**” (sapete cos'è, vero :P?)

Nelle prossime slide, uno alla volta, i requisiti del sistema!

Es2. Spotted:InfoB

Prima di tutto, abbiamo bisogno di un *server* (ossia una macchina su cui il vostro software “gira”, e permette agli utenti di accedere al servizio)

Un **server**, per semplicità, é una struttura dati che memorizza **utenti**, **spottate** e **risposte**. Definite degli opportuni campi in modo tale che 1) un utente *U* sia identificato univocamente sul sistema 2) una spottata *S* possa avere al più *R* risposte, e per ogni spottata si sappia quante e quali risposte sono state fornite 3) le risposte hanno un punteggio positivo (numero di *like*) e negativo (numero di *dislike*). 4) ogni risposta é associata ad un utente identificato univocamente sul sistema. 5) le spottate hanno un identificatore univoco

Es2. Spotted:InfoB

Il server realizza un ciclo di controllo in cui si chiede all'amministratore (che ipotizziamo riceva spottate e risposte per email) di **inserire una spottata** (tasto "i" - il sistema deve anche generare un codice univoco per la spottata, ricordatevi del requisito 5) **visualizzare una spottata e le sue risposte** (tasto "v" - il sistema chiede il numero della spottata e se non esiste fornisce un messaggio di errore), **inserire una risposta** (tasto "r" - il sistema deve anche chiedere rispetto a quale spottata inserire la risposta, e se non esiste fornisce un errore), **inserire un utente** (tasto "u" - il sistema genera un ID e chiede di inserire nome e cognome).

(Se questa cosa dell'inserimento attraverso email vi fa ridere é perché non conoscete la storia dell'informatica :P)

Es2. Spotted:InfoB

Il sistema é pronto!

Ora dobbiamo cercare di estrarre *conoscenza* dai dati grezzi!
Per esempio: definiamo **troll** un utente la cui somma di *dislike* superi la somma di *like*. Se avete generato sensatamente gli identificatori univoci degli utenti, questa operazione si può fare in maniera molto *smart*.

Aggiungete al ciclo di controllo precedente un tasto (“t”) per visualizzare i dati dei *troll* nel sistema (il sistema ritorna “nessun troll” se non vi sono *troll* nel sistema).

Es2. Spotted:InfoB

Ipotizziamo adesso di cercare di calcolare delle **statistiche** sui dati inseriti dagli utenti.

Per esempio, potrebbe essere interessante valutare la differenza nella frequenza con cui gli ingegneri (spotted:polimi-ing) e gli xxx-a-caso (spotted:nota-uni-milanese-a-caso) parlano di un dato argomento, allo scopo di pianificare una campagna reclutamento studenti dei primi anni (cioè, *voi*) più adatta alla popolazione.

É risaputo infatti che un messaggio tipo “Vi spaccherete la schiena su travi, byte e manovellismi” fa molto più presa sui futuri ingegneri del Polimi rispetto a “Vi spaccherete la schiena a forza di giocare a limbo alle feste universitarie”.

Lo scopo di questo codice é estrapolare nuovi “argomenti-marketing” a partire dagli interessi degli ingegneri. E Facebook/Google/similia hanno *praticamente tutti* i vostri dati, per cui non pensiate che questo tipo di esercizio sia pura fantasia.

Es2. Spotted:InfoB

Aggiungete al ciclo di controllo precedente un tasto (“s”) per visualizzare le statistiche.

Una volta premuto il tasto “s” il sistema richiede all’utente di inserire una *stringa* (la parola di cui cercare la *frequenza* nel testo). Nel codice, ipotizzate di avere a disposizione una funzione `int contiene(...)` che prende una stringa e restituisce N se la stringa é presente come parola singola in una stringa più lunga separata da spazi (ossia, il testo di una spottata o risposta), esattamente N volte. Inoltre, supponete di avere una funzione `int numeroParole(...)` che prende una stringa e restituisce il numero di parole separate da segni di punteggiatura e spazi, senza considerare le parole troppo corte per avere un significato proprio (tipo: e, a, ad, con, per ...). Viene restituito il rapporto `numeroVolte/numeroParole` interpretato come *float*, con `numeroVolte` il numero di volte che la stringa compare in risposte o spottate.