Esercizi MIPS - Architettura degli Elaboratori

Lucian D. Crainic - Franco Liberati

Repository GitHub con possibili soluzioni.

Contents

1 Capitolo 1

1.1 Traccia 1

Implementare il programma somma di due numeri.

1.2 Traccia 2

Implementare il seguente programma: $0=1 se il valore della variabile Batman (definita in memoria) è maggiore del valore della variabile Robin (definita in memoria).

1.3 Traccia 3

Implementare il seguente programma: $t0=1 se il valore della variabile Memole, definita in memoria, ha alla terza posizione meno significativa un 1.

1.4 Traccia 4

Descrivere l'algoritmo che dato un numero intero maggiore di 2 (definito in memoria) stabilisca se il numero è primo (valore 1 in $t2) o no (valore 2 in $t2). Provare ad implementare il programma utilizzando l'emulatore MARS.

Esempio numeri primi 1,3,5,7,11,13,...

PS: un numero è primo solo se è divisibile per se stesso e per 1.

1.5 Traccia 5

Si scriva un programma in linguaggio assembly MIPS che legga da tastiera i valori delle lunghezze dei tre lati di un triangolo e determina:

se il triangolo è scaleno (porre $t0=0)

se il triangolo è rettangolo (porre $t0=1)

se il triangolo è isoscele (porre $t0=2)

se il triangolo è equilatero (porre $t0=3)

NB: non è necessario usare la radice quadra né i numeri in virgola mobile

1.6 Traccia 6

Si scriva un programma in linguaggio assembly MIPS che acquisisca tre numeri interi da tastiera e metta in $t0 del valore maggiore.

1.7 Traccia 7

Si scriva un programma in linguaggio assembly MIPS che legga da tastiera una serie di numeri interi fino a quando la somma di tutti i numeri introdotti fino a quel momento non supera il valore 1000. A quel punto, il programma stampa il valore del prodotto di tutti i numeri inseriti.

NB:la stampa deve avvenire da consolle output dell'emulatore MARS

1.8 Traccia 8

Un utente inserisce da tastiera una serie di numeri interi positivi, ed il termine della serie è indicato dall'inserimento del valore -1. Il programma in linguaggio assembly MIPS, al termine dell'inserimento, stampa quanti numeri in totale sono stati inseriti

NB:la stampa deve avvenire da consolle output dell'emulatore MARS

1.9 Traccia 9

Si scriva un programma in linguaggio assembly MIPS per il calcolo dei quadrati perfetti per una sequenza di numeri. Il programma deve prima leggere un numero inserito da tastiera, e quindi stampare i primi quadrati perfetti sino al quadrato del numero.

ES:

INPUT=5

OUTPUT=1,4,9,16,25

1.10 Traccia 10

Un utente introduce da tastiera due numeri interi (INIT e LUNG). Il programma in linguagguo assembly MIPS deve stampare una serie di numeri interi consecutivi. La serie inizia al valore INIT ed è lunga LUN elementi.

Esempio:

INPUT:

7

4

OUTPUT: 7 8 9 10

1.11 Traccia 11

Si scriva un programma in linguaggio assembly MIPS che stampa 1 se, in una sequenza di dieci numeri inseriti da tastiera, ci sono almeno due o più numeri consecutivi uguali.

NB:la stampa deve avvenire da consolle output dell'emulatore MARS. In caso di esito negativo stampare -1

1.12 Traccia 12

Si scriva un programma in linguaggio assembly MIPS per calcolare il minimo comune multiplo (MCM) di due numeri interi positivi immessi da tastiera. Stampare il MCM. Dati due numeri interi N1 e N2, il minimo comune multiplo è il più piccolo numero M che è divisibile (con resto pari a zero) sia per N1 che per N2.

Suggerimento. Si considerino due numeri interi N1 e N2. Sia N1 più grande di N2. Il MCM è il primo multiplo di N1 che è divisibile (con resto uguale a zero) per N2.

NB:la stampa deve avvenire da consolle output dell'emulatore MARS.

1.13 Traccia 13

Si scriva un programma in linguaggio assembly MIPS per calcolare il massimo comun divisore (MCD) di due numeri interi positivi. Il MCD è definito come il massimo tra i divisori comuni ai due numeri. Stampare il MCD.

Suggerimento. Si considerino due numeri interi N1 e N2. Il MCD di N1 e N2 è il massimo tra i numeri che sono divisori (con resto uguale a zero) sia di N2 che di N1. In particolare, si supponga che sia N1 minore di N2.

Il MCD è il massimo tra i numeri compresi tra 1 e N1 che sono divisori (con resto uguale a zero) sia di N1 che di N2.

2 Capitolo 2

2.1 Traccia 1

Implementare un programma in linguaggio assembly MIPS che:

stampa a video “Primo numero: ” e prenda in input un numero a

stampa a video “Secondo numero: ” e prenda in input un numero b

stampa a video “Prodotto dei due numeri: ” e stampi a video axb

Esempio

OUTPUT: Primo Numero:

INPUT:5

OUTPUT:Secondo Numero:

INPUT:6

OUTPUT: Prodotto dei due numeri:30

2.2 Traccia 2

Implementare un programma in linguaggio assembly MIPS che legga da input un intero positivo *dividendo* (word) ed un intero positivo (word) *divisore* e restituisca in output il quoziente e resto della divisione a/b.

Esempio

INPUT (dividendo): 56

INPUT (divisore): 23

OUTPUT: Quoziente: 2 Resto:10

2.3 Traccia 3

Implementare un programma in linguaggio assembly MIPS che legga da input un intero e calcoli il numero di 1 della sua rappresentazione binaria.

Esempio

INPUT: 521 (in binario 1000001001)

OUTPUT:3

3 Capitolo 3

3.1 Traccia 1

Implementare un programma in linguaggio assembly MIPS che dati cinque interi positivi definiti in memoria calcola la media aritmetica

Esempio

INPUT: a=0,b=11;c=7;d=1982;e=10051980

OUTPUT:2010796

3.2 Traccia 2

Implementare un programma in linguaggio assembly MIPS che legga da input un intero positivo a>2(word) ed un intero positivo b>1(word) e ne restituisca in output il prodotto a*b. senza utilizzare l'istruzione mul.

Esempio:

INPUT (a): 10

INPUT (b): 5

OUTPUT: 50

3.3 Traccia 3

Dato il seguente programma in linguaggio assembly MIPS, determinare il valore di risb, rish, risw

3.4 Traccia 4

Dato un intero positivo a definito in memoria, stampare a video "Terzo bit 1" nel caso in cui il terzo bit del numero a sia 1 o "Terzo bit 0" altrimenti.

3.5 Traccia 5

Dato a intero positivo (da 0 a 255) inserito da tastiera, scrivere il valore binario di a al contrario. Esempio:

INPUT (a): 5 (cioè 00000101)

OUTPUT: 160 (10100000)

INPUT (a): 105 (cioè 01101001)

OUTPUT: 150 (10010110)

3.6 Traccia 6

Confrontare due interi positivi a e b, definiti in memoria, e mettere in $t0 il valore 0 se a e'maggiore di b, 1 altrimenti. Non è possibile utilizzare l'istruzione di comparazione tra valori: operare sui singoli bit dei valori.

4 Capitolo 4

4.1 Traccia 1

Leggere 7 valori interi da input e calcolarne la media (stampandola a video).

INPUT

12;

82;

11;

2345;

67;

123456;

675

OUTPUT

126069.428571428

NB: La media deve essere espressa con un numero reale (float). Utilizzare il coprocessore matematico.

4.2 Traccia 2

Effettuare la sommatoria di numeri reali poistivi immessi da input. La sommatoria è calcolata quando il valore immesso dall'utente è nullo o negativo.

INPUT

3.5;7.23;5.6;9.17;-1

OUTPUT

25.5

4.3 Traccia 3

L'ISTAT ha rivisto le stime dell'aspettativa di vita (life expentancy, LE) della popolazione italiana in 84.8 per le donne e 80.5 per gli uomini.

Realizzare un programma in assembly MIPS che acquisca da input un carattere F o M per discriminare il genere di un campione e un valore intero che rappresenta l'età e determinare se il campione immesso ha superato la media oppure no. Il programma termina quando l'utente inserisce il carattere X.

ESEMPIO:

F;85;

M;80;

M;82;

F;45;

X

OUTPUT

OLTRE LA MEDIA LE; SOTTO LA MEDIA LE; OLTRE LA MEDIA LE; SOTTO LA MEDIA LE

5 Capitolo 5

5.1 Traccia 1

Scrivere un programma in linguaggio assembly MIPS che riceve in ingresso una sequenza di N numeri interi. I numeri sono memorizzati in un vettore. Il valore N è inserito dall'utente, ma il vettore può contenere al massimo 30 numeri. Terminato l'inserimento della sequenza di numeri, il programma deve verificare se gli elementi del vettore sono tutti uguali tra loro.

5.2 Traccia 2

Un programma in linguaggio assembly MIPS deve leggere dall'utente due vettori di 5 numeri interi ciascuno. Il programma deve creare un ulteriore vettore, che contenga la copia dei soli elementi pari presenti nei due vettori di partenza, e stampare tale vettore.

5.3 Traccia 3

Un programma in linguaggio assembly MIPS deve inizializzare quindici valori interi e calcolare la media aritmetica (si deve usare il coprocessore matematico) degli elementi alla posizioni pari, alla posizioni dispari e quella complessiva.

5.4 Traccia 4

Scrivere un programma in linguaggio assembly MIPS che riceve in ingresso una sequenza di N numeri interi. I numeri sono memorizzati in un vettore. Il valore N è inserito dall'utente prima della lettura del vettore, ma il vettore può contenere al massimo 30 numeri. Terminato l'inserimento della sequenza di numeri, il programma deve verificare se il vettore contiene una sequenza di numeri ordinata in modo strettamente crescente.

ESEMPIO:

INPUT 5 - 3;5;8;10;22

OUTPUT: ORDINAMENTO STRETTAMENTE CRESCENTE

INPUT 5 - 3;5;8;22;10

OUTPUT: ORDINAMENTO CASUALE

5.5 Traccia 5

Scrivere un programma in linguaggio assembly MIPS che legga una stringa introdotta da tastiera. La stringa contiene sia caratteri maiuscoli che caratteri minuscoli, e complessivamente

al più 255 caratteri. Il programma deve svolgere le seguenti operazioni:

- Visualizzare la stringa inserita. - Stampare a video l'uppercase della stringa.

Ad esempio la frase "Che Bella Giornata" diviene "CHE BELLA GIORNATA".

5.6 Traccia 6

Scrivere un programma in linguaggio assembly MIPS che legga una stringa introdotta da tastiera. La stringa contiene sia caratteri maiuscoli che caratteri minuscoli, e complessivamente al più 100 caratteri. Il programma deve svolgere le seguenti operazioni:

- Visualizzare la stringa inserita. - Costruire una nuova stringa in cui il primo carattere di ciascuna parola nella frase di partenza è stato reso maiuscolo. Tutti gli altri caratteri devono essere resi minuscoli. Il programma deve memorizzare la nuova stringa. - Visualizzare la nuova frase.

Ad esempio la frase "cHe bElLA gIOrnaTa" diviene "Che Bella Giornata".

6 Capitolo 6

6.1 Traccia 1

Implementare in linguaggio asembly MIPS un programma che calcola il massimo tra n elementi immessi in input (la lettura termina quando si introduce un numero negativo. Utilizzare la sub-routine (funzione) MASSIMO che presi due elementi restituisce il massimo.

Esempio:

INPUT:

45; 66; 34; 156; 233; 234; 56; 0 ; -11

ANALISI

MASSIMONUM(45, 66, 34, 156,233,234,56,0,-11)=234

MASSIMO(MASSIMO(MASSIMO
(MASSIMO(MASSIMO(MASSIMO(MASSIMO
(45,66),34),156),233),234),56),0)=234

OUTPUT:

234

6.2 Traccia 2

Nella battaglia virtuale STUDENT BATTLE la milizia A composta da 100 informatici si posiziona ad una certa distanza dalla milizia B di 100 sociologhi. Gli informatici hanno un mortaio con cui sparare delle bombe di colore che colpiscono 10 persone e un elaboratore MIPS; mentre la milizia dei sociologhi dispone di libri di guerra e di una imponenente bomba ad acqua fredda che può colpire 100 persone.

Il generale comanda alla truppa degli informatici di muoversi ogni due minuti di dieci metri dalla distanza di 123 metri iniziali per evitare di essere colpiti in pieno dalla bomba ad acqua dei sociologhi e nel contempo chiede ai suoi capitani di realizzare un programma MIPS per determinare, in tempi rapidi, la velocità necessaria per far percorrere alle bombe di colore la giusta distanza (supponendo che il mortaio abbia una inclinazione fissa di 45°) al variare dei propri spostamenti.

Il calcolo della distanza della traiettoria deve essere realizzato con la funzione GITTATA (vo^2/g).

NB: $g = 9,823 m/s^2$.

Esempio:

Distanza:123 metri

Calcolo velocità:...

Numero vittime sociologhi:90

...

Ditanza:133 Calcolo velocità:...

Numero vittime sociologhi:80

...

fino ad esaurimento dei sociologhi

PS: per rendere il gioco interessante usare la syscall random per determinare degli spostamenti generici.

7 Capitolo 7

7.1 Traccia 1

Implementare un programma in linguaggio assembly MIPS che definiti R (numero di righe) e C (numero di colonne) rispettivamente in $t0 e $t1, stampa in maniera leggibile e conforme alla struttura tabellare una matrice M_RxC definita in memoria.

7.2 Traccia 2

Trasformare il programma nella Traccia 1 in una macro.

7.3 Traccia 3

Definita in memoria una matrice di 4 righe e 4 colonne con elementi byte, stampare in output la somma degli elementi presenti lungo una colonna

ESEMPIO:

MEMORIA

3 6 7 8

1 5 2 0

6 8 10 5

4 1 -9 2

OUTPUT

COLONNA 1: 14

COLONNA 2: 20

COLONNA 3: 10

COLONNA 4: 15

7.4 Traccia 4

Definita una matrice in memoria di 4 righe e 3 colonne A_{4x3} con elementi word, stampare in output la matrice trasposta $A^t_{3\cdot4}$. La matrice trasposta A^t è costituita dagli elementi alla posizione inversa della matrice originale A: cioè $A(a_{i,j})$ si trova in $A^t(a_{i;j})$.

ESEMPIO:

A

12 74 06 07

99 10 11 16

00 03 20 21

A^t

12 99 00

74 10 03

06 11 20

07 16 21

7.5 Traccia 5

Definita una matrice in memoria di 8 righe e 8 colonne con elementi halfword,$A_{8\cdot8}$, stampare in output una nuova matrice $B_{8\cdot4}$ in cui le colonne sono date dal prodotto degli elementi delle colonne della matrice originaria: cioè $B_{4\cdot8}b_{1,1} = a_{1,1} \cdot a_{1,2}; b_{1,2} = a_{1,3} \cdot a_{1,4}; b_{1,3} = a_{1,5} \cdot a_{1,6}; b_{1,4} = a_{1,7} \cdot a_{1,8}$.

ESEMPIO :

A

02 04 06 07 00 12 03 08

01 10 05 16 00 01 01 10

00 03 20 21 01 01 02 04

02 22 06 00 00 12 37 00

30 50 01 34 00 05 04 13

10 63 08 08 01 06 05 03

05 04 00 01 00 09 06 02

41 00 14 02 00 14 00 01

B

0008 0042 0000 0024

0010 0080 0000 0010

0000 0421 0001 0008

0044 0000 0000 0000

1500 0340 0000 0052

0630 0064 0006 0015

0020 0000 0000 0012

0000 0028 0000 0000

7.6 Traccia 6

Definita una matrice in memoria di N righe e N colonne con elementi halfword, $A_{N \cdot N'}$, stampare in output il messaggio con la scritta IDENTITA' se la matrice + la matrice identità o NON IDENTITA' altrimenti. Una matrice identità è una matrice costituita da elementi 0 eccetto quelli sulla diagonale principale che hanno valore 1.

ESEMPIO:

A

1 0 0

0 1 0

0 0 1

OUTPUT

IDENTITA'

ESEMPIO:

A

1 0 0

0 1 0

1 0 1

OUTPUT

NON IDENTITA'

8 Capitolo 8

8.1 Traccia 1

Si scriva un programma in linguaggio assembly MIPS che definiti due INSIEMI (cioè due vettori halfword) opera su di essi con tre subroutine UNIONE, INTERSEZIONE, DIFFERENZA.

In particolare:

- UNIONE ha in input "i due insiemi" (cioè gli indirizzi dei vettori) e restituisce un "nuovo insieme" (cioè l'indirizzo iniziale del vettori) che riporta l'unione dei due insiemi [NB: L'unione fra due insiemi è l'operazione che associa ai due insiemi l'insieme i cui elementi appartengono al primo oppure al secondo insieme].
- INTERSEZIONE ha in input "i due insieme" (cioè gli indirizzi dei vettori) e restituisce un "nuovo insieme" (cioè l'indirizzo iniziale del vettori) che riporta l'intersezione de due insiemi [NB: L'intersezione fra due insiemi è l'operazione che associa ai due insiemi l'insieme i cui elementi appartengono contemporaneamente al primo e al secondo insieme].
- DIFFERENZA ha in input "i due insiemi" (cioè gli indirizzi dei vettori) e restituisce un "nuovo insieme" (cioè l'indirizzo iniziale del vettori) he riporta la differenza dei due insiemi.[NB: Si definisce differenza fra due insiemi l'insieme degli elementi del primo insieme che non appartengono al secondo insieme;]

ESEMPIO:

Dati gli insiemi

A = 1, 2, 3, 4

B = 3, 4, 5, 6

AunioneB = 1, 2, 3, 4, 5, 6

AintersezioneB = 3, 4

AdifferenzaB = 1, 2

8.2 Traccia 2

Si scriva un programma in linguaggio assembly MIPS che legge due stringhe da tastiera e al suo interno usa una subroutine, denominata SIMILITUDINE, che valuta quanti caratteri alle stesse posizioni delle due stringhe sono uguali.

La routine riceve due parametri, "le stringhe" (cioè gli indirizzi delle stringhe), e restituisce un numero intero e lo stampa a video.

Ad esempio:

- SIMILITUDINE ("ciao", "cielo") restituisce 2 in quanto i primi due caratteri sono identici.
- SIMILITUDINE("ciao", "salve") restituisce 0 in quanto nessun carattere alle stesse posizioni sono uguali.

8.3 Traccia 3

Si scriva un programma in linguaggio assembly MIPS che mediante la subroutine ORDINA ordina un vettore immesso da input. La routine ORDINA ha come argomento il vettore (cioè l'indirizzo del vettore), e restituisce il vettore ordinato (l'indrizzo del vettore, con gli elementi ordinati).

Ad esempio

v=(1,10,6,3,2,4)

ORDINA(v)=(1,2,3,4,6,10)

9 Capitolo 9

9.1 Traccia 1

Scrivere un programma che inizializza un vettore di 10 elementi a 16bit con valori casuali compresi tra 0 e 65000 e che copia in un nuovo vettore il quadrato degli elementi del primo (utilizzare una funziona per realizzare il quadrato degli elementi).

ESEMPIO:

v1= 5,60000,0,1,45,76,99,456,4321,12876

v2= 25,3600000000,0,1,2025,5776,9801,207936,18671041,165791376

9.2 Traccia 2

Inizializzare, con valori casuali, un vettore di dimensione 10 con elementi word e realizzare la funzione

INS(vettore, DIM, ELEM, POS)

che inserisce l'elemento ELEM (immesso da tastiera) alla posizione POS (immessa da tastiera) slittando a destra gli elementi successivi alla posizione di inserimento

ESEMPIO:

v1=555,6,710,33,6071,789,5,-67,99,1000

lunghezzav1=10

INS(v1,10,2312,6)

v1=555,6,710,33,6071,2312,789,5,-67,99,1000

lunghezzav1=11

PS: per evitare la gestione del caso in cui l'utente inserisca una posizione al di fuori dal range [0,9] si consiglia di stamapre su videoterminale un segnale di avvertimento

9.3 Traccia 3

Si scriva un programma che definisca in memoria due vettori: V1 un vettore di cinque elementi di byte e V2 un vettore di cinque elementi di halfword.

Si applichi ala programma una funzione:

SOMMA(V1,V2,V3)

Che ha come argomenti i due vettori definiti dall'utente V1 e V2 e restituisce il vettore V3 formato dall'elemento più grande alla stessa posizione dei vettori V1 e V2.

ESEMPIO:

V1=3,56,12,45,33

V2=-4,67,89,11,47000

V3=3,67,89,45,47000

9.4 Traccia 4

Creare una funzione int DIVISORE (int n) che calcola quanti divisori ha un numero naturale n.

Creare poi un programma che ricevuto dall'utente un numero naturale n stampi il numero d che indica quanti divisori ha n.

ESEMPIO:

READ(x) //immessione di x=33

Y=DIVISORE(X)

PRINT(Y) //risultato 2 (cioè 3 e 11)

9.5 Traccia 5

Creare una funzione che ricevuti tre importi di denaro sposta gli eventuali debiti (si considerino debiti gli importi negativi) sul primo importo STOCK(int a, int b, int c).

Creare un main per testare la funzione:

ESEMPIO

X=5;

Y=-1;

Z=-2;

STOCK (X,Y,Z);

Print(X); //stampa 2

Print(Y); //stampa 0

Print(Z); //stampa 0

9.6 Traccia 6

Creare una funzione SUBFMAX(int val1, int val2) che ricevuti due valori sottrae al maggiore metà del valore del minore e divida per tre il minore.

Creare una programma che applica tre volte la funzione a valori inseriti dall'utente.

ESEMPIO:

x=30;

y=56;

SUBFMAN(x,y); dopo l'esecuzione: x=10 y=41

SUBFMAN(x,y); dopo l'esecuzione: x=3 y=36

SUBFMAN(x,y); dopo l'esecuzione: x=1 y=35

PRINT(x); //stampa 1

PRINT(y); //stampa 35

10 Capitolo 10

10.1 Traccia 1

Simulare il gioco di carta, forbice e sasso di due giocatori sapendo che:

1. la carta batte il sasso.
2. il sasso batte la forbice.
3. la forbice batte la carta.
4. il gioco termina dopo 10 lanci.
5. carta, sasso e forbice sono determinati in maniera casuale. il MIPS consente di generare un numero INTERO casuale mediante la syscall 42 in $v0 (il numero casuale generato si trova in $a0 dopo la chiamata a sistema). NB: prima della chiamata a syscall è possibile impostare ad n il registro $a1 per generare i valori interi inclusi tra 0 e n-1. Ad esempio impostando a 6 il registro $a1, dopo la syscall in $a0 si ha un numero compreso tra 0 e 5. In questo modo è possibile pensare di assegnare al sasso, forbice e carta un valore numerico che poi dovrà essere confrontato in base alle regole a), b), c).

Mostrare per ogni iterazione il segno prescelto dai concorrenti e riportare, alla fine del gioco, il nome del vincitore.