

Sistemi di Calcolo (A.A. 2023-2024)

Corso di Laurea in Ingegneria Informatica e Automatica
Sapienza Università di Roma

C

Compito (13/06/2024) – Durata 1h 30'

Inserire nome, cognome e matricola nel file `studente.txt`.

ISTRUZIONI PER STUDENTI DSA: svolgere a scelta due parti su tre.

Parte 1 (programmazione IA32)

È estremamente comune nel mondo dell'informatica effettuare conversioni di numeri, da rappresentazione esadecimale a decimale. In questo esercizio, si chiede di tradurre in assembly una funzione che converte una stringa esadecimale in un array di interi e restituisce la somma degli interi positivi letti. Nella directory E1, si traduca in assembly IA32 la seguente funzione C scrivendo un modulo `e1C.s`:

```
int hextodec_sum(unsigned char *in, int *out)
{
    int sum = 0;
    unsigned i = 0;
    int res;

    while (*in) {
        hextodec_helper(in, &res);

        if (res > 0) {
            sum += res;
        }

        out[i++] = res;
        in += 2;
    }

    return sum;
}
```

L'unico criterio di valutazione è la correttezza. Generare un file eseguibile `e1C` con `gcc -m32 -g`. Per i test, compilare il programma insieme al programma di prova `e1C_main.c`.

Non modificare in alcun modo `e1C_main.c`. Prima di tradurre il programma in IA32 si suggerisce di scrivere nel file `e1C_eq.c` una versione C equivalente più vicina all'assembly.

Parte 2 (programmazione di sistema POSIX)

Si scriva una funzione che aiuti il gestore di una agenzia di viaggi a tenere in ordine il proprio catalogo di destinazioni per villeggiatura. Nello specifico, si scriva nel file `E2/e2C.c` una funzione con il seguente prototipo:

```
void destinazioniSottoBudget(const char * filesorgente, const char *
partenza, int budget, const char * filedestinazione)
```

che, dato in input il nome `filesorgente` di un file contenente informazioni sulle connessioni disponibili, una stringa `partenza` ed un intero `budget`, scrive in un secondo file `filedestinazione` la lista di tutte le destinazioni raggiungibili da `partenza` con un costo

inferiore o uguale a `budget`. Per raggiungibile si intende tramite una connessione specifica tra località di partenza e di destinazione (ossia non vanno valutate eventuali connessioni attraverso località intermedie).

Il file `filesorgente` tiene traccia delle connessioni disponibili con righe aventi il seguente formato

DEF-UVZ-130

Ogni riga nel file rappresenta una connessione tra un punto di partenza, una destinazione ed il relativo costo. Le tre informazioni sono separate dal carattere '-'. Nell'esempio, partendo da DEF è possibile raggiungere UVZ con un costo di 130€.

Le connessioni scritte nel file `filedestinazione` devono utilizzare lo stesso formato ed apparire in ordine di costo crescente.

Per i test, compilare il programma insieme al programma di prova `e2C_main.c` fornito, che **non** deve essere modificato. Nota: non modificare il file `booked.txt` che riporta un esempio di file contenente alcune connessioni.

Parte 3 (quiz)

Si risponda ai seguenti quiz, inserendo le risposte (A, B, C, D o E per ogni domanda) nel file `e3A.txt`. Una sola risposta è quella giusta. Rispondere E equivale a non rispondere (0 punti).

Domanda 1 (Assembly)

Quale delle seguenti serie di istruzioni IA32 consente di copiare il valore di una variabile di tipo puntatore (situata sullo stack all'indirizzo `esp + 20`) in una locazione di memoria puntata dal registro `ecx`?

A	<code>movl 20(%esp), (%ecx)</code>	B	<code>movl 20(%esp), %ecx</code> <code>push %ecx</code>
C	<code>movl 20(%esp), %eax</code> <code>movl (%eax), %ecx</code>	D	Nessuna delle precedenti

Motivare la risposta nel file `M1.txt`. **Risposte non motivate saranno considerate nulle.**

Domanda 2 (pipelining)

Si consideri la seguente sequenza di istruzioni:

```
movl $5, %eax
incl %ebx
addl %ecx, %eax
subl %ebx, %eax
incl %edx
```

Considerando una semplice pipeline a 5 stadi (Fetch, Decode, Execute, Memory, Write-Back) per completare tutte le istruzioni e assumendo che gli hazard vengano risolti con stalli, quale delle seguenti affermazioni è **vera**?

A	Sono richiesti 20 cicli di clock per completare le istruzioni	B	Sono richiesti 14 cicli di clock per completare le istruzioni senza riordinarle
----------	---	----------	---

C	Riordinando istruzioni è possibile ridurre il numero di cicli di clock a 12.	D	Non è possibile riordinare le istruzioni senza cambiare la semantica
----------	--	----------	--

Motivare la risposta nel file M2.txt. **Risposte non motivate saranno considerate nulle.**

Domanda 3 (Processi)

Si consideri il seguente programma C.

```
#include <stdio.h>
#include <unistd.h>
int main(){
    int my_fork = fork();

    if( my_fork = fork() ){
        printf("fork\n");
    } else {
        printf("fork\n");
    }
    printf("fork\n");
}
```

Quali delle seguenti affermazioni è vera?

A	“fork” è stampato 6 volte	B	“fork” è stampato 8 volte
C	Nessuna delle precedenti	D	“fork” è stampato 3 volte

Motivare la risposta nel file M3.txt. **Risposte non motivate saranno considerate nulle.**

Domanda 4 (stack)

Si consideri una funzione `foo(int arg1, char* arg2)`. Nel prologo vengono salvati 3 registri callee-save e riservati 12 byte per ospitare una variabile locale e due argomenti di tipo short che verranno passati ad una funzione `bar(short s1, short s2)` chiamata nel corpo di `foo`. Volendo calcolare l'indirizzo della variabile locale al termine del prologo e copiarlo nel registro `ecx`, qual è l'istruzione **corretta**?

A	<code>leal 4(%esp), %ecx</code>	B	<code>leal 8(%esp), %ecx</code>
C	<code>movl %esp, %ecx</code>	D	<code>leal 6(%esp), %ecx</code>

Motivare la risposta nel file M4.txt. **Risposte non motivate saranno considerate nulle.**