

Sistemi di Calcolo (A.A. 2025-2026)

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



Compito (21/09/2026) – 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)

La scelta di una password robusta rappresenta un aspetto fondamentale per la sicurezza dei sistemi informatici e disporre di un metodo sistematico per valutarne la qualità consente di individuare password potenzialmente deboli e vulnerabili. In questo esercizio si richiede di tradurre in linguaggio assembly una funzione che, data in input una password, ne calcoli un punteggio di sicurezza. Il punteggio viene determinato sulla base della differenza tra caratteri consecutivi e della loro tipologia, distinguendo tra lettere, cifre e caratteri speciali. La funzione deve memorizzare, tramite il parametro puntatore `score`, il punteggio calcolato e restituire come valore di ritorno 1 o 0 se la password può essere considerata sicura o no. Una password è considerata sicura se il punteggio ottenuto è maggiore di 10. Nella directory E1, si traduca in assembly IA32 la seguente funzione C scrivendo un modulo `e1A.s`:

```
int password_security(const char* password, int* score){
    int previous = 0, type;
    *score = 0;
    while (*password) {
        char type(*password, &type);
        if (type != previous) {
            *score += type;
        }
        previous = type;
        password++;
    }
    return *score > 10;
}
```

L'unico criterio di valutazione è la correttezza. Generare un file eseguibile `e1A` con `gcc -m32 -g`. Per i test, compilare il programma insieme al programma di prova `e1A_main.c`.

Non modificare in alcun modo `e1A_main.c`. Prima di tradurre il programma in IA32 si suggerisce di scrivere nel file `e1A_eq.c` una versione C equivalente più vicina all'assembly.

Parte 2 (programmazione di sistema POSIX)

Si scriva una funzione che aiuti l'utente nella scelta della squadra che parteciperà alla prossima competizione nazionale di CyberChallenge.it. Nello specifico, si scriva nel file E2/e2A.c una funzione con il seguente prototipo:

```
void selezione(const char * input, const char * output, int
num_players)
```

che, dato in input il nome `input` del file contenente informazioni sui partecipanti generi un file il cui nome è in `output`. Il file `input` contiene una riga per ogni partecipante con il nickname e il punteggio ottenuto nelle 4 categorie delle CTF (ovvero "cat1", "cat2", "cat3", "cat4") nel formato

```
nickname punti_cat1 punti_cat2 punti_cat3 punti_cat4
```

I punteggi sono interi non negativi. Il programma deve calcolare il punteggio totale ottenuto da tutti i player in ogni categoria.

Il file di output generato deve contenere una riga per ogni categoria nel formato

`categoria,punteggio`

Le righe devono essere ordinate in ordine decrescente in base al punteggio. In caso di righe con lo stesso punteggio le stesse devono essere ordinate in base al nome della categoria in ordine alfabetico crescente.

Per i test, compilare il codice prodotto insieme al programma di prova `e2A_main.c` fornito, che **non** deve essere modificato. Nota: non modificare i file `test*.txt` che riportano esempi di file contenenti la lista partecipanti.

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 (Processi)

Si consideri il seguente frammento di codice in cui `x` è una variabile intera globale:

```
int x = 10;
int main() {
    pid_t pid = fork();
    if (pid == 0) {
        x = x + 5;
        return 0;
    } else {
        wait(NULL);
        printf("%d", x);
    }
    return 0;
}
```

Cosa stamperà il processo padre alla fine dell'esecuzione?

A	15	B	10
C	Il valore è indefinito poiché dipende dallo scheduling	D	Errore di segmentazione

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

Domanda 2 (Assembly)

Si consideri la seguente istruzione assembly:

```
leal 4(%eax, %ebx, 8), %edx
```

Assumendo che il registro `%eax` contenga il valore decimale 1000 e il registro `%ebx` contenga il valore decimale 10, quale valore decimale verrà scritto nel registro `%edx` dopo l'esecuzione dell'istruzione?

A	Il valore contenuto all'indirizzo di memoria	B	1024
C	1084	D	8044

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

Domanda 3 (Pipelining)

Si consideri la seguente sequenza di istruzioni che vengono eseguite da una CPU con pipeline a 5 stadi (Fetch, Decode, Execute, Memory, Write-Back)

```
1: movl %(esp), %ebx
2: addl $0x10, %eax
3: movl %(ebx), %ecx
4: addl $0x42, %edx
```

Quale tipo di hazard si verifica al momento dell'esecuzione dell'istruzione nella linea 3?

A	Hazard strutturale	B	Hazard sui dati
C	Hazard sul controllo	D	Non si verifica nessun hazard

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

Domanda 4 (Segnali)

In un sistema UNIX-like in quali condizioni un processo può riprendere l'esecuzione dopo aver ricevuto un segnale di tipo SIGSEGV?

A	Se il processo ha registrato un gestore per SIGSEGV con <code>sigaction</code> e ha i privilegi di root.	B	Solo se il kernel consente la propagazione del segnale a un thread secondario che può ignorarlo.
C	Se il gestore del segnale corregge la causa del fault e il contesto di esecuzione viene ripristinato correttamente.	D	Sempre, poiché SIGSEGV è un segnale recuperabile in qualsiasi condizione di accesso a memoria non valida.

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