

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

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

# B

### Compito (08/06/2026) – 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)

Nel campo della Data Science, la somma dei quadrati è una funzione chiave, utile a svolgere analisi statistiche su una serie di dati per misurarne la variazione, la dispersione e molto altro. In questo esercizio, si chiede di tradurre in assembly una funzione che, dato un insieme di numeri interi `data` e la sua dimensione `n`, calcola e ritorna la somma dei quadrati dei soli valori compresi nell'intervallo aperto definito dai valori di soglia `min` e `max`. Nella directory `E1`, si traduca in assembly IA32 la seguente funzione C scrivendo un modulo `e1B.s`:

```
int threshold_squares_sum(int* data, int n, int min, int max)
{
    int i, sum = 0;
    int res;

    for (i = 0; i < n; i++) {
        if (data[i] < max && data[i] > min) {
            square(data[i], &res);
            sum += res;
        }
    }

    return sum;
}
```

L'unico criterio di valutazione è la correttezza. Generare un file eseguibile `e1B` con `gcc -m32 -g`. Per i test, compilare il programma insieme al programma di prova `e1B_main.c`.

**Non** modificare in alcun modo `e1B_main.c`. Prima di tradurre il programma in IA32 si suggerisce di scrivere nel file `e1B_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/e2B.c` una funzione con il seguente prototipo:

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

che, dato il nome `input` del file contenente informazioni sui partecipanti genera 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 identificare i 4 top player sulla base del punteggio totale ottenuto, ovvero la somma dei punteggi ottenuti dal giocatore nelle 4 categorie.

Il file di output generato deve contenere una riga per ognuno dei 4 giocatori selezionati nel formato:

nickname,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 nickname in ordine alfabetico crescente.

Per i test, compilare il programma insieme al programma di prova `e2B_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 (Interrupt e System Call)

Cosa succede in un processo dopo che questo ha richiesto l'esecuzione di una system call?

<b>A</b>	Il processo termina	<b>B</b>	Il controllo passa al kernel
<b>C</b>	La cache viene svuotata	<b>D</b>	Il processo viene sempre messo in stato READY

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

---

#### Domanda 2 (Stack)

Si consideri il seguente codice:

```
f:
1000: movl    $4,%eax
1005: pushl    $0x7
1007: pushl    %eax
1008: call     g
100d: addl     $0x7,%eax
1010: addl     $0x8,%esp
1013: ret

g:
1014: movl     $0xa,%eax
1019: ret
```

Qual è lo stato dello stack frame di **f** durante l'esecuzione di **g**?

<b>A</b>	0x7 0x4 0xa	<b>B</b>	0x7 0x4 0x100d
----------	-------------------	----------	----------------------

<b>C</b>	0x7 0x4 0x1019	<b>D</b>	0x7 0x4 0x1019 0xa
----------	----------------------	----------	-----------------------------

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

---

### Domanda 3 (Layout di memoria)

Si consideri l'esecuzione del seguente frammento di programma:

```
char* input;

void foo(unsigned int max_len){
    input = (char*) malloc(max_len);
    fgets(input, max_len, 0);
    return strlen(input);
}
```

Una sola delle seguenti affermazioni che riguarda la collocazione degli oggetti in memoria in un programma C in un tipico ambiente Linux è **FALSA**, quale?

<b>A</b>	La variabile input risiede in .bss e la variabile max_len risiede in stack	<b>B</b>	La variabile *input risiede in heap e foo risiede in .text del processo in esecuzione
<b>C</b>	La variabile max_len è in stack mentre fgets è in .text del processo in esecuzione	<b>D</b>	La variabile input è in .bss e strlen è nella sezione che ospita le librerie[lib]

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

---

### Domanda 4 (Memoria Virtuale)

Quale di queste configurazioni in un indice della tabella delle pagine genera un page fault alla risoluzione di un puntatore corrispondente?

<b>A</b>	Il bit di validità è settato a 1 e la pagina è allocata nella memoria virtuale	<b>B</b>	Il bit di validità è settato a 1 e il frame è allocato nella memoria fisica
<b>C</b>	Il bit di validità è settato a 0 e la pagina si trova in swap	<b>D</b>	Il bit di validità è settato a 0 e la pagina non è allocata

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