

Esercizio: gestione dello Stack

Questo esercizio serve a comprendere come viene gestito lo **stack** di un programma durante le chiamate di funzione, in particolare il contenuto del **frame dello stack**, che include:

- **Parametri formali**
- **Variabili locali**
- **Indirizzo di ritorno**

Consegna: Disegna lo stack a mano o usa un programma per creare un diagramma che mostri chiaramente i diversi stack frame durante l'esecuzione del programma. Spiega brevemente i passaggi di crescita e riduzione dello stack a ogni chiamata e ritorno da una funzione.

Usa il tool [Python Tutor](#) che abbiamo visto in classe per simulare l'esecuzione del codice e aiutarti nello svolgimento dell'esercizio.

Descrizione

Considera il seguente programma C semplificato:

```
#include <stdio.h>
void funcB(int x, int y) {
    int z = x + y;
    printf("z = %d\n", z);
}

void funcA() {
    int a = 5;
    int b = 10;
    funcB(a, b);
}

int main() {
    funcA();
    return 0;
}
```

Passaggi da seguire

1. **Traccia lo stack durante l'esecuzione del programma:** Per ogni chiamata di funzione, disegna il frame dello stack e indica le seguenti informazioni:
 - **Parametri formali** (i parametri della funzione chiamata)
 - **Variabili locali** (dichiarate all'interno della funzione)
 - **Indirizzo di ritorno** (dove il programma tornerà una volta completata la funzione)
2. **Simulazione dello stack:**
 - Inizia con la chiamata alla funzione `main()`.
 - Poi, rappresenta la chiamata a `funcA()`.
 - Infine, rappresenta il frame dello stack quando viene chiamata `funcB()` con i parametri passati da `funcA()`.

Stack

Heap

main

funcA

a	int
	?

b	int
	?

Stack

Heap

main

funcA

a	int 5
b	int 10

funcB

x	int ?
y	int ?
z	int ?

Esempio di traccia:

a) Chiamata a main():

- Cosa viene inserito nello stack all'inizio del programma? Disegna lo stack.
Risposta i parametri formali argc e argv indirizzo di ritorno. Il main non ha variabili locali.

b) Chiamata a funcA() da main():

- Cosa accade nello stack quando funcA() viene chiamata?
Risposta: Viene creato il frame della funzione **funcA** e viene eseguita un'operazione di **push** sullo stack. Il frame contiene i parametri formali (nessuno), le variabili locali **a** e **b**, e l'indirizzo di ritorno.

- Aggiungi il frame di `funcA()` allo stack, con le sue variabili locali `a` e `b`.

Risposta Vedere immagine tool online

c) Chiamata a `funcB(int x, int y)` da `funcA()`:

- Mostra cosa viene inserito nello stack quando `funcB()` viene chiamata con i valori `a` e `b` passati da `funcA()`.
- Disegna il frame di `funcB()` che include:
- Parametri formali `x` e `y`
- La variabile locale `z`
- L'indirizzo di ritorno a `funcA()`

Risposta: vedere immagine tool online

3. Domanda aggiuntiva:

- Dopo il completamento di `funcB()`, cosa succede allo stack? Indica come i frame vengono rimossi e qual è il prossimo indirizzo di ritorno eseguito.

Risposta: Viene eseguita prima un'operazione di **pop** dalla cima dello stack per rimuovere **funcB**, seguita da un'altra operazione di **pop** per rimuovere **funcA**, e infine un **pop** per rimuovere il **main**.