Verifica di TPSI

La seguente verifica deve essere svolta entro la scadenza indicata dal docente. La verifica sarà valutata sulla base della correttezza delle formule e dei risultati, e sulla capacità di applicare correttamente i concetti visti in classe. L'intervallo dei voti è [2-10] calcolati su 10 punti; così organizzati: voto scritto calcolato su 9 punti + 1 punto elaborato ordinato

Per l'assegnazione dei punti saranno considerati i seguenti criteri: **Aderenze alla traccia** (la risposta è pertinente e rispetta la traccia), **Correttezza logica e completezza della soluzione/risposta** (la risposta è completa, precisa e correttamente motivata). Per l'assegnazione del punto relativo all'elaborato ordinato, saranno presi in considerazione i seguenti aspetti: il foglio della verifica e il foglio protocollo devono essere **correttamente intestati**, l'elaborato deve essere **leggibile** e **ben organizzato**.

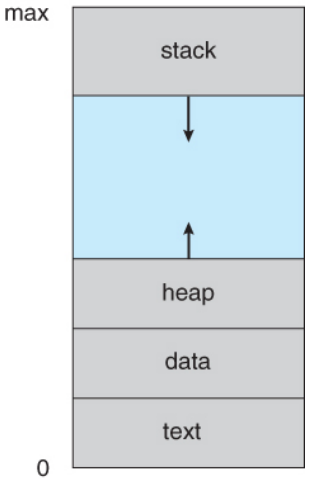
Rispondere a tutte le domande su foglio protocollo, riportando la domanda in modo ordinato.
Esempio: Domanda 1 - Risposta:

MISURE DISPENSANTIVE/COMPENSATIVE APPLICATE:

(3 pt) Domanda 1 - Struttura della memoria di un processo
Considerando la struttura dell'area di memoria di un processo come mostrata nell'immagine, spiega il ruolo delle varie sezioni di memoria in un processo. Descrivi le differenze tra le sezioni Text, Data, BSS, Stack e Heap. Perché le prime tre sono considerate statiche mentre le ultime due sono considerate dinamiche?

Considera il seguente codice in C:

```c
#include <stdio.h>
#include <stdlib.h>
int p = 7;
int a;
void calcola(int a, int b) {
    int calcolo = (a * b)+ a + b;
    printf("Calcolo: %d\n", calcolo);
    return calcolo;
}
int main() {
    int *n = (int*)malloc(sizeof(int));
    *n = 15;
    int calcolo = calcola(p, tempVar);
    free(n);
    return 0;
}
```



spiega a quale area appartengono le variabili nel codice d'esempio in C (Stack, Heap, Segmento Dati, Segmento BSS).

1. variabile p
2. variabile a
3. variabile calcolo all'interno della funzione calcola(int a, int b)
4. variabile calcolo all'interno della funzione main()
5. variabile n all'interno della funzione main()

(2 pt) Domanda 2 - Simulazione Stack Java
Dato il seguente codice Java:

```
class SocialProfile {
    String username;
    int followers;
    int postCount;
    public SocialProfile(String username, int followers, int postCount) {
        this.username = username;
        this.followers = followers;
        this.postCount = postCount;
    }
    public void addPosts(int newPosts) {
        this.postCount += newPosts;
    }
    public void updateFollowers(int newFollowers) {
        this.followers += newFollowers;
    }
    public void showProfileInfo() {
        System.out.println("Profile: @" + username + ", Followers: " + followers + ", Posts: " +
postCount);
    }
}
public class StackHeapSimulation {
    public static void main(String[] args) {
        String userName = "naif_cool_influencer";
        int initialFollowers = 1000;
        int initialPosts = 50;
        SocialProfile profile = new SocialProfile(userName, initialFollowers, initialPosts);
        // The user gains new followers
            int newFollowers = 300;
            profile.updateFollowers(newFollowers); // rappresentare stato stack/heap qui, prima del
ritorno.
                // rappresentare stato stack/heap qui, prima della chiamata showProfileInfo
            profile.showProfileInfo();
    }
}
```

rappresenta lo stato dello stack e dell'heap in due momenti

- quando viene chiamato il metodo `_updateFollowers_` nel main
- quando il metodo `_updateFollowers_` ha finito la sua esecuzione, subito prima che venga chiamato il metodo `_showProfileInfo_` nel main
  Specifica in modo preciso le informazioni contenute in uno stack frame di attivazione come visto a lezione.

(2 pt) Domanda 3 - Scheduling processi
Spiegare qual è la differenza tra un algoritmo Multiple Level Queue Scheduling e un algoritmo Multiple Level Feedback Queue Scheduling e spiegare la problematica che viene risolta.

(2 pt) Domanda 4 - Scheduling processi
Spiega perché è necessario che un Sistema Operativo mantenga un PCB per ogni processo attivato. Fai un esempio di come i PCB dei processi sono usati dal Sistema operativo.