BUFFER OVERFLOW

Nonostante io abbia provato ad eseguire il programma riportato nelle slide sulla VM di Kali, il compilatore mi correggeva in automatico l'errore di buffer overflow, consentendomi di inserire tutti i caratteri che volevo.
Non trovando una soluzione pratica al problema riporterò l'esecuzione dei programmi attraverso un compilatore online: **Programiz**.

Prima di tutto verifichiamo il programma riportato nelle slide:
lo trascriviamo sul nostro compilatore e lo eseguiamo e come ci si aspettava riporta **Segmentation fault**.
Cioè i caratteri inseriti sono maggiori rispetto a quelli consentiti consentiti dall'array (*char buffer[10]*.

```c
#include <stdio.h>

int main () {
    char buffer [10];
    printf("Inserire il nome utente:");
    scanf("%s", buffer);
    printf("Nome utente inserito: %s\n", buffer);

    return 0;
}
```

```
/tmp/qXw1mcG61k.o
Inserire il nome utente:bufferoverflowvediamosefunziona
Nome utente inserito: bufferoverflowvediamosefunziona
Segmentation fault
```

Per risolvere questo problema ho trovato 2 soluzioni possibili:

1. Aumentare la dimensione del buffer come mostrato nella prima figura qua a destra; in questo modo diamo più spazio alla memoria del buffer dove scrivere i dati inseriti.
2. Blocchiamo la possibilità di inserire più caratteri di quelli consentiti; definendo con *%9s*, in) il numero massimo di caratteri che verranno salvati in memoria, quello che eccede non verrà ne salvato ne quindi riportato a schermo.

```c
#include <stdio.h>

int main () {
    char buffer [30];
    printf("Inserire il nome utente:");
    scanf("%s", buffer);
    printf("Nome utente inserito: %s\n", buffer);

    return 0;
}
```

```
/tmp/qXw1mcG61k.o
Inserire il nome utente:bufferoverflowvediamosefunziona
Nome utente inserito: bufferoverflowvediamosefunziona
```

```c
#include <stdio.h>

int main() {
    char buffer[10];

    printf("Inserire il nome utente (massimo 9 caratteri): ");
    scanf("%9s", buffer);

    printf("Nome utente inserito: %s\n", buffer);

    return 0;
}
```

```
/tmp/qXw1mcG61k.o
Inserire il nome utente (massimo 9 caratteri):
    bufferoverflownonfunziona
Nome utente inserito: bufferove
```

Il terzo punto della consegna è stato più difficoltoso e sono riuscito a trovare una soluzioni penso parziale riuscendo ad indicare l'indirizzo di memoria dove vengono salvati i caratteri inseriti uno ad uno, sia quelli dentro che quelli fuori dall'array.
Immagine a destra del codice:
la prima parte del codice è uguale a quelle precedenti e permette di inserire una serie di caratteri a piacere, ma salva in memoria esclusivamente il numero consentito dall'array.
Dobbiamo come prima cosa aggiungere una nuova variabile: *int bufferL = 10*, che definisce il numero intero massimo a 10 dei caratteri.
Introduciamo 2 costrutti *for:*
1
Scorre tutta la lunghezza dell'array, definita dalla variabile bufferL con il valore di 10, e stampa a schermo per ogni valore=carattere, l'indirizzo di memoria dove viene salvato.

2
Il secondo for fa la medesima cosa del primo ma per tutti i valori che escono dalla lunghezza definita con bufferL, cioè tutti i caratteri dopo il decimo.
%d è un intero che fa riferimento a ***i*** cioè i valori definiti all'interno del ciclo for che formano l'array.
%p è un puntatore che fa riferimento a *(void*)&buffer[i]* cioè l'indirizzo di memoria del valore di i indicato.
Vorrei sottolineare che il *+ 100* è arbitrario: ho dovuto inserire una quantità di caratteri dopo l'array per i quali volevo conoscere l'indirizzo di memoria.

Output che verrà stampato a schermo è nella pagina successiva.

```c
#include <stdio.h>

int main() {
    int bufferL = 10;  // Lunghezza massima del buffer
    char buffer[10];

    printf("Indirizzo di memoria del buffer: %p\n", (void
        *)buffer);

    printf("Si prega di inserire il nome utente:");

    // Usa %9s per evitare buffer overflow
    if (scanf("%9s", buffer) != 1) {
        fprintf(stderr, "Errore di input.\n");
        return 1;
    }

    // Stampa gli indirizzi di memoria di ciascun carattere
        nel buffer
    for (int i = 0; i < bufferL; ++i) {
        printf("Indirizzo di memoria di buffer[%d]: %p\n", i,
            (void*)&buffer[i]);
    }

    // Stampa gli indirizzi di memoria al di fuori del buffer
    for (int i = bufferL; i < bufferL + 100; ++i) {
        printf("Indirizzo di memoria al di fuori del
            buffer[%d]: %p\n", i - bufferL, (void*)&buffer[i]
            );
    }

    return 0;
}
```

```
Indirizzo di memoria del buffer: 0x7ffc46af7c4a
Si prega di inserire il nome utente:bufferoverflowperchènonfunzioni
Indirizzo di memoria di buffer[0]: 0x7ffc46af7c4a
Indirizzo di memoria di buffer[1]: 0x7ffc46af7c4b
Indirizzo di memoria di buffer[2]: 0x7ffc46af7c4c
Indirizzo di memoria di buffer[3]: 0x7ffc46af7c4d
Indirizzo di memoria di buffer[4]: 0x7ffc46af7c4e
Indirizzo di memoria di buffer[5]: 0x7ffc46af7c4f
Indirizzo di memoria di buffer[6]: 0x7ffc46af7c50
Indirizzo di memoria di buffer[7]: 0x7ffc46af7c51
Indirizzo di memoria di buffer[8]: 0x7ffc46af7c52
Indirizzo di memoria di buffer[9]: 0x7ffc46af7c53
Indirizzo di memoria al di fuori del buffer[0]: 0x7ffc46af7c54
Indirizzo di memoria al di fuori del buffer[1]: 0x7ffc46af7c55
Indirizzo di memoria al di fuori del buffer[2]: 0x7ffc46af7c56
Indirizzo di memoria al di fuori del buffer[3]: 0x7ffc46af7c57
Indirizzo di memoria al di fuori del buffer[4]: 0x7ffc46af7c58
Indirizzo di memoria al di fuori del buffer[5]: 0x7ffc46af7c59
Indirizzo di memoria al di fuori del buffer[6]: 0x7ffc46af7c5a
Indirizzo di memoria al di fuori del buffer[7]: 0x7ffc46af7c5b
Indirizzo di memoria al di fuori del buffer[8]: 0x7ffc46af7c5c
Indirizzo di memoria al di fuori del buffer[9]: 0x7ffc46af7c5d
Indirizzo di memoria al di fuori del buffer[10]: 0x7ffc46af7c5e
Indirizzo di memoria al di fuori del buffer[11]: 0x7ffc46af7c5f
Indirizzo di memoria al di fuori del buffer[12]: 0x7ffc46af7c60
Indirizzo di memoria al di fuori del buffer[13]: 0x7ffc46af7c61
Indirizzo di memoria al di fuori del buffer[14]: 0x7ffc46af7c62
```