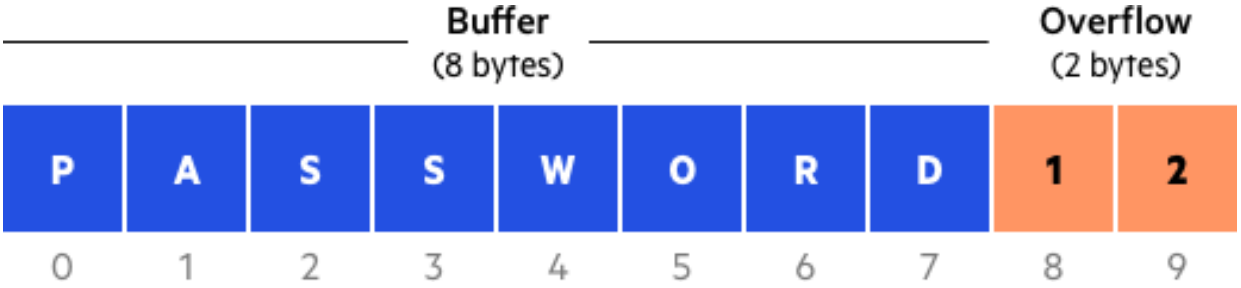
[1] Traccia:

L'esercizio di oggi, ci chiede di effettuare un BoF (Buffer Overflow) su un programma compilato in C con un array di interi avente dimensione 10.

[2] Definizione di buffer overflow:

Il buffer overflow (trabbocamento della memoria), è una condizione che si verifica quando in un'area di memoria definita, si inseriscono più dati di quanti previsti.



[3] Funzionamento del programma:

Il programma riportato di seguito, non fa altro che prendere 10 numeri interi in input, confrontarli grazie ad algoritmi di ordinamento realizzati con una serie di cicli for e una variabile di appoggio, e infine stampare il vettore con i numeri ordinati.

```c
#include <stdio.h>

int main () {

int vector [10], i, j, k;
int swap_var;

printf ("Inserire 10 interi:\n");

for (i = 0; i < 10; i++)
    {
    int c= i+1;
    printf("[%d]:", c);
    scanf ("%d", &vector[i]);
    }

printf ("Il vettore inserito e':\n");
for (i = 0; i < 10; i++)
        {
        int t= i+1;
        printf("[%d]: %d", t, vector[i]);
    printf("\n");
    }

for (j = 0; j < 10 - 1; j++)
    {
    for (k = 0; k < 10 - j - 1; k++)
        {
            if (vector[k] > vector[k+1])
            {
            swap_var=vector[k];
            vector[k]=vector[k+1];
            vector[k+1]=swap_var;
            }
        }
```

```
    }

printf("Il vettore ordinato e':\n");
for (j = 0; j < 10; j++)
    {
    int g = j+1;
    printf("[%d]:", g);
    printf("%d\n", vector[j]);
    }

return 0;

}
```

[3] Modifica del codice:

In questo codice non sono presenti errori che possano far verificare il segmentation fault. Per far si che si verifichi un buffer overflow, andiamo a modificare dove l'utente inserisce i valori in input.

PRIMA:

```
printf ("Inserire 10 interi:\n");

for (i = 0; i < 10; i++)
    {
    int c= i+1;
    printf("[%d]:", c);
    scanf ("%d", &vector[i]);
    }
```

DOPO:

```
printf ("Inserire 10 interi:\n");

for (i = 0; i < 11; i++)
    {
    int c= i+1;
    printf("[%d]:", c);
    scanf ("%d", &vector[i]);
    }
```

[4] Compilazione ed esecuzione del codice:

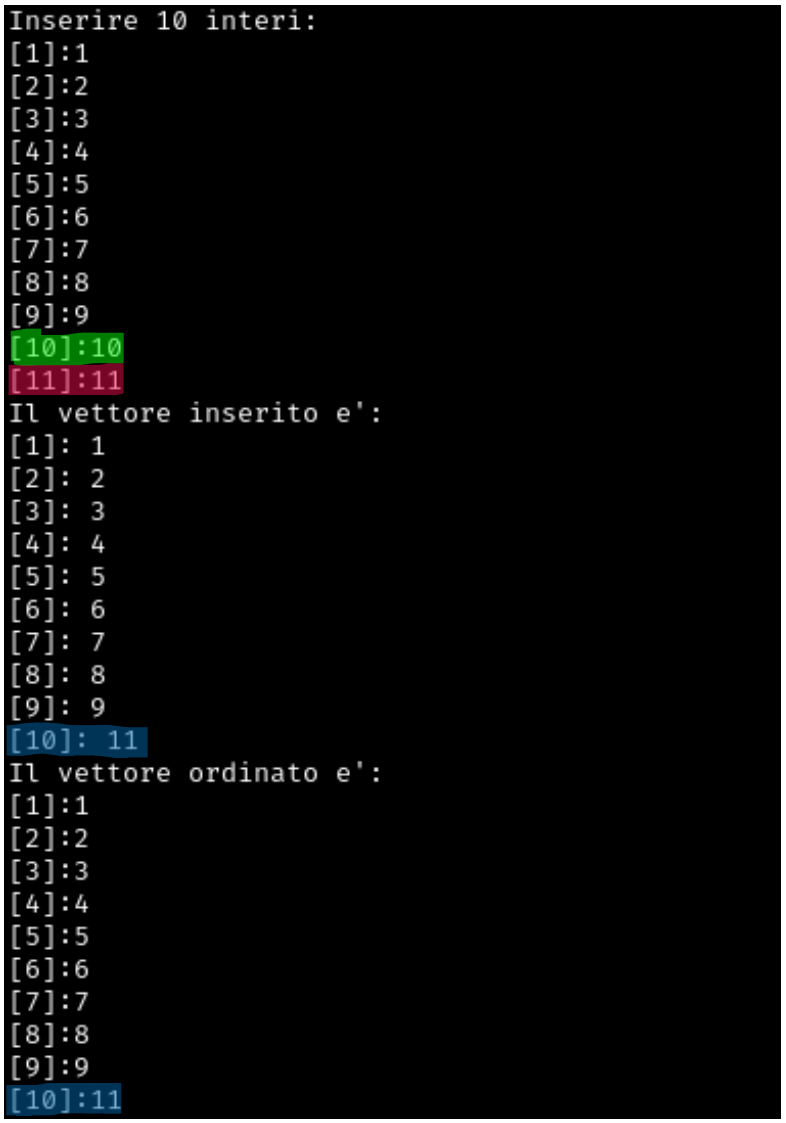
Come prima cosa modifichiamo il programma in C utilizzando l'editor di testo nano.
Una volta scritto il programma lo compiliamo con il comando gcc (GNU Compiler collection) e lo eseguiamo con ./<nome del programma> .

```
C:\home\kali\Desktop\esC> gcc -g BW_D3_BOF.c -o BoF

C:\home\kali\Desktop\esC> ./BoF
```

[5] Conclusioni:

Come notiamo, inserendo 11 caratteri, si verifica l'errore di segmentetion fault che non causa il crash del programma, ma ne potrebbe modificare il comportamento. Sotto in figura viene mostrato dove viene influenzato il codice:



Il valore inserito nella posizione 10 dell'array non corrisponde a quello stampato in figura, ma corrisponde al valore inserito nella posizione 11 che nell'array non è definita.