

TRACCIA

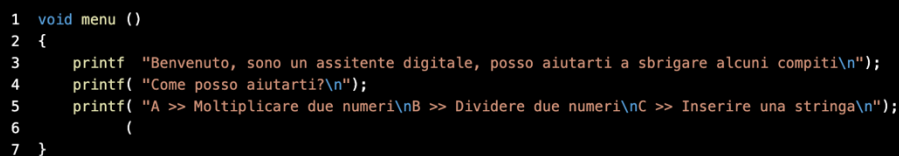
Per agire come un Hacker bisogna capire come pensare fuori dagli schemi. L'esercizio di oggi ha lo scopo di allenare l'osservazione critica. Dato il codice in allegato, si richiede allo studente di:

- Capire cosa fa il programma senza eseguirlo (1)
- Individuare dal codice sorgente le casistiche non standard che il programma non gestisce (esempio, comportamenti potenziali che non sono stati contemplati) (2)
- Individuare eventuali errori di sintassi / logici (3)
- Proporre una soluzione per ognuno di essi (4)

(1) :

Il programma è un semplice assistente digitale scritto in C. Il suo scopo è quello di accogliere l'utente e offrire alcune opzioni per compiti che può svolgere. Ecco una spiegazione delle tre opzioni fornite:

1. **Moltiplicare due numeri (Opzione A):** Se l'utente seleziona l'opzione A, il programma chiederà all'utente di inserire due numeri e quindi eseguirà l'operazione di moltiplicazione, mostrando il risultato dell'operazione in output.
2. **Dividere due numeri (Opzione B):** Se l'utente seleziona l'opzione B, il programma chiederà all'utente di inserire due numeri (numeratore e denominatore) e quindi eseguirà l'operazione di divisione, mostrando il risultato dell'operazione in output.
3. **Inserire una stringa (Opzione C):** Se l'utente seleziona l'opzione C, il programma chiederà all'utente di inserire una stringa.



```
1 void menu ()
2 {
3     printf "Benvenuto, sono un assistente digitale, posso aiutarti a sbrigare alcuni compiti\n";
4     printf( "Come posso aiutarti?\n");
5     printf( "A >> Moltiplicare due numeri\nB >> Dividere due numeri\nC >> Inserire una stringa\n");
6     (
7 }
```

(2-3-4):

1.

Tipo di dato errato per la variabile "scelta":

- La variabile **scelta** viene inizializzata con un valore nullo, ma l'uso delle parentesi graffe ({} in questo caso non è necessario, in quanto viene utilizzato quando si vuole dichiarare un array.

2. **Input non corretto per la variabile "scelta":**

linea 14 : scanf ("%d", &scelta);

- Nell'istruzione **scanf**, dovremmo usare **%c** per leggere un carattere singolo. Lo spazio prima di **%c**, è importante per ignorare eventuali caratteri di spaziatura residui (ad esempio, newline) nell'input.

scanf(" %c", &scelta);

3. **"moltiplica()":**

- short int a,b = 0; => short int a,b;
 - Posso evitare l'inizializzazione a 0;
- scanf("%f", &a); => scanf("%d",&a);
 - Per ottenere un risultato intero cambio %f con %d poiché %f legge numeri reali

```
1 void moltiplica ()
2 {
3     short int a,b;
4     printf "Inserisci i due numeri da moltiplicare: ";
5     scanf ("%d", &a);
6     scanf ("%d", &b);
7     (
8     short int prodotto = a * b;
9
10    printf "Il prodotto tra %d e %d e': %d", a,b,prodotto);
11 }
12
```

4. Errore di formato in "scanf" in "ins_string()":

- Nella funzione **ins_string()**, bisogna usare **%9s** nel posto di **%s** per evitare un potenziale overflow del buffer. Inoltre, non è necessario l'operatore di indirizzo (**&**).

`scanf("%9s", stringa);`

```
1
2 void ins_string ()
3 {
4     char stringa[10];
5     printf "Inserisci la stringa: ";
6     scanf("%9s", stringa);
7 }
```

5. Errore nella funzione "dividi()":

- Il calcolo dell'operazione di divisione (**divisione = a % b;**) è un resto e non una divisione. Dobbiamo usare **/** per eseguire la divisione.

`int divisione = a / b;`

Inoltre bisogna validare che b non sia 0 in quanto la divisione con 0 non è possibile.

```
1
2 void dividi ()
3 {
4     int a,b;
5     printf "Inserisci il numeratore: ";
6     scanf ("%d", &a);
7     printf "Inserisci il denominatore: ";
8     scanf ("%d", &b);
9     (
10    if(b!=0){
11        int divisione = a % b;
12        printf "La divisione tra %d e %d e': %d", a,b,divisione);
13    }else{ (
14        printf "Non è possibile dividere per 0");
15    } (
16    }
17 }
```