BUG Hunting S2/L5

Manuel Di Gangi 2 febraio 2024

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.
- Individuare dal codice sorgente le casistiche non standard che il programma non gestisce (esempio, comportamenti potenziali che non sono stati contemplati).
- Individuare eventuali errori di sintassi / logici.
- Proporre una soluzione per ognuno di essi.

Funzionalità

Il programma in questione permette all'utente di effettuare: moltiplicazioni, divisioni e l'inserimento di una stinga, selezionando l'operazione mediante un menu.

Ogni operazione ha una funzione dedicata, compreso il menu.

Il codice

```c
#include <stdio.h>

//Prototipi delle funzioni
void menu ();
void moltiplica ();
void dividi ();
void ins_string();


int main ()
{
    char scelta = {'\0'}; //mettendo le {} lo riconosce come array
```

- Errore di sintassi: per inizializzare un char vuoto la forma corretta è char scelta= ' '. Utilizzando { } viene interpretato come array di char, potrebbe compilare correttamente, ma creare problemi successivamente dal momento che non viene gestito come tale. In C \0 indica la fine di una stringa o una stringa vuota.

```c
    menu ();
    scanf ("%d", &scelta); //sintax err. scelta è una var char quindi %c
```

- Errore di sintassi: la variabile scelta è un char per cui l'identificatore corretto da inserire nello scanf è %c.

```c
    switch (scelta)
    {
        case 'A': // - non gestisce la casistica 'a'
```

```
        moltiplica();
        break;
        case 'B':// - non gestisce la casistica 'b'
                dividi();
                break;
       case 'C':// - non gestisce la casistica 'c'
                ins_string();
                break;
     // - non gestisce caso default
    }
```

- Casistica non gestita: qualora l'utente inserisca a, b, c il programma non procede con l'esecuzione. E' corretto aggiungere altri case con i caratteri minuscoli.
- Casistica non gestita: qualora l'utente inserisca un valore che non rientra nelle opzioni il programma termina. E' più indicato prevedere una situazione di default che comunica all'utente che l'input non è valido e gli permetta di ripetere l'inserimento.

```
return 0;
}

void menu ()
{
    printf ("Benvenuto, sono un assitente digitale, posso aiutarti a
sbrigare alcuni compiti\n"); //assistente manca una s
    printf ("Come posso aiutarti?\n");
    printf ("A >> Moltiplicare due numeri\nB >> Dividere due numeri\nC
>> Inserire una stringa\n");
}

void moltiplica ()
{
    short int  a,b = 0;
    printf ("Inserisci i due numeri da moltiplicare:");
    scanf ("%f", &a); //non %f ma %hd
    scanf ("%d", &b); //non %d ma %hd
```

- Errore di sintassi: le variabili 'a' e 'b' sono short int per cui l'identificatore corretto da inserire nello scanf è %hd.

```
    short int prodotto = a * b;
    printf ("Il prodotto tra %d e %d e': %d", a,b,prodotto); //%hd
}
```

- Errore di sintassi: le variabili 'a' e 'b' sono short int per cui l'identificatore corretto da inserire nello scanf è %hd.

```
void dividi ()
{
        int  a,b = 0;
        printf ("Inserisci il numeratore:");
        scanf ("%d", &a);
    printf ("Inserisci il denumeratore:"); //denominatore
        scanf ("%d", &b); // != da 0
```

- Casistica non gestita: qualora l'utente inserisca 0 il programma restituirebbe un errore. E' indicato inserire una condizione che non permetta di dividere per 0 facendo inserire all'utente un altro numero.

```
        int divisione = a % b; // / non %
```

- Errore di logica: l'operatore % restituisce il resto della divisione, per la divisione si utilizza l'operatore /.
- Casistica non gestita: dal momento che il risultato della divisione spesso è un numero decimale, potrebbe sarebbe utile utilizzare una variabile di tipo, cambiando di conseguenza anche il relativo identificatore nell'istruzione seguente.

```
        printf ("La divisione tra %d e %d e': %d", a,b,divisione);
}

void ins_string ()
{
   char stringa[10];
        printf ("Inserisci la stringa:");
        scanf ("%s", &stringa);
```

- Errore di sintassi: in C le stringhe sono elaborate come array di char, di conseguenza la variabile è già un puntatore, per questo non va messa la & davanti al nome della variabile.
- Casistica non gestita: qualora l'utente inserisca più di 10 caratteri il programma andrebbe in stack overflow. Per agirare questo problema sarebbe ideale leggere solo i primi 10 caratteri che l'utente digita specificando %10s.

```
}
```