S2/L5

Traccia:

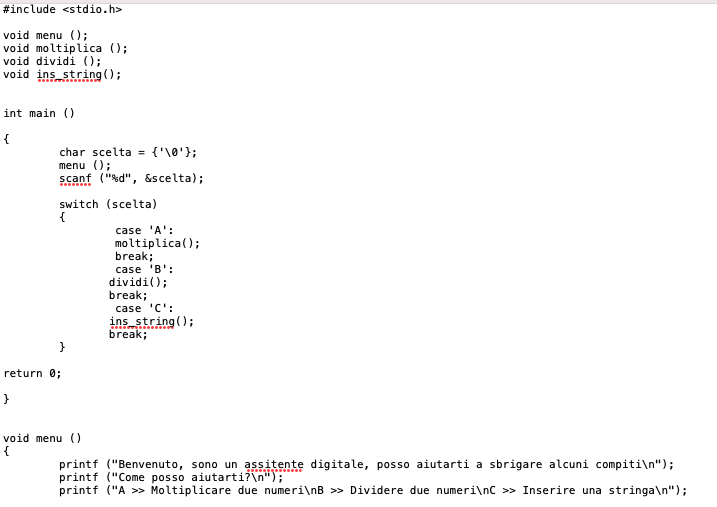
Per agire come un Hacker bisogna capire come pensare fuori dagli schemi. L’esercizio di oggi ha lo scopo di allenare l’osservazione critica. Dato il codice in allegato, si richiede allo studente di:

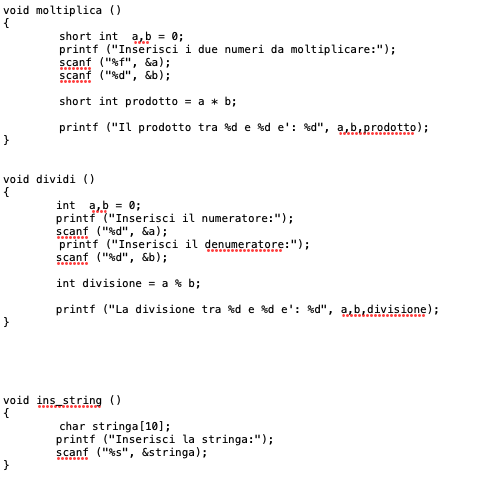
-Capire cosa fa il programma senza eseguirlo

-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





La prima cosa che uscirà avviando questo codice sarà: “Benvenuto sono un assistente digitale, posso aiutarti a sbrigare alcuni compiti. Come posso aiutarti?

A >>> moltiplicare due numeri

B >>> dividere due numeri

C >>> inserire una stringa

Per cui andando a cliccare i rispettivi tasti della tastiera (A, B o C) faremo partire un diverso blocco di codice. Nel codice infatti, abbiamo l’istruzione condizionale switch, detto anche di “selezione multipla,” ovvero scrivendo switch e il nome della variabile tra parentesi (in questo caso è la variabile “scelta” di tipo char che identifica un carattere di 1 byte) troveremo tre diverse casistiche.

“Char” in questo caso è un array di caratteri come stringa, e per gestirla come tale non è necessario conoscere a priori la sua lunghezza ma è importante inserire {‘\0’} come terminatore.

Un errore logico in questo blocco di codice si trova nell’istruzione scanf, in quanto il tipo di valore “char scelta” non è %d; bensì:

scanf (“%c”, &scelta);

Si aprono varie ipotesi all’interno dello switch, se il suo valore “case” è:

“A” allora verranno eseguite le istruzioni definite in “moltiplica”,

“B” verrà eseguito il blocco di codice identificato in “dividi”,

“C” verrà eseguito il blocco ins\_string.

All’interno di questo blocco di codice, aggiungiamo l’istruzione:

“default: printf (“selezione non valida”); ovvero viene impostata una casistica di default che non era stata programmata. L’istruzione default viene eseguita se nessun “case” A B o C viene eseguito.

Nella funzione “main” abbiamo infine il valore di ritorno indicato con “return 0” ovvero quando si completa con successo.

Di seguito troviamo il codice per la variabile void menu (), in cui è presente un errore di sintassi: assitente anziché assistente.

Troviamo poi il codice per le tre funzioni (moltiplica, dividi, ins\_string):

In “moltiplica” vi è un errore di logica:

short int a,b = 0;

printf ("Inserisci i due numeri da moltiplicare:");

scanf ("%f", &a);

scanf ("%d", &b);

short int prodotto = a \* b;

printf ("Il prodotto tra %d e %d e': %d", a,b,prodotto);

Scriviamo come tipo di variabile a,b INT per entrambe e non SHORT (per entrambi scegliamo INT ovvero un numero intero di 4 byte).

Per cui di conseguenza, in scanf, il tipo di argomento da inserire in input è (“%d”, &a) e non “%f”

Inoltre, non inizializziamo le due variabili “a” e “b” = 0

Per cui il blocco di codice per la funzione moltiplica sarà:

void moltiplica () {

int a, b;

printf (“inserisci i due numeri da moltiplicare:”);

scanf (“%d %d , &a, &b);

int prodotto = a \* b;

printf ("Il prodotto tra %d e %d e': %d", a, b, prodotto);

Nella funzione dividi () troviamo un primo errore di sintassi, ovvero:

denumeratore anziché denominatore;

ed un errore di logica:

int divisione = a % b; questo simbolo % non indica la divisione, bensì il modulo ovvero divide i due numeri ma restituisce come risultato il resto. Per cui sarà:

int divisione = a / b;

Inoltre, anche in questo caso eliminiamo la dichiarazione di “a” e “b” = 0; ed aggiungiamo una casistica non contemplata in fase di programmazione, ovvero:

se dividiamo un numero per 0; cosa succede?

Possiamo scrivere una funzione di If – else in cui scriviamo che se il denominatore, in questo caso “b” è diverso da zero, allora si procederà con la divisione, altrimenti se è zero uscirà come output “errore”.

Per cui la il codice della funzione if – else sarà:

if (b != 0) {

int divisione = a / b;

printf (“la divisione tra %d e %d e’ : %d, a, b, divisione);

} else {

Printf (“errore”); }

Nella funzione ins\_string a differenza delle prime due funzioni con numeri interi, abbiamo una funzione di caratteri. Lo vediamo dall’array di Char stringa[10] e dall’istruzione scanf (“%s”, &stringa); ciò significa che potrò inserire un massimo di 10 caratteri, altrimenti darà errore.

Possiamo aggiungere che, la libreria standard di C dispone di funzioni molto utili per la gestione delle stringhe. Per poter utilizzare tali funzioni bisogna includere nel proprio file sorgente il file di libreria: #include <string.h>

