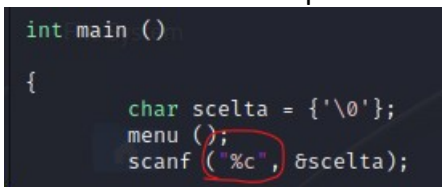
Il programma è un assistente digitale che dà la possibilità all'utente di eseguire operazioni semplici come la moltiplicazione e la divisione e la creazione di una stringa.
Presenta un menù di scelta iniziale in cui l'utente può decidere quale funzione usare in base a tre possibilità presentate tramite un ciclo switch: A per la moltiplicazione, B per la divisione, C per l'inserimento di una stringa di massimo 10 byte.

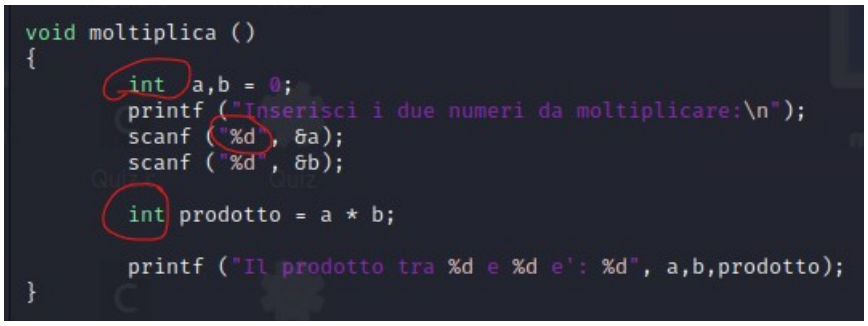
Errori di sintassi e logici.

Alla riga 14 la sintassi della variabile indica un numero intero (%d), ma il tipo di variabile indica un carattere (**char**); la sintassi corretta dovrebbe quindi essere **%c**.



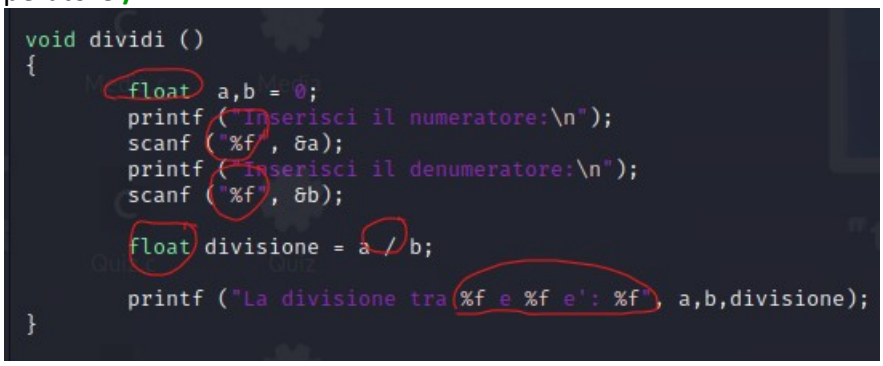
Alla riga 45 sono indicati due tipi di funzione, **short** e **int** (errore logico); in questo caso abbiamo bisogno di un solo tipo, possiamo scegliere short se si necessitano numeri interi di piccole dimensioni (2 byte), int per numeri interi grandi (4 byte). Scegliamo **int**. Facciamo lo stesso alla riga 50.

Alla riga 47 abbiamo abbiamo un'altro errore di sintassi: la variabile int prevede la sintassi **%d**.



Alle righe 58 e 64 la funzione divisione dovrebbe avere tipo di variabile **float**, non int, in quanto il risultato della divisione potrebbe essere un numero decimale, quindi non intero (errore logico). Andiamo a cambiare anche la sintassi del tipo di variabile alle righe 60, 62 e 66, scrivendo %f.

Inoltre, nella stessa riga si usa l'operatore aritmetico % per indicare la divisione, ma questo è un errore di sintassi; % restituisce il modulo (cioè il resto di una divisione), mentre il quoziente è restituito dall'operatore **/**.



*Non si tratta di un errore di sintassi, ma potremmo aggiunger il comando **\n** per una nuova riga alla fine delle funzioni printf, per rendere più chiara la visualizzazione a schermo.

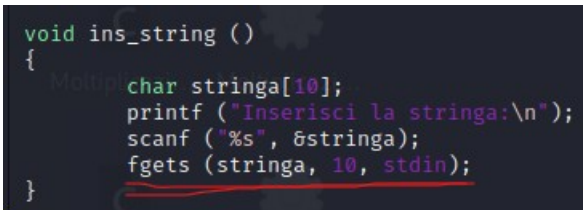
Casi non contemplati.

Se nel menù iniziale dovessi inserire una lettera diversa da A, B o C o un numero, il programma terminerebbe automaticamente. Si potrebbe rimediare tramite un ciclo **do-while**, che mi faccia tornare al menù se inserisco una opzione incorretta.

```
do {
menu ();
scanf ("%c", &scelta);

switch (scelta)
        {
        case 'A':
        moltiplica();
        break;
        case 'B':
        dividi();
        break;
        case 'C':
        ins_string();
        break;
        }
}
while ((scelta ≠ 'A') || (scelta ≠ 'B') || (scelta ≠ 'C'));
```

Nella funzione “ins_string” il programma non termina sia che inseriamo una stringa entro i limiti richiesti sia che superiamo detti limiti. Potremmo usare la funzione fgets per limitare il numero di caratteri.



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