A cosa serve il programma?

il programma è progettato per essere un assistente che può eseguire tre diverse operazioni in base alla scelta dell'utente: moltiplicare due numeri, dividere due numeri o inserire una stringa.

Quali sono le casistiche non gestite?

- Il programma assume che l'utente inserisca dati validi, ma non gestisce situazioni in cui l'input non è valido. (Ad esempio, se l'utente inserisce una lettera o un carattere speciale invece di un numero, ciò potrebbe causare comportamenti imprevisti.)
- Le variabili **a** e **b** nella funzione moltiplica sono dichiarate come **short int**, ma vengono lette come **float** e **int** rispettivamente, questo può causare un overflow.
- Non c'è gestione degli errori per la divisione per zero nella funzione dividi.
- L'input della stringa **(ins_string)** potrebbe comportarsi in modo imprevedibile se la stringa contiene spazi.

Individua errori di sintassi/logici

- L'assegnazione di un array di caratteri a **scelta (char scelta = {'\0'};)** non è corretta. **scelta** dovrebbe essere dichiarato come **char scelta = '\0';**
- L'uso di **%f** per leggere un intero in **moltiplica** è errato. Deve essere **%d**.
- L'uso di **%s** per leggere una stringa in **ins_string** può portare a un comportamento imprevedibile se la stringa contiene spazi.

Soluzioni

Risolvo il primo errore di sintassi sopra scritto dunque.

```
]{
    char scelta = '\0';
    menu ();
    scanf ("%d", &scelta);
```

Utilizzo **%c** al posto di **%d** per leggere un singolo carattere e aggiungere un carattere newline dopo la scanf per consumare eventuali caratteri rimanenti nell'input buffer.

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

    switch (scelta)
```

Vado a correggere qualche errore nelle variabili **void moltiplica()**. (spiego nello screen)

```
void moltiplica ()
{
    short int a,b; //rimuovo =0
    printf ("Inserisci i due numeri da moltiplicare:");
    scanf ("%d", &a); //%f legge un numero reale ma a noi serve intero
    scanf ("%d", &b);
```

Aggiungo una condizione per verificare se il denominatore è zero prima di eseguire la divisione.

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


        if (b != 0) {
    int divisione = a / b;
    printf("La divisione tra %d e %d è: %d", a, b, divisione);
} else {
    printf("Impossibile dividere per zero.\n");
}

}
```

```
Come posso aiutarti?
A >> Moltiplicare due numeri
B >> Dividere due numeri
C >> Inserire una stringa
B
Inserisci il numeratore:20
Inserisci il denumeratore:4
La divisione tra 20 e 4 e': 5
```

```
Come posso aiutarti?
A >> Moltiplicare due numeri
B >> Dividere due numeri
C >> Inserire una stringa
B
Inserisci il numeratore:20
Inserisci il denumeratore:0
Impossibile dividere per zero.
```

Modifico l'ultima cosa nel programma ovvero nelle variabili **void ins_string ().** Utilizzo **%9s** per evitare overflow del buffer e consentire la lettura di massimo 9 caratteri. Rimuovero l'operatore **&** poiché l'array di caratteri è già un puntatore.

```
void ins_string ()
{
      char stringa[10];
           printf ("Inserisci la stringa
           scanf("%9s", stringa);

}
```