PROGETTO S2/L5 MARTA ARAMU

Consegna: 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

1. Questo programma è un assistente digitale avente un menù con tre opzioni (moltiplica, dividi, inserisci stringa) e svolge le operazioni basandosi sulla scelta fornita in input dall'utente.

2. Ci sono alcune casistiche che potrebbero causare problemi:

- **Input non valido nel menu**: se l'utente inserisce un carattere diverso da 'A', 'B' o 'C', il programma non gestisce questa situazione.
- **Divisione per zero**: nel caso in cui l'utente scelga di usare l'operazione di divisione e inserisca 0 come denominatore, il programma non gestisce la divisione per zero.
- **Input non numerici**: se l'utente sceglie di eseguire un'operazione matematica ma inserisce dei valori non numerici, il programma potrebbe comportarsi in modo imprevedibile.
- **Overflow nella moltiplicazione**: se l'utente inserisce numeri molto grandi per la moltiplicazione, potrebbe verificarsi un overflow se il risultato supera il limite massimo rappresentabile.
- **Stringhe troppo lunghe**: nella funzione ins_string, se l'utente inserisce una stringa più lunga di 10 caratteri, potrebbe verificarsi un overflow del buffer.
- **Conversioni errate tra tipi di dati**: nella funzione di moltiplicazione, i numeri vengono inseriti come float, ma poi vengono letti come int e ciò potrebbe causare problemi nelle conversioni.
- **Uso di scanf per stringhe**: l'uso di scanf("%s", stringa) per leggere una stringa potrebbe portare a problemi se l'utente inserisce spazi o caratteri speciali.

3-4. Ecco alcune correzioni che apporterei al codice:

- nella funzione main modificherei il contenuto di scanf perché nel codice originale è stato messo un %d, ma sarebbe invece corretto un %c perché %d è utilizzato per la conversione di dati di tipo intero mentre %c è utilizzato per la conversione di dati di tipo carattere, e "scelta" è di tipo char.
- sempre nel main nell'operatore switch non è contemplato il caso in cui l'utente inserisca un'opzione non valida, perciò inserirei un caso di default per gestire le situazioni in cui l'utente inserisca un'opzione non valida, quindi così: printf("Scelta non valida.\n");
- nella funzione moltiplica cambierei il tipo di dato e metterei per entrambi gli operandi il tipo float e in scanf cambierei da %d a %f, in modo da evitare problemi di conversione.
- anche nella funzione dividi cambierei il tipo di dato a float in caso di divisione col resto.

- nella funzione ins_string penso che i caratteri inseriti inizialmente siano troppo pochi, quindi aumenterei il numero di char (ad esempio da 10 a 99) per gestire stringhe più lunghe ed evitare così un overflow del buffer.
- sempre nella funzione ins_string cambierei la lettura della stringa da scanf("%s", stringa) a scanf("%99s", stringa) sempre per evitare overflow del buffer. Tuttavia per un miglior controllo sarebbe meglio usare la funzione fgets, che viene utilizzata per leggere una riga di testo da un file o da uno stream di input, come ad esempio la tastiera. Se la lunghezza di una stringa supera la dimensione dell'array, fgets leggerà (nel nostro esempio) i primi 98 caratteri (poiché uno è riservato per il carattere terminatore \0) o tutti i caratteri disponibili fino al carattere di nuova riga (\n) o fino alla fine del file, a seconda di quale si verifica prima.