```
#include <stdio.h>

void menu ();
void moltiplica ();
void dividi ();
void ins_string();


int main ()

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

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

return 0;

}


void menu ()
{
	printf ("Benvenuto, sono un assitente digitale, posso aiutarti a sbrigare alcuni compiti\n");
	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);
	scanf ("%d", &b);

	short int prodotto = a * b;
```

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


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

      int divisione = a % b;

      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);
}
```

Gli errori di sintassi sono :

1- Dichiarazione di 'scelta':
Il tipo di dato di 'scelta' è di dichiarazione come 'char' ma nel 'switch' si stanno confrontando caratteri con interi. si dovrebbe dichiarare 'scelta come 'char'(es.) char scelta = '/0' -->ulizza'/0' per inizializzare la variabile char

2-errore nella dichiarazione di 'scelta':

La assegnazione di caratteri dovrebbe essere fatta con singoli apici, non con le parantesi graffe
//sbagliato
char scelta = {'\0'};
//giusto
char scelta = '\0';

3-utilizzo errato di '%f' nel metodo 'moltiplica':

I njmeri 'a' e 'b' sono dichiarati come 'short int', ma stai usando '%f' per la scansione. Dovresti usare '%hd' per leggere corretamente i valori ----> es scanf("%hd", &a);

4- dichiarazione di 'prodotto' nel metodo 'motiplica':

La variabile 'prodotto' è dichiarata come 'short int' , ma il risultato della moltiplicazione può superare il limite di un ' short int'. è
consigliabile usare 'int' o 'long' a seconda delle necessità

-->int prodotto = a*b;

5- Errore nella dichiarazione di variabili in ' dividi':
La variabile 'dividere' dovrebbe essere dichiarata come 'float' o 'double' se vuoi ottenere il risultato della divisione con i decimali .inoltre, '%' restituisce il resto, non il risultato della divisione --->float divisione = (float)a/b;

6-Dichiarazione di 'stringa' in 'ins_string':
Il tuo array si stringhe dovrebbe avere una dimenione sufficiente grande per contenere la stringa, considerando anche il carattere terminatore"\0".inoltre, non è necessario utilizzare l'operatore di indirizzo '&'con '%s'.--->char stringa[20]; // dimensione sufficientemente grande
scanf("%s", stringa); // senza l'operatore di indirizzo &

Errori Logici:
1-Scansione di carattere in 'main':
nella scansione della scelta stai usando '%d' che è adatto per interi, ma 'scelta' è di tipo ' char' .Utilizza '%c' per la scnasione dei caratteri.-->scanf(" %c", &scelta); // aggiungi uno spazio prima di %c per ignorare eventuali spazi e nuove linee precedenti

2-Operazione '%' nella divisione :
Nella funzione 'dividi', stai utilizando l'operatore '5' che restituisce il resto della divisione . Se vuoi ottenere il risultato della divisione , usa '/'

Individuare dal codice sorgente le casistiche non standard che il programma non gestisce (esempio, comportamenti potenziali che non sono stati contemplati).

Nel codice sorgente fornito, ci sono alcune casistiche con comportamenti potenziali non contemplati che posso causare errori durante l'esecuzione del programma.

1- Gestione dell'input non valido nel menu:
Il programma assume che l'utente inserirà solo i caratteri 'A','B', o 'C' come scelta nel menu. Se l'utente inserisce un input diverso, il comportamento è imprevedibile. puoi gestire quetso caso nel codice, ad esempio ,ignorando l'input non valido.
---->default:
    printf("Scelta non valida. Riprova.\n");
    break;

2-Gestione dell'input nel metodo 'moltiplica':
Nel metodo 'moltiplica', il programma assume che l'utente inserirà due numeri validi come input. Se l'utente inserisce una stringa o un carattere invece di un numero, il comportamento sarà imprevedibile. È necessario gestire questo caso con controlli di input appropriati.

3-Gestione dell'input nel metodo 'dividi':
Simile al metodo moltiplica, il metodo dividi assume input numerici validi. È necessario aggiungere controlli per gestire input non validi.

4-Gestione della stringa nel metodo 'ins_string':
La funzione 'ins_string' dichiara un array di caratteri' stringa[10]', ma non gestisce il caso in cui l'utente inserisca una stringa più lunga di 10 caratteri, potenzialmente causando un overflow del buffer. È consigliabile utilizzare '%9s' nella scanf per evitare questo problema.
-->scanf("%9s", stringa);

Capire cosa fa il programma senza eseguirlo.

Menu e Selezione:
Il programma inizia mostrando un menu che offre tre opzioni:

Moltiplicare due numeri
Dividere due numeri
Inserire una stringa
L'utente può selezionare un'opzione inserendo il carattere corrispondente.

Switch e Chiamate di Funzione:
Il programma utilizza un costrutto 'switch' per gestire la selezione dell'utente. Se l'utente inserisce 'A', chiamerà la funzione 'moltiplica'(), se inserisce 'B', chiamerà la funzione 'dividi()', e se inserisce 'C', chiamerà la funzione 'ins_string()'.

Funzione Moltiplica:
La funzione 'moltiplica()' richiede all'utente di inserire due numeri interi ('a' e' b') e calcola il prodotto di questi due numeri. Tuttavia, ci sono alcuni errori nella gestione degli input e nel calcolo del prodotto, come discusso precedentemente.

Funzione Dividi:
La funzione' dividi()' richiede all'utente di inserire due numeri interi (a e b) e calcola il resto della divisione di 'a' per' b'. Tuttavia, ci sono anche errori nella gestione degli input e nel calcolo del risultato, come discusso precedentemente.

Funzione Inserisci Stringa:
La funzione 'ins_string()' richiede all'utente di inserire una stringa di massimo 9 caratteri. Tuttavia, non gestisce correttamente il caso in cui l'utente inserisce una stringa più lunga di 9 caratteri, potenzialmente causando un overflow del buffer.

Errori di Input Handling:
Il programma manca di gestione completa degli input non validi o imprevisti. Ad esempio, se l'utente inserisce un carattere non valido durante la selezione del menu o inserisce una stringa invece di un numero durante le operazioni di moltiplicazione o divisione, il programma può

comportarsi in modo imprevedibile.

Un potenziale errore di overflow si trova nella funzione ins_string(), quando l'utente inserisce una stringa più lunga di 9 caratteri. Questa situazione può causare un overflow del buffer stringa, poiché la dimensione dell'array è definita come char stringa[10].

Proporre una soluzione per ognuno di essi.

Gestione dell'input non valido nel menu:
Aggiungi un controllo nel blocco switch per gestire input non validi e stampare un messaggio di errore. Usa un loop per richiedere l'input finché non viene fornito un input valido.

Gestione dell'input nel metodo moltiplica:
Aggiungi controlli per gestire input non validi e assicurarti che l'utente inserisca numeri corretti.

Gestione dell'input nel metodo dividi:
Aggiungi controlli per gestire input non validi e assicurarti che l'utente inserisca numeri corretti e che il denominatore non sia zero.

Gestione della stringa nel metodo ins_string:
Limita la lunghezza della stringa inserita e gestisci il caso in cui l'utente inserisce una stringa più lunga di quanto il buffer può gestire.

Errori di Input Handling:
Implementa un loop e una gestione degli errori per garantire che l'utente fornisca input corretti durante la selezione del menu e durante le operazioni di moltiplicazione e divisione.

Errore overflow
Per risolvere l'errore di overflow è possibile utilizzare 'fgets()' invece di 'scanf()'
.inoltre è necessario rimuovere il carattere di nuova linea('\n')
aggiunto da 'fgets()'