Il programma in questione si prepone come “Assistente digitale” per l’esecuzione di alcune funzioni basilari:

- Moltiplicazione tra due numeri
- Divisione tra due numeri
- Inserimento di una stringa (di utilità pari a 0)

Di seguito il riepilogo del codice con i relativi punti errati o inutili ai fini del funzionamento del programma:

```
#include <stdio.h>

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


int main ()

{
    char scelta = {'\0'}; //Nella dichiarazione di scelta, 'scelta' è dichiarata come char, ma viene utilizzata come int
    menu ();
    scanf ("%d", &scelta); //Qui si dovrebbe utilizzare %c per leggere un carattere al posto di %d che è invece usato per i numeri interi

    switch (scelta)
    {
        case 'A':
        moltiplica();
        break;
        case 'B':
                dividi();
                break;
        case 'C':
                ins_string();
                break;
    }                           //Il programma si chiude senza restituire risposta all'utente, si dovrebbero aggiungere delle righe di default che avvertono l'utente di eventuali errori o di uscita dal programma (ad es.
default:
    printf("Scelta non valida.\n");
```

```c
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; La variabile inizializzata a zero nella funzione
moltiplica() è inutile, poiché viene sovrascritta quando l'utente inserisce
il valore del denominatore

    printf ("Inserisci i due numeri da moltiplicare:");
    scanf ("%f", &a); // Qui si sta utilizzando %f per leggere un numero
intero (short int). Si dovrebbe usare %hd per leggere un tipo di dato short
int
    scanf ("%d", &b); // Qui si sta utilizzando %d per leggere un numero
intero (short int). Si dovrebbe usare %hd per leggere un tipo di dato short
int

    short int prodotto = a * b;

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


void dividi ()
{
        int  a,b = 0; La variabile inizializzata a zero nella funzione
dividi() è inutile, poiché viene sovrascritta quando l'utente inserisce il
valore del denominatore, inoltre sarebbe meglio usare la variabile float
piuttosto che la variabile int per una divisione
        printf ("Inserisci il numeratore:");
        scanf ("%d", &a);
    printf ("Inserisci il denumeratore:"); //La parola denumeratore non
esiste nel vocabolario di lingua italiana, forse si intendeva dire
“denominatore”
        scanf ("%d", &b);
```

```
// Qui sarebbe bene aggiungere una funzione di controllo per evitare la divisione per 0 (ad es. un if/else)

        int divisione = a % b; //Qui si sta adoperando l'operatore % per ottenere il resto di una divisione, bisognerebbe invece usare l'operatore / per ottenere una divisione

        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);
  // La scanf %s potrebbe causare overflow se l'utente inserisce una stringa più lunga di 9 caratteri. Si dovrebbe specificare la larghezza massima consentita, ad esempio %12s. Ad ogni modo potrebbe essere più appropriato utilizzare la funzione fgets() per leggere una linea intera.

//Qui si dovrebbe aggiungere una riga di codice che richiama la stringa inserita, così da comprovarne l'utilità effettiva (ad es.
printf("Stringa inserita: %s\n", stringa);


}
```

Il programma inoltre non gestisce completamente l'input non valido. Ad esempio, se l'utente inserisce un carattere anziché un numero quando richiesto di inserire un numeratore o un denominatore, la scanf potrebbe lasciare il buffer di input in uno stato imprevisto.