Progetto Epicode S2/L5.

Ecco una spiegazione su perchè il codice originale presentava problemi:

1. Inizializzazione di "scelta": Le parentesi graffe { } sono utilizzate per inizializzare array, ma 'scelta' è un singolo carattere. L'utilizzo di { } in questo contesto causerebbe un errore di tipo, quindi dovrebbe essere inizializzato con le virgolette singole '0'.

2. Input non corretto in "scanf": %d in scanf è utilizzato per leggere un intero, ma "scelta" è di tipo carattere. Ciò avrebbe potuto causare comportamenti imprevisti, inclusi i potenziali errori di lettura da parte dell'utente. Quindi bisognava mettere (%c).

3. Dichiarazione di variabili in "moltiplica" e "dividi": Nelle funzioni moltiplica e dividi, la dichiarazione di a e b sulla stessa linea può portare a comportamenti imprevisti. b viene inizializzato a 0, ma a non ha un valore iniziale definito.

4. Input errato in "moltiplica": %f in scanf è utilizzato per leggere un numero in virgola mobile, ma a è dichiarato come short int. Ciò avrebbe potuto causare un comportamento imprevisto, inclusi errori di conversione. Questo avrebbe causato errori se l'utente avesse inserito un intero.

5. Operazione di modulo in "dividi": L'operatore % restituisce il resto della divisione intera, non il risultato della divisione. Questo avrebbe causato un risultato errato quando si cerca di calcolare la divisione tra a e b. Per ottenere il risultato della divisione è stato corretto utilizzando l'operatore /.

6. Dichiarazione di "prodotto" in "moltiplica": prodotto è dichiarato come short int, il che potrebbe portare a un overflow se il risultato della moltiplicazione supera i limiti di rappresentazione di short int. È corretto dichiarandolo come int.

7. Dimensione dell'array "stringa": L'array "stringa" è dichiarato con una dimensione

di 10, che potrebbe essere insufficiente per alcune stringhe più lunghe, causando overflow del buffer durante la lettura.

In sostanza, questi errori potrebbero causare comportamenti imprevisti, malfunzionamenti e potenziali problemi di sicurezza nel programma. Le correzioni proposte mirano a risolvere questi problemi e rendere il codice più robusto e sicuro.

```
#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);
}
```

Sotto

Usare virgoletta singola per dichiarare un carattere

"%c" perche scelta è di tipo char
per leggere un carattere

CASO A

prodotto

"%d" = input di interi

CASO B

int a,b;

}else{
printf("impossibile dividere per 0");
}

CASO C

[50]

non utilizzare & perché leggi la stringa con %s

```c
#include <stdio.h>
void menu();
void moltiplica();
void dividi();
void ins_string();

int main() {
  char scelta = 'O';
  menu();
  scanf("%c", &scelta);

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

  return 0;
}

void menu () {
  printf("Benvenuto, sono un assistente 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() {
  int a,b = 'O';
  printf("Inserisci i due numeri da moltiplicare: ");
  scanf("%d", &a);
  scanf("%d", &b);
  int prodotto = a * b;
  printf("Il prodotto tra %d e %d è: %d, a,b,prodotto");
 }

void dividi() {
```

```
    break;
  case 'C':
    ins_string();
    break;
  }

  return 0;
}

void menu () {
  printf("Benvenuto, sono un assistente 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() {
  int a,b = 'O';
  printf("Inserisci i due numeri da moltiplicare: ");
  scanf("%d", &a);
  scanf("%d", &b);
  int prodotto = a * b;
  printf("Il prodotto tra %d e %d è: %d, a,b,prodotto");
 }

void dividi() {
  int a,b;
  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 è: %d, a,b,divisione");
 }

void ins_string() {
  char stringa[50];
  printf("Inserisci la stringa: ");
  scanf("%s", stringa);
}
```