Int Main()

- Char **scelta = {'\0'}.**
  In questo caso le parentesi graffe sono usate per assegnare valori alle stringhe. Scelta ha il ruolo di singolo carattere e non di stringa.
  Soluzione: **Char scelta;**
- scanf ("'%d", &scelta)
  Scelta è di tipo char quindi invece di usare il %d dobbiamo usare il %c.
  Soluzione: **scanf("%c", &scelta);**
- Controllo case sensitive su **scelta**. L'utente potrebbe inserire un carattere minuscolo anziché maiuscolo e questo non verrebbe letto.
  Soluzione: Si potrebbe usare una funzione come **toupper()**. Questa funzione prende, ad esempio, la variabile char scelta e ne trasforma il contenuto. Es.
  scelta = 'a' (valore inserito dall'utente)
  scelta = toupper(scelta) -> scelta = 'A'
  nel codice scriveremo in questa maniera: **scelta = toupper(scelta);**
- Controllo input diverso da quelli del menù. Se l'utente inserisce, ad esempio, 'd' il programma non farà altro che terminare visto che nessuna casistica nello switch è verificata.
  Soluzione: Possiamo inserire un case di default all'interno dello switch cosi da gestire il caso in cui nessun case è verificato;

```
default:
        printf("\nOpzione inesistente." );
break;
```

void moltiplica ()

- scanf ("%f", &a). Nella funzione abbiamo scanf ("%f", &a) per l'inserimento della variabile a. Questa variabile è però uno short int e non un float.
  Soluzioni 1: convertire a e b in int e modificare negli scanf il %f in %d;
  Soluzione 2: mantenere lo short int ma modificare in entrambi gli scanf da %f in %hd:
  scanf ("%hd", &a); scanf ("%hd", &b);
- Aggiunta di un printf. Si consiglia di separare il printf per l'inserimento dei dati in due distinti printf così da rendere più chiaro l'inserimento di dati da parte dell'utente.
  Soluzione:

```
printf ("Inserisci il primo numero da moltiplicare: ");
scanf ("%d, &a);
printf ("\nInserisci il secondo numero da moltiplicare:");
scanf ("%d", &b);
```

- short int prodotto = a * b. È consigliabile dichiarare le variabili all'inizio della funzione per una visione più chiara del progetto.
- short int a,b = 0. Così facendo solo la variabile b è settato a 0 mentre la a rimane con un numero stabilito dal sistema.
  Soluzioni: short int a = 0,b = 0;

void dividi()

- int a,b = 0. Così facendo solo la variabile b è settato a 0 mentre la a rimane con un numero stabilito dal sistema.
- Tipo intero. Visto che la funzione deve dividere due valori bisogna usare il tipo float sia per le variabili a e b che per la variabile divisione.
  Soluzione: float a, b, divisione;

- Simbolo divisione. La formula per la divisione è sbagliata in quanto vi è il simbolo del modulo (%) invece di quello della divisione (/).
  Soluzione: divisione = a / b;
- Errore di stampa. Visto che il tipo corretto da usare è float dobbiamo modificare il printf sostituendo i %d con %f. Per avere una visione più chiara possiamo usare il %,2f che riduce le cifre significative a 2.
  Soluzione: printf ("La divisione tra %.2f e %.2f e': %.2f", a, b, divisione);
- Controllo denominatore. Se il denominatore della funzione è uguale a 0 la divisione non è possibile e porta ad un errore. Per evitare che ciò accada dobbiamo inserire un controllo.
  Soluzione:

```
if ( b  != 0 ) {
        float divisione = a / b;
        printf("La divisione tra %.2f e %.2f e': %.2f", a, b, divisione);
}else
        printf("\nIl denominatore deve essere diverso da zero.\n");
```

Volendo possiamo aggiungere un ciclo do-while per ripetere l'inserimento del denominatore. La condizione del do-while sarà: b == 0.

void ins_string ()

- String overflows. L'utente potrebbe inserire una stringa superiore al limite massimo (10 in questo caso).
  Soluzione: scanf ("%10s", &stringa). Il %10s farà in modo che ci sia un limite nel numero di caratteri che il programma può accettare impedendo così lo string overflows.
- Ulteriori implementazioni. Questa funzione non fa nulla oltre che prendere in input una stringa. Magari aggiungere almeno una funzione di print per aver il feedback del corretto inserimento. **Soluzione**: printf("%s",stringa);

Miglioramenti

- Ciclo do-whie().Possiamo inserire un ciclo do-while per far ripetere il programma fino a quando l'utente non inserisce una opzione valida o quella di uscita.
  Soluzione: Possiamo aggiungere nella funzione menu() un opzione per l'uscita con, ad esempio, la lettera 'D' come trigger. Dopo implementiamo il ciclo do-while.

```
do{
        menu()
        |
        |
        |
}while(scelta != 'D');
```

- Formattazione del codice. Per maggiore chiarezza una corretta formattazione del codice è fondamentale