ESERCIZIO SETTIMANALE WEEK 2

L'esercizio questa settimana consisteva nel leggere, correggere e eventualmente migliorare un codice in linguaggio C.

FASE 1

Per prima cosa ho letto il codice e iniziato a segnare con dei commenti le parti che non andavano, inserendoli tra "/* */". Questo mi ha permesso di non "sporcare" il codice e avere una situazione più chiara.

```
char scelta = '\0' ; /* qui hotolto le parentesi graffe perchè esse delimitano un blocco di codice */
```

Esempio di commento

Il programma ad un primo sguardo permetteva di fare operazioni matematiche molto semplici, addizione sottrazione e divisione, con l'aggiunta di una stampa a video di numeri o caratteri tramite una stringa.
Il tutto introdotto da un piccolo menù iniziale.

ISPEZIONE SWITCH MAIN
Nel main vi era una scelta (switch) che permetteva di scegliere quale operazione andare ad eseguire. In questo caso ho aggiunto altre scelte e il default che in questo caso era assente.

Per far si che anche inserendo lettere minuscole oppure numeri questo menù potesse funzionare e far proseguire l'utente.

```
switch (scelta) /* Il comando switch è utile per valutare caso per caso i valori di una variabile.In questo cado ho aggiunto più possibilità rispettoal codice originale*/
{
        case 'A' ;  /* L'elenco dei case a/b/c sono i valori della variabile (scelta) che potrebbe assumere. In base al suo valore si eseguira l'istruzione scritta. */
        moltiplica();
        break;      /* Il Break dopo ogni case permette di uscire dallo switch*/

        case 'a':       /* Ho messo la possibilità di inserire anche lettere minuscole come nel caso della scelta dell'esercizio in C di questi giorni*/
        moltiplica();
        break;

        case 'B' :
        dividi();
        break;

        case 'b' :      /* Lettera minuscola anche qui*/
        dividi();
        break;

        case 'C':
        ins_string();
        break;

        case 'c' :       /* Lettera minuscola anche qui*/
        ins_string();
        break;

        default :       /* Per un errore logico qui ho inserito il "Default" che viene eseguito se i casi in precedenza non sono soddisfatti*/
        printf("**Errore**\n Inserisci una scelta valida\n");  /* Se si inserirebbe un valore errato come per esempio un numero in questo comparirebbe il messaggio di errore che in precedenza era assente*/
        scanf("%s", &scelta);
        break;
```

ISPEZIONE MENÙ
Il menù era tutto correttamente scritto. Non ho notato errori logici, tranne per un piccolo errore di sintassi. Un piccolo spazio all'inizio dell'ultima **printf.**

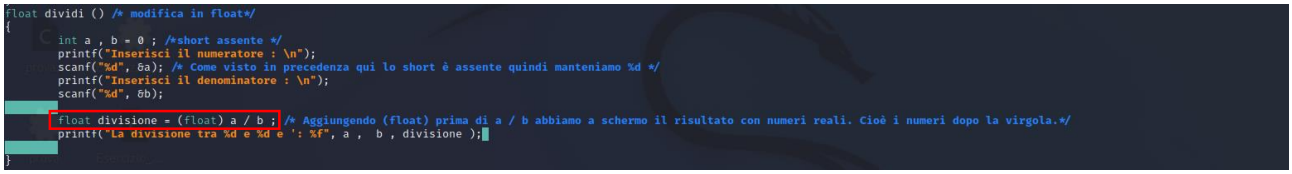
```
void menu ()

{
        printf("Benvenuto, sono un assistente digitale, posso aiutarti a sbrigare alcuni compiti\n");
        printf ("Come posso aiutarti?\n");
        printf(" A >> Moltiplicare due numeri \n B >> Dividere due numeri\n C >> Inserire una stringa\n"); /*Qui ho aggiunto uno spazio prima della A >> per allineare il tutto*/
}
```

ISPEZIONE MOLTIPLICA
Nella sezione moltiplica ho cambiato il tipo in int da void. Avendo solamente input e output con numeri interi. Con il tipo short int si intendeva interi con valore massimo di 2byte. Le modifiche che ho apportato sono state: \n nel primo pritf per mandare a capo, nel primo scanf ho sostituito il %f con %hd.
%f è utilizzato per numeri reali di 4byte e mi è sembrato più appropriato inserire %hd per essere coerente con lo short int. Stessa cosa nel secondo scanf solo che qui era presente %d, utilizzato per numeri interi da 4byte.

```
short  int  a , b = 0  ;
printf("\nInserisci i due numeratori da moltiplicare: \n"); /* \n visto anche in precedenza serve per n ritorno a capo. Qui era assente*/
scanf("%hd", &a); /* %f è utilizzato per interi. In questo caso è più appropriato %hd perchè la shor int accetta solo interi*/
scanf("%hd", &b); /* %d è utilizzato per interi. In questo caso short int indica interi di 2byte(short) invece che 4byte(int).Anche in questo caso è più appropriato %hd*/
```

ISPEZIONE DIVIDI

Nella sezione dividi ho cambiato in float da void. Avendo la possibilità di avere risultati con numeri reali ho cambiato il tipo. Aggiungendo float a divisione e (float) dopo l'uguale abbiamo come risultati numeri reali con il resto in stampa



ISPEZIONE INS_STRINGA

Nella sezione ins_stringa ho aggiunto fgets che limita i caratteri massimi inseribili e una printf che ci mostra a video la stringa inserita in precedenza