"PROGRAMMA DA ANALIZZARE"

```
#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;(ci va lo slash)
```

```
        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);
}
```

"RICHIESTA DELL'ESERCIZIO"

Traccia:

per agire come un Hacker bisogna capire come pensare fuori dagli schemi. L'esercizio di oggi ha lo scopo di allenare l'osservazione critica.

Dato il codice su scritto si richiede di

1. Capire cosa fa il programma senza eseguirlo
2. Individuare dal codice sorgente le caratteristiche non standard che il programma non gestisce (comportamenti potenziali che non sono stati contemplati)
3. Individuare errori di sintassi/logici
4. Proporre una soluzione per ognuno di essi.

"SVOLGIMENTO"

(punto 1)

Il seguente programma ci consente di accedere ad un **ASSISTENTE VIRTUALE** che ci proporrà due operazioni che il programma sarà in grado di farci eseguire: **MOLTIPLICAZIONI E DIVISIONI** .

(punto 2,3,4)

- Nel comando di indentificato con "CHAR SCELTA " vi e un errore di debug cioè il codice ci butterà fuori dal programma in qualsiasi altro caso come riportato qui sotto ed evidenziato nel cerchio blu, mentre nel cerchio evidenziato in rosso il char viene trattato come un **"ARRAY"**:

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

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

Una delle possibili soluzioni e eliminare **"\0"** e lasciare solo gli apici (**''**) all'interno.(in questo caso ci stiamo riferendo a cio che e all'interno del cerchio rosso)
Mentre all'interno di cio che è evidenziato nel cerchio blu va sostituito con il seguente codice :

```
default: printf("non posso andare avanti"\n);
retourn 0;
break;
```

- Nella riga di comando "**VOID DIVIDI**" vi è un errore di logica inquanto il programma non eseguirà un operazione di divisione come si evince dal **cerchio rosso** inoltre la funzione così strutturata ci permetterà di dividere un numero per 0 creando incongruenze "**guardare all'interno dei cerchi blu**".

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

        int divisione = a % b;(ci va lo slash)

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

Per tutto ciò che e evidenziato in rosso la soluzione è sostituire "**%**" con uno "**/**"
Mentre per ciò che è evidenziato in blu basta sostituire "**%d**" con "**%f**"

- Nel campo "**VOID INS_STRING()**", guarda immagine sotto

```
void ins_string ()
{
      char stringa[10];
         printf ("Inserisci la stringa:");
         scanf ("%s", &stringa);
}
```

In questo campo vi sono due errori (**logici e ortografici**)e un errore grave di **OVERFLOW**
La risoluzione del problema potrebbe essere nel punto evidenziato in **GIALLO**
VI E UN ERRORE DI OVERFLOW.

Es. corretto : **char stringa[];**

Mentre nel punto evidenziato in **VERDE** va eliminato il comando end "&"

Es.corretto : **scanf("%s", stringa);**