

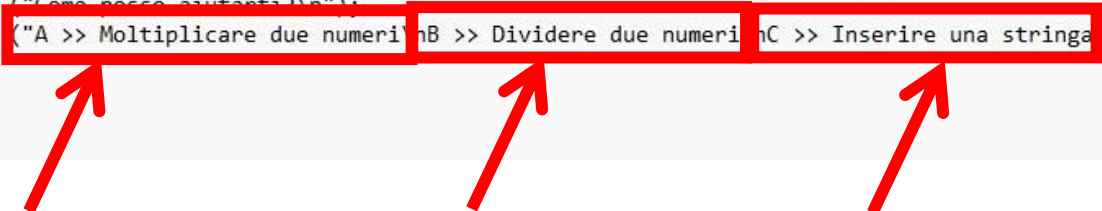
## BAG HUNTING 12/05/2023

### 1) COSA CI PERMETTE DI FARE IL PROGRAMMA?

Nel programma proposto come modello, scritto in linguaggio C, compare a console un menù che ci dà la possibilità di scegliere 3 opzioni a seconda del “char” selezionato.

Nello specifico, con il comando “A” il programma ci consente di svolgere una moltiplicazione tra due numeri, con il comando “B” una divisione tra numeratore e denominatore, mentre con il comando “C” abbiamo la possibilità di inserire una stringa di testo.

```
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");
}
```



### 2) CASISTICHE STANDARD CHE IL PROGRAMMA NON GESTISCE

Il programma in questione al suo interno ha molti comportamenti non contemplati.

- Notiamo subito che manca il caso di default nello switch case, motivo per cui nel caso l'utente immettesse una lettera diversa da A, B o C, il programma si chiuderebbe, senza dar possibilità all'utente di immettere nuovamente un altro carattere;
- Notiamo altresì che nel programma non è stata contemplata la possibilità e risoluzione di inserimento a console di numeri non interi in mancanza della dichiarazione **float**, per l'appunto, mai dichiarata;
- Ho notato altresì che per quanto riguarda l'opzione C (“inserisci una stringa”) non è considerata la digitazione di un Array di char che abbia più di 10 caratteri, motivo per cui, al superamento della soglia limite, il programma si chiuderebbe;
- In ultimo, bisogna segnalare che, svolta una delle operazioni (A, B o C), il programma non dà all'utente la possibilità di continuare il programma, in quanto lo stesso, dopo aver svolto un'operazione, dovrebbe chiudersi.


3) **4) INDIVIDUAZIONE DI ERRORI DI SINTASSI ED ERRORI LOGICI (IN ROSSO) E RISOLUZIONE PER OGNUNO DI ESSI (IN BLU)**

All'interno del codice del programma in esame vi sono altresì errori di sintassi ed errori logici. In totale ne ho trovati **7**.

**PRIMO ERRORE (in rosso) E RELATIVA SOLUZIONE (in blu)**

- **char scelta = {'\0'};** in cui lo stesso sarebbe “end of string” risulta un comando sbagliato in quanto le parentesi non hanno senso; comando giusto è **char scelta = '\0';**

```
int main ()
{
    char scelta = {'\0'};
    menu ();
    scanf ("%d", &scelta);          // ("%d") sbagliato, va messo ("%c")
    switch (scelta)
    {
```



**SECONDO ERRORE (in rosso) E RELATIVA SOLUZIONE (in blu)**

- **scanf ("%d", &scelta);** è sbagliato, va messo **( "%c" )**, in quanto **&d** si usa per gli interi e non per i caratteri, dove invece si usa **%c**. Senza **%c**, motivo per cui, inserendo un intero anziché un carattere, il programma in esame non partirebbe;

```

int main ()
{
    char scelta = {'\0'}; // lo
    menu ();
    scanf ("%d", &scelta); // ("%d") sbagliato, va messo ("%c")
    switch (scelta)
    {
        case 'A':

```

### TERZO ERRORE, QUARTO E QUINTO ERRORE (in rosso) E RELATIVA SOLUZIONE (in blu)

- **scanf ("%f", &a);** e **scanf ("%d", &b);** anche in questo caso (all'interno del **void moltiplica**) vi è un altro errore, in quanto, in quanto **a** e **b** sono dichiarati come **short int**, motivo per cui, essendo più piccoli di un intero, non si può usare &d bensì **&hd**. Si specifica altresì che nel primo **scanf ("%f", &a)** (errore segnalato con cerchio giallo in figura), che non è mai stato dichiarato, essendo stato dichiarato soltanto **short int a,b = 0**. Ergo, in entrambi i casi si dovrebbe usare **scanf ("%hd", &a);** e **scanf ("%hd", &b);**

```

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);
}

```

LEGENDA: Quarto e quinto errore evidenziati da rettangoli rossi, terzo errore evidenziato da freccia e cerchio giallo


### SESTO ERRORE (in rosso) E RELATIVA SOLUZIONE (in blu)

- Come detto in precedenza, all'interno del **void dividi**, sono presi in considerazione soltanto numeri interi e non i numeri reali (float), motivo per cui le uniche operazioni possibili potrebbero essere tra numeri interi (escluso 0). Si specifica che nel caso in cui si dovesse dividere per un numero più grande, l'utente potrebbe vedere soltanto la parte intera del numero; si noti altresì l'errore **int divisione = a % b**, specifichiamo che % non è l'operatore della divisione, ma del resto, ergo per svolgere correttamente una divisione è necessario correggere e scrivere **int divisione = a/b**;

```
void dividi ()
{
    int a,b = 0;
    printf ("Inserisci il numeratore:");
    scanf ("%d", &a); // metto f al posto di d
    printf ("Inserisci il denominatore:");
    scanf ("%d", &b); // metto f al posto di d

    int divisione = a % b;

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



### SETTIMO ERRORE (IN ROSSO) E RELATIVA SOLUZIONE (in blu)

- scanf ("%s", &stringa);** è sbagliato in quanto vi è la **& di troppo**, la forma corretta è **scanf ("%s", stringa);**

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

