





Capisci cosa fa il programma :

Quando il programma viene avviato inizializza una variabile char su null, poi avvia la funzione menu. La funzione menu stampa tre righe di testo su terminale e chiede all'utente di scegliere se effettuare un prodotto tra due valori, se effettuare un quoziente tra due valori o se inserire una stringa; per effettuare la scelta chiede di dare in input rispettivamente A, B o C. A questo punto esce dalla funzione menu e riceve l'input dell'utente. A seguire fa partire uno switch con tre case a seconda di ciò che è stato inserito dall'utente.

Se l'utente ha inserito A il programma entra nel case A e fa partire la funzione moltiplica. La funzione moltiplica definisce due variabili short int a e b e inizializza b=0, poi stampa su terminale la richiesta di inserimento di due numeri, a questo punto l'utente inserisce il primo valore e la funzione assegna il valore inserito alla variabile a, poi l'utente inserisce un altro valore e la funzione lo assegna alla variabile b. A questo punto la funzione definisce una variabile short int prodotto e gli assegna il valore dato dal risultato del prodotto tra la variabile a e la variabile b, dopodiché stampa su terminale il valore della variabile a, della variabile b, ed infine il valore della variabile prodotto. Dopodiché esce dallo statement switch e incontra il comando return 0 che termina il programma.



Capisci cosa fa il programma:

Se l'utente ha inserito B il programma entra nel case B e fa partire la funzione `dividi`.

La funzione `dividi` definisce due variabili `int a` e `b` con `b` inizializzata a 0, poi stampa su terminale la richiesta all'utente di "inserire il numeratore".

L'input dell'utente viene assegnato alla variabile `a`, a seguire il programma stampa su terminale la richiesta di "inserire il denominatore".

L'input dell'utente viene assegnato alla variabile `b`.


Adesso la funzione definisce una variabile `int divisione` e gli assegna il valore del resto del quoziente tra il valore della variabile `a` e quello della variabile `b`.

Dopodichè stampa su terminale il valore della variabile `a`, quello della variabile `b` e quello della variabile `divisione`, poi esce dallo `statement switch` e trova `return 0` che fa terminare il programma.

Se l'utente ha inserito C il programma entra nel case C e fa partire la funzione `ins_string`.

La funzione `ins_string` definisce un array `char stringa` di dimensione 10, poi stampa su terminale la richiesta all'utente di "inserire la stringa", dopodichè la funzione assegnerà l'input dell'utente all'array.

Fatto ciò uscirà dallo `statement switch` e troverà il comando `return 0` che farà terminare il programma.



Individuare dal codice sorgente le casistiche non standard che il programma non gestisce (esempio, comportamenti potenziali che non sono stati contemplati)

Il programma non è in grado di gestire alcun tipo di input diverso da "A" , "B" e "C" quando viene chiamato lo statement switch.

la funzione dividi non è in grado di gestire alcun tipo di input diverso da valori interi molto piccoli.

La funzione moltiplica non è in grado di gestire input diversi da valori interi.

La funzione ins_string non è in grado di gestire input più lunghi di 10 caratteri alfa-numeric.

Individuare eventuali errori di sintassi / logici:

errore riga 14: per assegnare correttamente il valore alla variabile char si usa %c

errore riga 27: nello statement switch è assente il default case.

```
9  int main ()
10
11  {
12      char scelta = {'\0'};
13      menu ();
14      scanf ("%d", &scelta); //%d è un errore, la variabile char richiede %c
15
16      switch (scelta)
17      {
18          case 'A':
19              moltiplica();
20              break;
21          case 'B':
22              dividi();
23              break;
24          case 'C':
25              ins_string();|
26              break;
27          //manca lo statement default
28      }
29
30      return 0;
31
32  }
```

Individuare eventuali errori di sintassi / logici:

errore riga 46: la variabile short int non permette la divisione tra numeri reali.

errore riga 48, 49 e 53: la variabile short int richiede %hd

errore riga 51: l'uso della variabile short int per un prodotto potrebbe risultare in un errore di stack overflow.

```
35 void menu ()
36 {
37     printf ("Benvenuto, sono un assistente digitale, posso aiutarti a sbrigare alcuni compiti\n");
38     printf ("Come posso aiutarti?\n");
39     printf ("A >> Moltiplicare due numeri\nB >> Dividere due numeri\nC >> Inserire una stringa\n");
40
41 }
42
43
44 void moltiplica ()
45 {
46     short int a,b = 0; //short int non permette la moltiplicazione tra numeri reali
47     printf ("Inserisci i due numeri da moltiplicare:");
48     scanf ("%f", &a); //short int richiede %hd
49     scanf ("%d", &b); //short int richiede %hd
50
51     short int prodotto = a * b; //short int non permette la moltiplicazione tra numeri reali, inoltre è facile che vada in stack overflow
52
53     printf ("Il prodotto tra %d e %d e': %d", a,b,prodotto); //short int richiede %hd
54 }
55
56
```

Individuare eventuali errori di sintassi / logici:

errore riga 59: la variabile int non permette la divisione tra numeri reali.

errore riga 65: l'operatore corretto per effettuare la divisione è "/"

(da notare che tutte le righe di codice sono segnate in giallo, questo significa che il programma non è mai stato compilato)

```
57 void dividi ()
58 {
59     int a,b = 0; //int non permette la divisione tra numeri reali
60     printf ("Inserisci il numeratore:");
61     scanf ("%d", &a); //il programma non gestisce l'inserimento di caratteri
62     printf ("Inserisci il denominatore:");
63     scanf ("%d", &b);
64
65     int divisione = a % b; // l'operatore corretto per la divisione è "/"
66
67     printf ("La divisione tra %d e %d e': %d", a,b,divisione);
68
69 }
70
71
72
73
74
75 void ins_string ()
76 {
77     char stringa[10];
78     printf ("Inserisci la stringa:");
79     scanf ("%s", &stringa);
80     //il programma non gestisce l'inserimento di più di 10 caratteri
81 }
```

Proporre una soluzione:

Vediamo le soluzioni ai problemi riscontrati nel codice sorgente nel nuovo codice che ho scritto:

Adesso vengono utilizzati valori numerici per selezionare la scelta dell'operazione da effettuare; il codice controlla che l'input dell'utente sia un numero intero compreso tra 1 e 4, qualora l'utente inserisse valori diversi da quelli accettabili il programma restituisce un messaggio di errore e ripropone la scelta all'utente esortandolo ad inserire uno dei valori corretti.

```
1  #include <stdio.h>
2
3  void menu ();
4  void moltiplica ();
5  void dividi ();
6  void ins_string();
7
8  int main()
9  {
10     int scelta;
11
12     printf("Questo programma ti da la possibilita' di effettuare tre operazioni.\n");
13
14     while (1<2)
15     {
16         do {
17             menu ();
18
19             if (scanf("%d", &scelta) != 1 || scelta < 1 || scelta > 4)
20             {
21                 while (getchar() != '\n');
22                 printf("\nInput non valido, per favore inserire un numero intero fra 1 e 4.\n\n");
23             } else {
24                 break;
25             }
26         } while (1<2);
27
28         switch (scelta)
29         {
30             case 1:
31                 moltiplica();
32                 break;
33             case 2:
```

```
48     void menu ()
49     {
50         printf("Digita il numero corrispondente per effettuare la scelta:\n\n");
51         printf("1. Moltiplica due numeri\n");
52         printf("2. Dividi due numeri\n");
53         printf("3. Inserisci una stringa\n");
54         printf("4. Esci dal programma\n");
55         printf("\nInserisci la tua scelta (1-4): ");
56     }
```


Proporre una soluzione:

Se l'utente inserisce un valore numerico intero compreso tra 1 e 4 il programma esegue uno switch e avvia la funzione corrispondente al numero inserito dall'utente;

```
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46

switch (scelta)
{
    case 1:
        multiplica();
        break;
    case 2:
        dividi();
        break;
    case 3:
        ins_string();
        break;
    case 4:
        return 0;
        break;
}
```

Proporre una soluzione:

Adesso la funzione `moltiplica` usa variabili `float` per calcolare il prodotto, questo permette di effettuare moltiplicazioni anche tra numeri decimali.

Lo stesso tipo di controllo sull'input dell'utente che abbiamo visto nella funzione `main` viene riproposto anche qua., Se l'input inserito dall'utente non è un valore numerico la funzione restituisce un messaggio d'errore e ripropone l'inserimento del valore all'utente.

```
58 void moltiplica ()
59 {
60     float a=0 , b=0;
61     printf ("\nInserisci il primo valore da moltiplicare: ");
62
63     do {
64         if (scanf("%f", &a) != 1)
65         {
66             while (getchar() != '\n');
67             printf ("\nInput non valido, per favore inserire un valore numerico.\n");
68             printf ("\nInserisci il primo valore da moltiplicare: ");
69         } else {
70             break;
71         }
72     } while (1<2);
73
74     printf ("\nInserisci il secondo valore da moltiplicare: ");
75
76     do {
77         if (scanf("%f", &b) != 1)
78         {
79             while (getchar() != '\n');
80             printf ("\nInput non valido, per favore inserire un valore numerico.\n");
81             printf ("\nInserisci il secondo valore da moltiplicare: ");
82         } else {
83             break;
84         }
85     } while (1<2);
86
87     float prodotto = a * b;
88
89     printf ("\nIl prodotto tra %f e %f e': %f\n\n", a,b,prodotto);
90 }
```

Proporre una soluzione:

La funzione `dividi` è stata modificata in modo da essere molto simile alla funzione `moltiplica`.

Le variabili sono di tipo `float` e l'input dell'utente deve essere di tipo numerico altrimenti viene mostrato un messaggio d'errore.

Per quanto riguarda il denominatore viene visualizzato un messaggio d'errore anche quando viene inserito il valore '0' in quanto la divisione per 0 non è un'operazione definita in matematica.

```
92 void dividi ()
93 {
94     float a=0 ,b=0;
95     printf ("\nInserisci il numeratore: ");
96
97     do {
98         if (scanf("%f", &a) != 1)
99         {
100             while (getchar() != '\n');
101             printf ("\nInput non valido, per favore inserire un valore numerico.\n");
102             printf ("\nInserisci il numeratore: ");
103         } else {
104             break;
105         }
106     } while (1<2);
107
108     printf ("\nInserisci il denominatore: ");
109
110     do {
111         if (scanf("%f", &b) != 1 || b==0)
112         {
113             while (getchar() != '\n');
114             printf ("\nInput non valido, per favore inserire un valore numerico diverso da 0.\n");
115             printf ("\nInserisci il denominatore: ");
116         } else {
117             break;
118         }
119     } while (1<2);
120
121     float divisione = a / b;
122
123     printf ("\nLa divisione tra %f e %f e': %f\n\n", a,b,divisione);
124 }
125
```

Proporre una soluzione:

Purtroppo non sono riuscito a trovare una soluzione al caso in cui l'utente inserisce una stringa più lunga della lunghezza dell'array quando viene definito.

Per mitigare il problema ho aumentato la dimensione dell'array a 100, ma il programma si chiuderà in modo anomalo se l'utente decide di inserire una stringa più lunga di 100 caratteri.

Dopo che una delle tre funzioni (moltiplicazione, divisione e ins_stringa) sono state eseguite correttamente il programma ripropone all'utente il menu per scegliere se effettuare una nuova operazione. Il programma può essere terminato dall'utente digitando '4' dal menu, questo è l'unico modo per terminare il programma.

```
126 void ins_string ()
127 {
128     char stringa[100];
129     printf ("\nInserisci una stringa: ");
130     scanf ("%s", &stringa);
131     printf ("\nLa stringa inserita e': %s\n\n", stringa);
132 }
133
```