

Epicode Project 2

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 in allegato, si richiede allo studente di:

- Capire cosa fa il programma senza eseguirlo
- Individuare dal codice sorgente le casistiche non standard che il programma non gestisce (esempio, comportamenti potenziali che non sono stati contemplati)
- Individuare eventuali errori di sintassi / logici
- Proporre una soluzione per ognuno di essi



Esercizio_10_Epicode.c

Svolgimento:

Apro il file con un block notes per farmi una prima idea generale del codice, d'ora in poi anche detto “traccia”.

```
Esercizio_5_Epicode_w2
File Modifica Visualizza

#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 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");
}

Linea 1, colonna 1
```

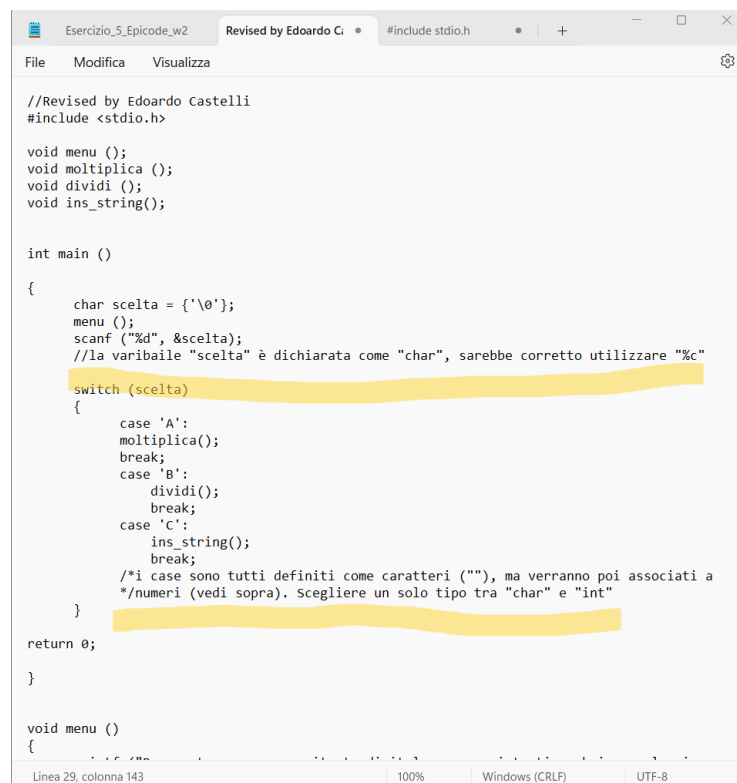
Da una prima occhiata deduco che dovrebbe trattarsi di un piccolo “assistente digitale” in grado di fornire tre tipi di aiuto: eseguire delle moltiplicazioni, delle divisioni, o inserire una stringa a nostro piacimento.

Altrettanto velocemente noto che la dichiarazione delle variabili avviene in momenti diversi, risulta “spezzettata”, verranno pertanto sistemate e dichiarate in maniera uniforme durante la nuova stesura del codice, per minimizzare gli errori di compilazione e successiva esecuzione; inoltre, la variabile “char” è inizializzata erroneamente.

Manca completamente, inoltre, la libreria “<string.h>”.

Per praticità, andrò a parlare degli altri errori trovati nel codice, raggruppandoli per tipo: di sintassi o logico/strutturali.

Per quanto riguarda la prima, si noti che:



```
//Revised by Edoardo Castelli
#include <stdio.h>

void menu ();
void moltiplica ();
void dividi ();
void ins_string();

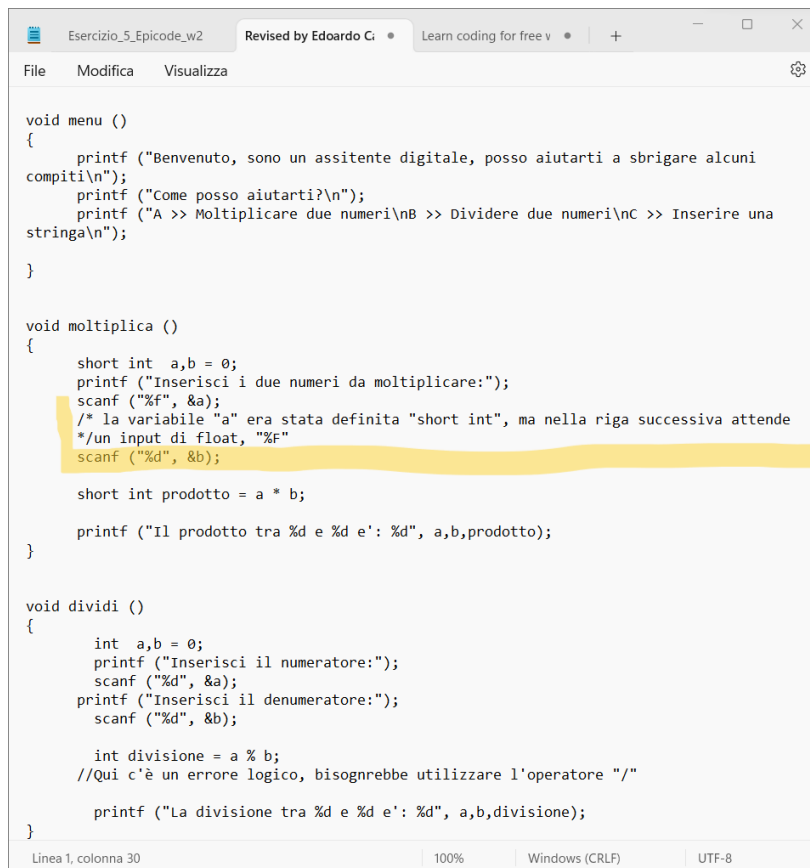
int main ()
{
    char scelta = {'\0'};
    menu ();
    scanf ("%d", &scelta);
    //la variabile "scelta" è dichiarata come "char", sarebbe corretto utilizzare "%c"
    switch (scelta)
    {
        case 'A':
            moltiplica();
            break;
        case 'B':
            dividi();
            break;
        case 'C':
            ins_string();
            break;
        /*i case sono tutti definiti come caratteri (""), ma verranno poi associati a
        */numeri (vedi sopra). Scegliere un solo tipo tra "char" e "int"
    }
    return 0;
}

void menu ()
{
    printf ("Menu\n");
    printf ("1. Moltiplica\n");
    printf ("2. Dividi\n");
    printf ("3. Inserisci stringa\n");
    printf ("4. Esci\n");
    printf ("Scegli un'opzione\n");
}
```

The screenshot shows a code editor with the following C code. Several lines are highlighted in yellow to indicate errors:

- Line 14: `scanf ("%d", &scelta);` - A comment indicates that the variable `scelta` is declared as `char`, so `%c` should be used instead of `%d`.
- Line 15: `switch (scelta)` - The variable `scelta` is a `char`, so it should be compared with character literals like `'A'`, `'B'`, and `'C'` instead of integers.
- Line 24: `return 0;` - A comment indicates that the `case` labels are defined as characters, but the program expects numeric input, suggesting a mismatch between the input type and the logic.

The editor interface includes a menu bar (File, Modifica, Visualizza), a toolbar, and a status bar at the bottom showing "Linea 29, colonna 143", "100%", "Windows (CRLF)", and "UTF-8".



```
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");
}

void moltiplica ()
{
    short int a,b = 0;
    printf ("Inserisci i due numeri da moltiplicare:");
    scanf ("%f", &a);
    /* la variabile "a" era stata definita "short int", ma nella riga successiva attende
    */un input di float, "%f"
    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 denominatore:");
    scanf ("%d", &b);

    int divisione = a % b;
    //Qui c'è un errore logico, bisognerebbe utilizzare l'operatore "/"

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

-1 Nella funzione “main()”, “scelta” è definita come un carattere, “char”, appunto. Nel commento al codice ho aggiunto il modo giusto in cui dovrebbe essere utilizzato nella funzione “scanf()”, ovvero “%C”.

-2 Nello “Switch” della funzione “main ()”, I casi sono definti ancora da caratteri (‘B’), per poi essere comparati a numeri.

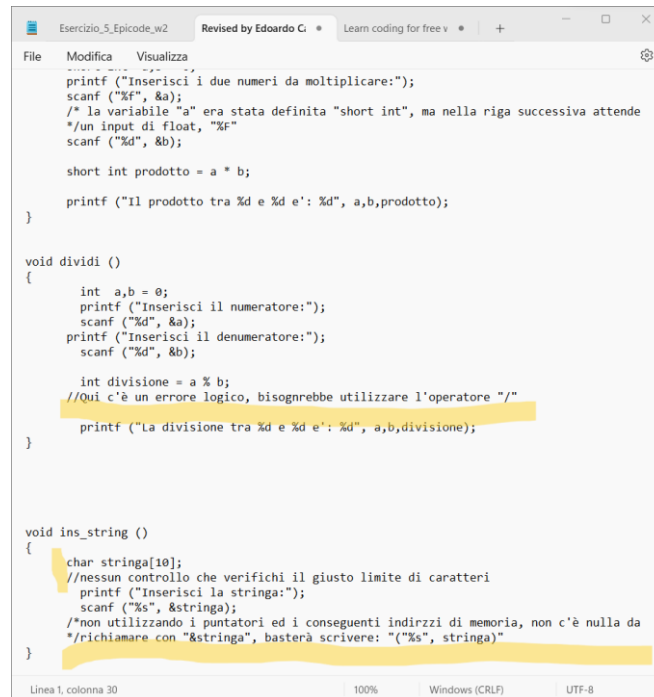
-3 Nella funziona “moltiplica()”, la variabile “a” viene definita come “short integer”, ma nella successiva “funzione scanf()” viene utilizzato “%f” (“float”), invece di “%hd”. Si noti inoltre come il linguaggio Utilizzato in questa fase sembri essere C++, e non C.

Analizzo ora gli errori logici, che trovo nell’ordine:

-1 Nella funzione “dividi()”, viene utilizzato l’operatore “modulo” (%) invece dell’operatore di divisione (/).

-2 Nella funzione “ins_string”, non c’è bisogno di utilizzare “&” poiché nessun puntatore è stato utilizzato in questo codice; la riga corretta è stata redatta dal sottoscritto e presente nel commento allegato.

- 3 Sempre nella stessa funzione, nessun controllo è stato programmato per assicurarsi che l’array “stringa, sia effettivamente di dieci caratteri.



```
File Modifica Visualizza
-----
printf("Inserisci i due numeri da moltiplicare:");
scanf("%f", &a);
/* la variabile "a" era stata definita "short int", ma nella riga successiva attende
 *un input di float, "%f"
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 denominatore:");
    scanf("%d", &b);

    int divisione = a % b;
    //Qui c'è un errore logico, bisognerebbe utilizzare l'operatore "/"
    printf("La divisione tra %d e %d e': %d", a,b,divisione);
}

void ins_string ()
{
    char stringa[10];
    //nessun controllo che verifichi il giusto limite di caratteri
    printf("Inserisci la stringa:");
    scanf("%s", &stringa);
    /*non utilizzando i puntatori ed i conseguenti indirizzi di memoria, non c'è nulla da
    *richiamare con "&stringa", basterà scrivere: ("%s", stringa)"
}

Linea 1, colonna 30 100% Windows (CRLF) UTF-8
```

Per quanto riguarda la logica nel senso più ampio del termine, e di conseguenza la struttura del codice, non può non balzare all’occhio come nessun tipo di “check”, o controllo, venga eseguito dal programma per evitare ulteriori errori. Di seguito si evidenziano i più importanti:

```
Esercizio_5_Epicode_w2    Revised by Edoardo Ci    Learn coding for free v    +    -    □    X
File    Modifica    Visualizza

switch (scelta)
{
    case 'A':
        moltiplica();
        break;
    case 'B':
        dividi();
        break;
    case 'C':
        ins_string();
        break;
    /*i caso sono tutti definiti come caratteri (""), ma verranno poi associati a
    */numeri. Sarebbe il caso di definirli subito tali
}
return 0;
}

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");
    //Nessun controllo che limiti le scelte dell'utente, rischio di errore
}

void moltiplica ()
{
    short int a,b = 0;
    printf ("Inserisci i due numeri da moltiplicare:");
    scanf ("%f", &a);
    /* la variabile "a" era stata definita "short int", ma nella riga successiva attende
    */un input di float, "%f"
}
```

-1 Nella funzione “menu()”, si potrebbe implementare un ciclo “do-while” che garantisca una scelta valida da parte dell’utente.

```
Esercizio_5_Epicode_w2    Revised by Edoardo Ci    Learn coding for free v    +    -    □    X
File    Modifica    Visualizza

    printf ("come posso aiutarti?\n");
    printf ("A >> Moltiplicare due numeri\nB >> Dividere due numeri\nC >> Inserire una
    stringa\n");
    //Nessun controllo che limiti le scelte dell'utente, rischio di errore
}

void moltiplica ()
{
    short int a,b = 0;
    printf ("Inserisci i due numeri da moltiplicare:");
    scanf ("%f", &a);
    /* la variabile "a" era stata definita "short int", ma nella riga successiva attende
    */un input di float, "%f"
    scanf ("%d", &b);
    //Nessun controllo che limiti le scelte dell'utente, rischio di errore
    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 denominatore:");
    scanf ("%d", &b);
    //Nessun controllo che limiti le scelte dell'utente, rischio di errore
    int divisione = a % b;
    //Qui c'è un errore logico, bisognerebbe utilizzare l'operatore "/"

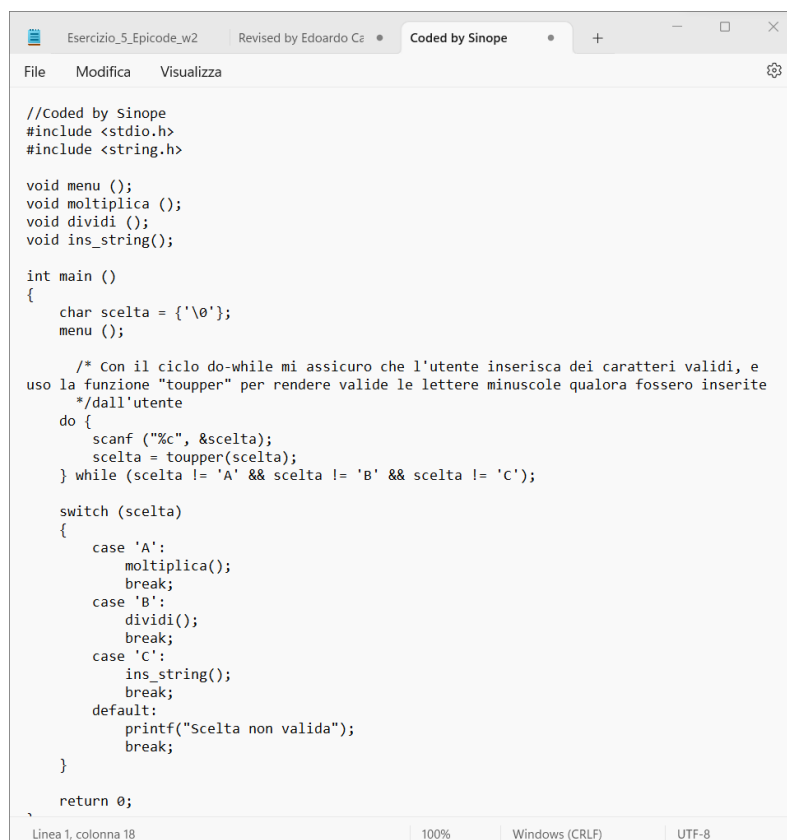
    printf ("La divisione tra %d e %d e': %d", a,b,divisione);
}
```

2 Nella funzione “moltiplica()”, così come in “dividi()”, si potrebbe aggiungere un controllo tramite l’operatore “if”, andando a limitare in questo modo le opzioni a disposizione dell’utente.

Ritenendomi soddisfatto dell’analisi eseguita, inizio a correggere il codice, che allegherò dopo le conclusioni.

Conclusioni:

La poca esperienza con il linguaggio C ha reso particolarmente ostica, all'inizio, l'individuazione degli errori di sintassi, ma la corretta esecuzione del programma lascerebbe pensare che siano stati interamente risolti. I controlli aggiuntivi al menù principale diminuiscono la probabilità di errori in fase di esecuzione o dopo l'input dell'utente, così come quelli aggiunti nelle varie funzioni del programma. La struttura del programma in generale si direbbe più robusta, solida.



```
//Coded by Sinope
#include <stdio.h>
#include <string.h>

void menu ();
void moltiplica ();
void dividi ();
void ins_string();

int main ()
{
    char scelta = {'\0'};
    menu ();

    /* Con il ciclo do-while mi assicuro che l'utente inserisca dei caratteri validi, e
    uso la funzione "toupper" per rendere valide le lettere minuscole qualora fossero inserite
    */dall'utente
    do {
        scanf ("%c", &scelta);
        scelta = toupper(scelta);
    } while (scelta != 'A' && scelta != 'B' && scelta != 'C');

    switch (scelta)
    {
        case 'A':
            moltiplica();
            break;
        case 'B':
            dividi();
            break;
        case 'C':
            ins_string();
            break;
        default:
            printf("Scelta non valida");
            break;
    }

    return 0;
}
```

Linea 1, colonna 18 | 100% | Windows (CRLF) | UTF-8

```
Esercizio_5_Epicode_w2    Revised by Edoardo Ca    Coded by Sinope
File  Modifica  Visualizza

}

return 0;
}

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");
}

void moltiplica ()
{
    short int  a,b = 0;

    printf ("Inserisci i due numeri da moltiplicare:");
    scanf ("%hd", &a);
    scanf ("%hd", &b);

    // Utilizzo "if" per assicurarmi di poter eseguire un check delle variabili
    if (a < 0 || b < 0) {
        printf("Input non valido");
        return;
    }

    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);
}
```

Linea 1, colonna 18 100% Windows (CRLF) UTF-8

```
Esercizio_5_Epicode_w2    Revised by Edoardo Ca    Coded by Sinope
File  Modifica  Visualizza

void dividi ()
{
    int  a,b = 0;

    printf ("Inserisci il numeratore:");
    scanf ("%d", &a);
    printf ("Inserisci il denominatore:");
    scanf ("%d", &b);

    // Controllo anche qui, con if, che gli input inseriti siano validi.
    if (b == 0) {
        printf("Denominatore non valido");
        return;
    }

    int divisione = a / b;

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

void ins_string ()
{
    char stringa[11];
    printf ("Inserisci la stringa:");

    // Usare fgets per ricevere l'input correttamente e evitare errori di buffer. In caso
di errori, improbabili, consultare la documentazione di C
    fgets(stringa, sizeof(stringa), stdin);

    if (stringa[strlen(stringa) - 1] == '\n') {
        stringa[strlen(stringa) - 1] = '\0';
    }

    // Uso if per il check finale
    if (strlen(stringa) > 10) {
        printf("Stringa troppo lunga");
        return;
    }
}
```

Linea 1, colonna 18 100% Windows (CRLF) UTF-8

Esercizio_5_Epicode_w2 Revised by Edoardo C Coded by Sinope

File Modifica Visualizza

```
        printf ("Inserisci il numeratore:");
        scanf ("%d", &a);
        printf ("Inserisci il denominatore:");
        scanf ("%d", &b);

        // Controllo anche qui, con if, che gli input inseriti siano validi.
        if (b == 0) {
            printf("Denominatore non valido");
            return;
        }

        int divisione = a / b;

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

void ins_string ()
{
    char stringa[11];
    printf ("Inserisci la stringa:");

    // Usare fgets per ricevere l'input correttamente e evitare errori di buffer. In
    caso di errori, improbabili, consultare la documentazione di C
    fgets(stringa, sizeof(stringa), stdin);

    if (stringa[strlen(stringa) - 1] == '\n') {
        stringa[strlen(stringa) - 1] = '\0';
    }

    // Uso if per il check finale
    if (strlen(stringa) > 10) {
        printf("Stringa troppo lunga");
        return;
    }

    printf("La string inserita è: %s\n", stringa);
}
```

Linea 1, colonna 18 100% Windows (CRLF) UTF-8