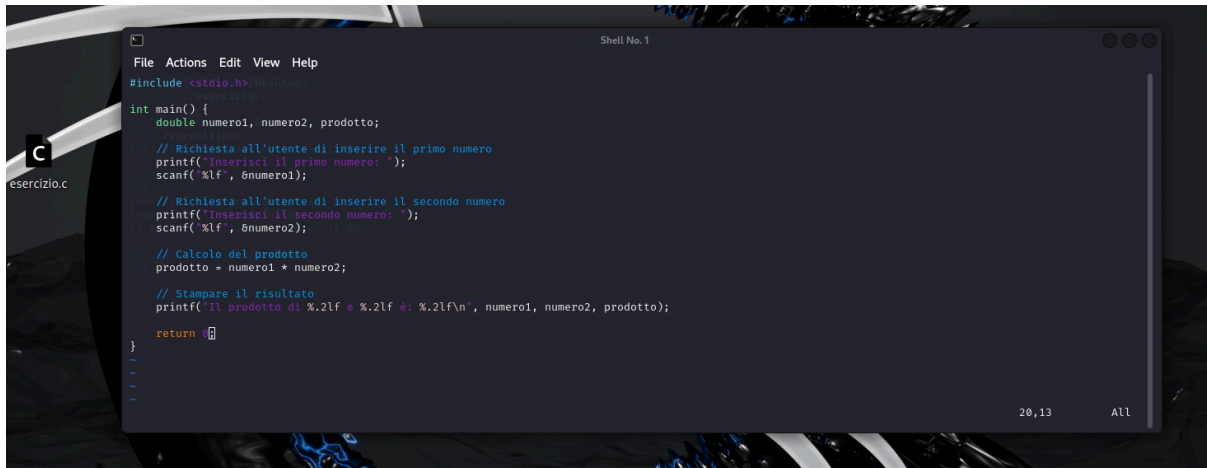


S2.L3 Programmazione in C/Kali linux



```
File Actions Edit View Help
#include <stdio.h>

int main() {
    double numero1, numero2, prodotto;

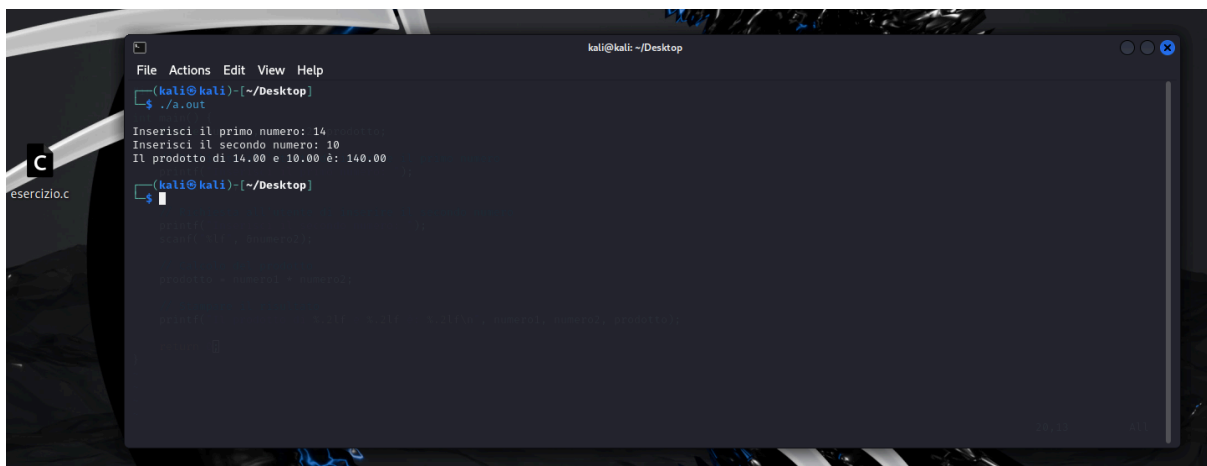
    // Richiesta all'utente di inserire il primo numero
    printf("Inserisci il primo numero: ");
    scanf("%lf", &numero1);

    // Richiesta all'utente di inserire il secondo numero
    printf("Inserisci il secondo numero: ");
    scanf("%lf", &numero2);

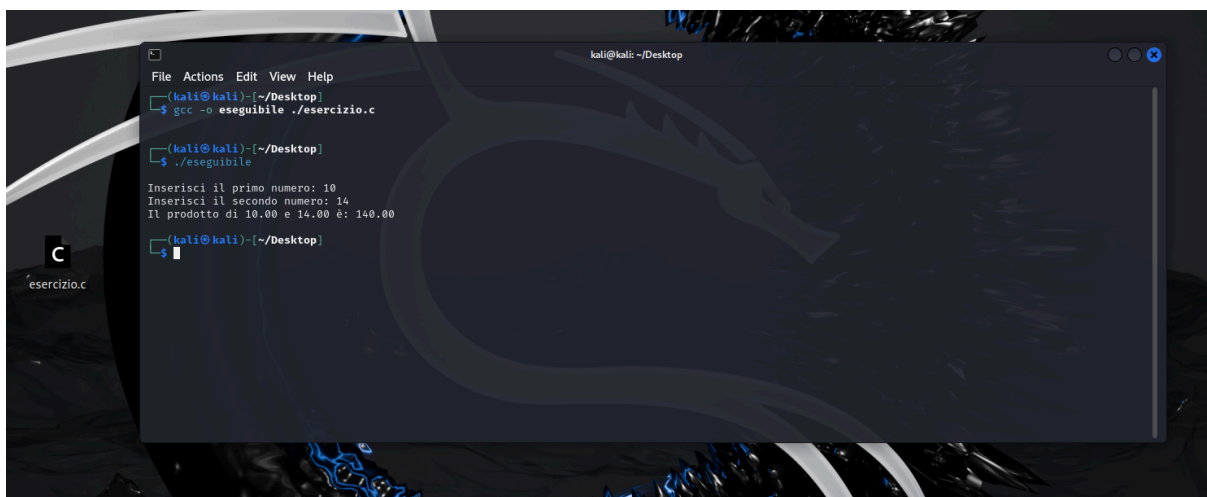
    // Calcolo del prodotto
    prodotto = numero1 * numero2;

    // Stampare il risultato
    printf("Il prodotto di %.2lf e %.2lf è: %.2lf\n", numero1, numero2, prodotto);

    return 0;
}
```



```
kali@kali: ~/Desktop
File Actions Edit View Help
~(kali@kali)-[~/Desktop]
$ ./a.out
Inserisci il primo numero: 14
Inserisci il secondo numero: 10
Il prodotto di 14.00 e 10.00 è: 140.00
~(kali@kali)-[~/Desktop]
$
```



```
kali@kali: ~/Desktop
File Actions Edit View Help
~(kali@kali)-[~/Desktop]
$ gcc -o eseguibile ./esercizio.c
~(kali@kali)-[~/Desktop]
$ ./eseguibile
Inserisci il primo numero: 10
Inserisci il secondo numero: 14
Il prodotto di 10.00 e 14.00 è: 140.00
~(kali@kali)-[~/Desktop]
$
```

Apri l'editor di testo per iniziare a scrivere il codice. assicurandomi di includere la libreria `stdio.h` per poter utilizzare funzioni di input e output. Per questo specifico esercizio, ho scelto di utilizzare variabili di tipo `double` anziché `int` per immagazzinare i numeri inseriti dall'utente. La mia intenzione era dimostrare come i `double` possano gestire numeri con precisione

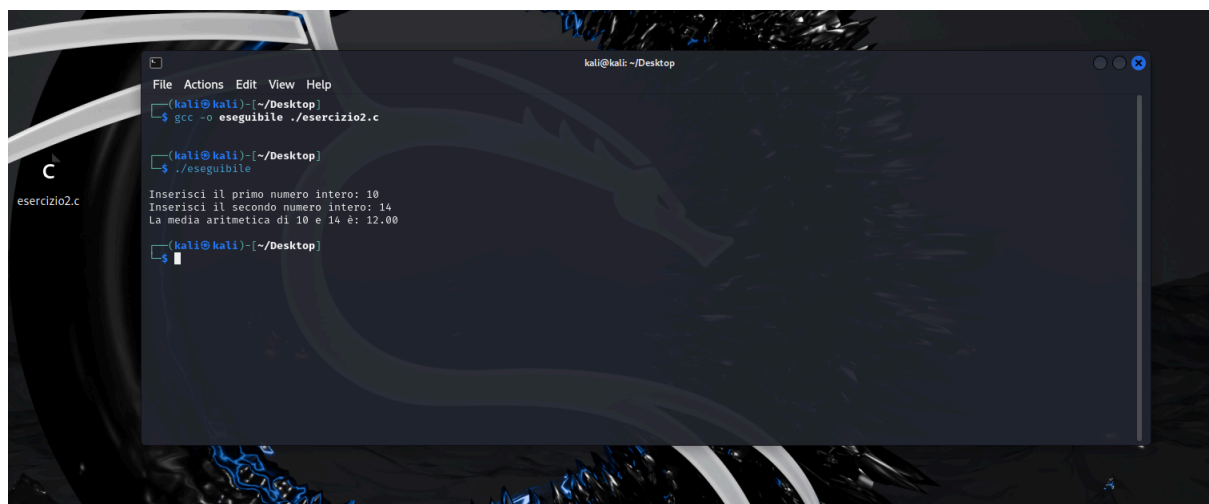
decimale, il che è cruciale quando si lavora con calcoli che possono generare risultati frazionari.

Nel programma, ho implementato la logica per richiedere all'utente di inserire due numeri, utilizzando `printf()` per visualizzare il prompt e `scanf()` per catturare i valori digitati dall'utente. Ho fatto particolare attenzione a usare il specificatore di formato `%lf` con `scanf()`, che è necessario quando si leggono valori di tipo `double`.

Dopo aver acquisito i numeri, ho proceduto al calcolo del prodotto, che ho poi stampato a video, insieme ai numeri iniziali inseriti dall'utente. Mi sono assicurato che l'output del prodotto fosse presentato con una precisione di due decimali, per illustrare la capacità dei `double` di rappresentare numeri con parte decimale.

Una volta soddisfatto del codice che avevo scritto nel file `esercizio.c`, ho utilizzato il terminale per compilare il programma con il comando `gcc -o esercizio ./esercizio.c`. Questo ha prodotto un eseguibile che ho poi lanciato con `./esercizio`. Il programma ha funzionato esattamente come previsto, confermando che l'uso di `double` era la scelta giusta per questo tipo di operazione matematica.

Esercizio 2

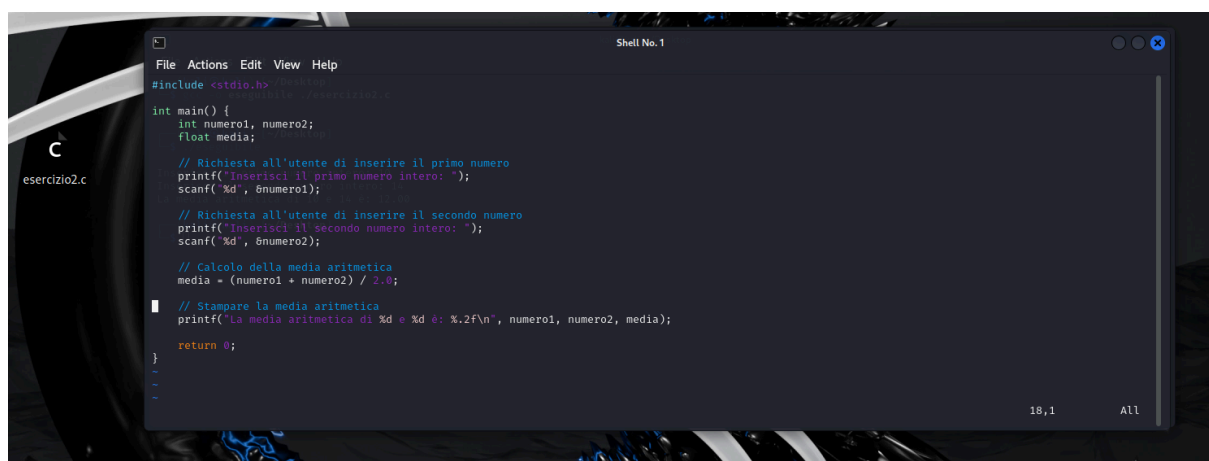


```
kali@kali: ~/Desktop
File Actions Edit View Help
(kali@kali)~/Desktop
$ gcc -o eseguibile ./esercizio2.c

(kali@kali)~/Desktop
$ ./eseguibile

Inserisci il primo numero intero: 10
Inserisci il secondo numero intero: 14
La media aritmetica di 10 e 14 è: 12.00

(kali@kali)~/Desktop
$
```



```
Shell No. 1
File Actions Edit View Help
#include <stdio.h>

int main() {
    int numero1, numero2;
    float media;

    // Richiesta all'utente di inserire il primo numero
    printf("Inserisci il primo numero intero: ");
    scanf("%d", &numero1);

    // Richiesta all'utente di inserire il secondo numero
    printf("Inserisci il secondo numero intero: ");
    scanf("%d", &numero2);

    // Calcolo della media aritmetica
    media = (numero1 + numero2) / 2.0;

    // Stampare la media aritmetica
    printf("La media aritmetica di %d e %d è: %.2f\n", numero1, numero2, media);

    return 0;
}
```

Il programma che ho creato richiede all'utente di inserire due numeri interi, per cui ho definito due variabili int, numero1 e numero2. Per calcolare la media, ho introdotto una variabile float chiamata media, dato che volevo che il risultato fosse visualizzato con una precisione decimale, anche se stavo lavorando con interi.

Dopo aver scritto il codice sorgente in un file denominato esercizio2.c, ho proceduto alla compilazione utilizzando il comando gcc -o eseguibile ./esercizio2.c per generare un file eseguibile. Una volta compilato il programma senza errori, ho eseguito l'eseguibile con il comando ./eseguibile. Il programma ha funzionato come previsto: ha richiesto l'input, ha calcolato la media e l'ha stampata a schermo mostrando "La media aritmetica di 10 e 14 è: 12.00".

. La scelta di usare float per la variabile media era intenzionale: anche se i numeri inseriti erano interi, volevo che il programma potesse gestire numeri reali e mostrare risultati con decimali, qualora gli input fossero stati diversi.