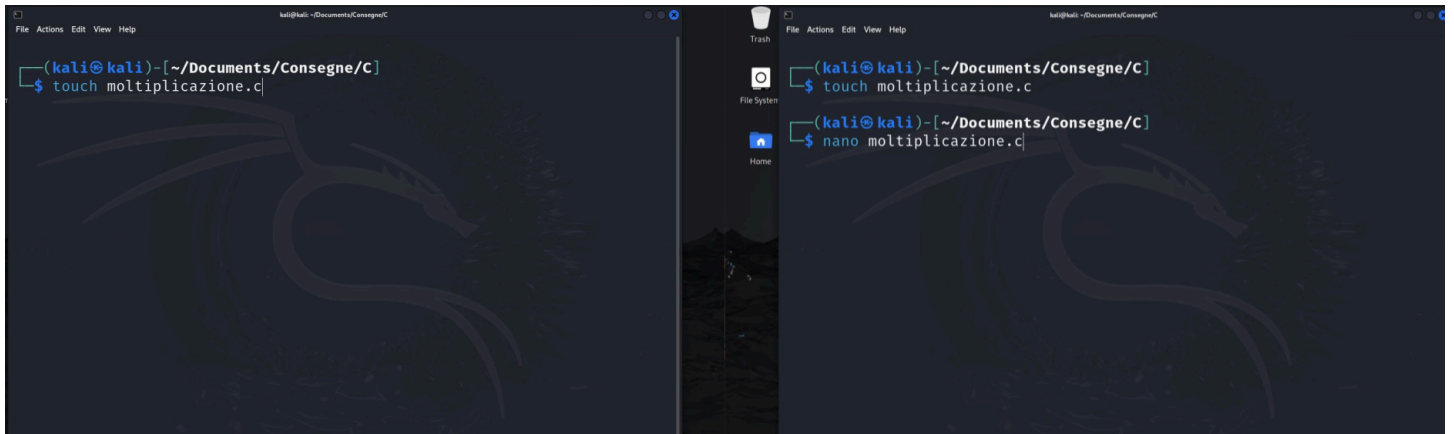


Consegna S2/L2: programmazione in C

Il laboratorio di oggi prevede la realizzazione di due semplici programmi in linguaggio C.

La prima richiesta è stata quella di realizzare un programma che esegua la moltiplicazione tra due numeri inseriti dall'utente.

Ho realizzato il progetto in una macchina virtuale Kali Linux, utilizzando l'editor di testo **nano**. I seguenti passaggi mostrano i comandi che ho inserito da riga di comando per creare e modificare il codice del programma.



Il comando **touch** modifica e aggiorna il tempo di accesso e modifica del file che gli viene passato come parametro. Qualora il file non esistesse, il comando touch lo crea.

Tramite l'editor di testo nano ho creato lo script, importando la libreria stdio che contiene molte delle funzioni per l'input e l'output. Ho poi creato la funzione **main** in cui ho inserito il codice per realizzare la moltiplicazione. Dopo aver dichiarato due variabili di tipo float, ci ho salvato i valori inseriti dall'utente. Ho infine stampato il prodotto tra i due valori con la funzione print.

```
kali@kali: ~/Documents/Consegne/C
File Actions Edit View Help
GNU nano 8.2 moltiplicazione.c *
#include <stdio.h>

int main() {
    float primoNumero, secondoNumero; //Dichiaro due variabili di tipo float

    printf("Inserisci il primo valore: ");
    scanf("%f", &primoNumero); //Salvo il primo valore inserito nella variabile primoNumero

    printf("Inserisci il secondo valore: ");
    scanf("%f", &secondoNumero); //Salvo il secondo valore nella variabile secondoNumero

    printf("%.2f * %.2f = %.2f\n", primoNumero, secondoNumero, primoNumero * secondoNumero);

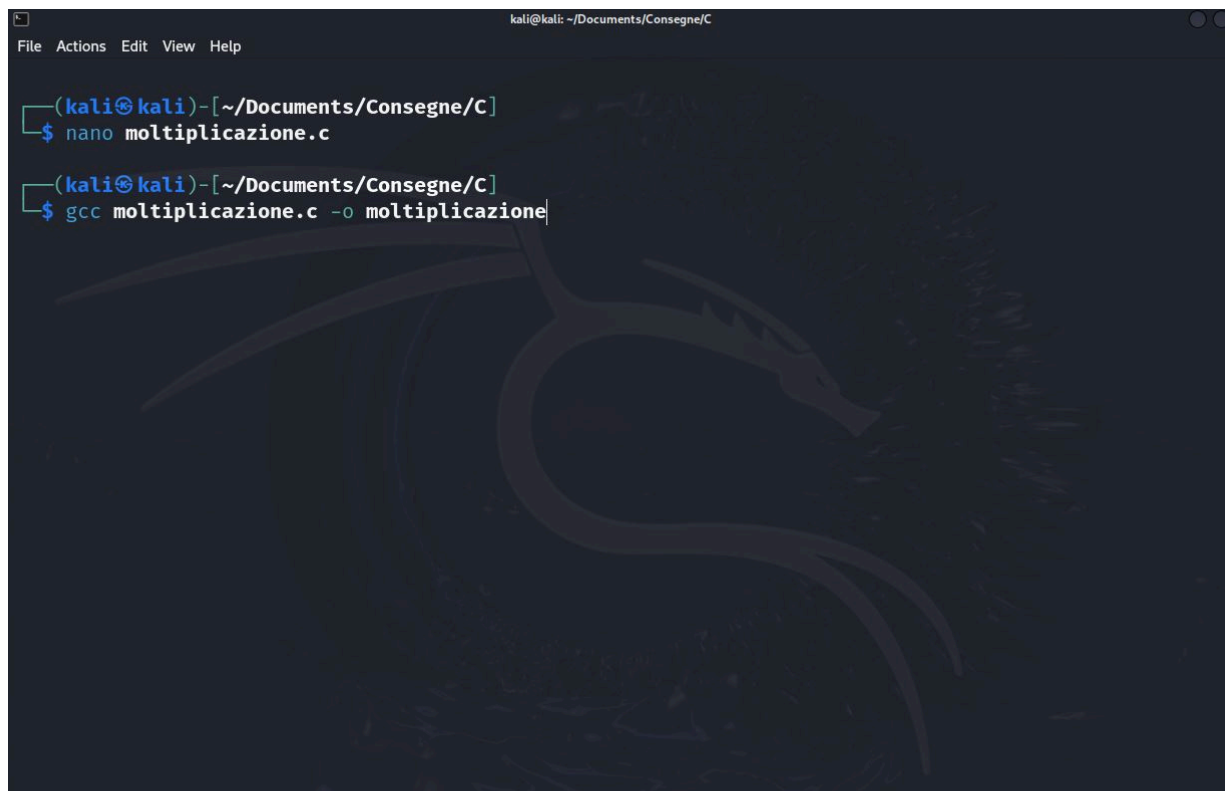
    return 0;
}
```

Help Write Out Where Is Cut Execute Location M-U Undo
Exit Read File Replace Paste Justify Go To Line M-E Redo

Nota: `%.2f` è una variante dello specificatore di formato dei valori float (`%f`) che permette di visualizzare esattamente 2 cifre decimali, completandole a 0 qualora non vi fossero cifre decimali nel numero da stampare.

Ho inserito l'intero 0 come valore di ritorno per indicare che la funzione è stata eseguita con successo.

Ho usato il compilatore **gcc** e rinominato in "moltiplicazione" l'output attraverso l'opzione **-o**.

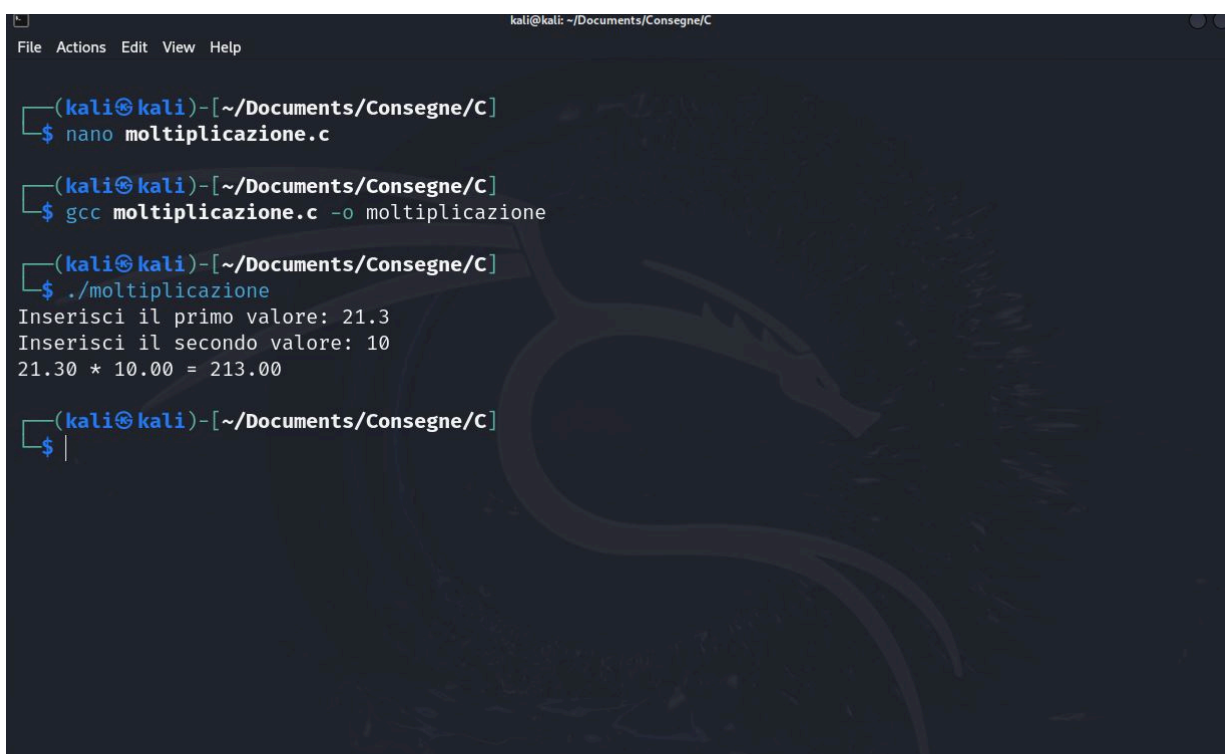


```
kali@kali: ~/Documents/Consegne/C
File Actions Edit View Help

(kali@kali)-[~/Documents/Consegne/C]
$ nano moltiplicazione.c

(kali@kali)-[~/Documents/Consegne/C]
$ gcc moltiplicazione.c -o moltiplicazione
```

Ho infine avviato l'esecuzione del programma tramite il comando da terminale `./moltiplicazione`



```
kali@kali: ~/Documents/Consegne/C
File Actions Edit View Help

(kali@kali)-[~/Documents/Consegne/C]
$ nano moltiplicazione.c

(kali@kali)-[~/Documents/Consegne/C]
$ gcc moltiplicazione.c -o moltiplicazione

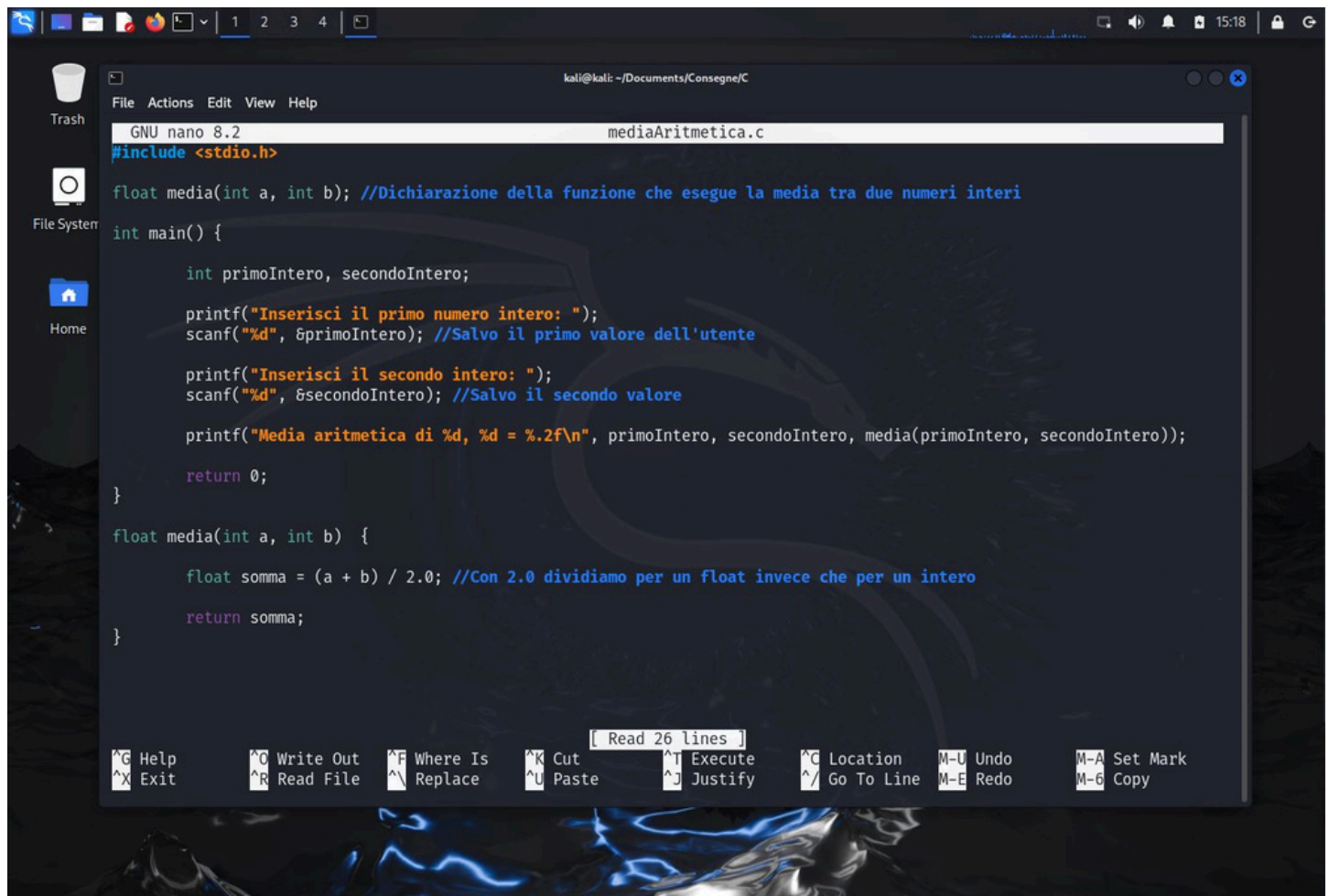
(kali@kali)-[~/Documents/Consegne/C]
$ ./moltiplicazione
Inserisci il primo valore: 21.3
Inserisci il secondo valore: 10
21.30 * 10.00 = 213.00

(kali@kali)-[~/Documents/Consegne/C]
$ |
```

Come da aspettative, il programma chiede all'utente di fornire due numeri e ne calcola il prodotto.

La seconda richiesta di oggi era creare un programma che prendesse in input dall'utente due numeri interi e ne calcolasse la media aritmetica.

Ho creato un nuovo programma, includendo nuovamente la libreria stdio e creando due funzioni: **main** che crea due variabili e immagazzina i valori (interi) inseriti dall'utente, e la funzione **media** che esegue il calcolo della media e ritorna il valore (float) calcolato.



```
GNU nano 8.2 mediaAritmetica.c
#include <stdio.h>

float media(int a, int b); //Dichiarazione della funzione che esegue la media tra due numeri interi

int main() {
    int primoIntero, secondoIntero;

    printf("Inserisci il primo numero intero: ");
    scanf("%d", &primoIntero); //Salvo il primo valore dell'utente

    printf("Inserisci il secondo intero: ");
    scanf("%d", &secondoIntero); //Salvo il secondo valore

    printf("Media aritmetica di %d, %d = %.2f\n", primoIntero, secondoIntero, media(primoIntero, secondoIntero));

    return 0;
}

float media(int a, int b) {
    float somma = (a + b) / 2.0; //Con 2.0 dividiamo per un float invece che per un intero

    return somma;
}
```

La funzione media ritorna un valore float poiché il risultato del calcolo della media potrebbe essere un numero decimale e quindi altrimenti perderemmo parte del risultato. Dividendo i due numeri interi passati in input per un intero (2), otterremo un numero intero a sua volta che viene "castato" a float, perdendo i possibili valori decimali della divisione. Pertanto ho scritto il codice in modo tale da dividere per 2.0 effettuando un cast esplicito.

Poiché il compilatore legge il file una sola volta, partendo dall'inizio del codice, ho seguito le linee guida che consigliano di dichiarare le funzioni prima della funzione main e poi implementarle dopo il corpo della funzione principale.

Ho infine salvato il programma, compilato con gcc e avviato la sua esecuzione.

