Come prima cosa andiamo a creare uno nuova directory Esempio_C, e spostandoci dentro alla directory con il comando nano esempio1.c(il .c serve a creare un file eseguibile in linguaggio C).
Aperto il nostro file andiamo a scrivere al suo interno.

```
┌──(kali㉿kali)-[~]
└─$ mkdir /home/kali/Desktop/Esempio_C

┌──(kali㉿kali)-[~]
└─$ cd /home/kali/Desktop/Esempio_C

┌──(kali㉿kali)-[~/Desktop/Esempio_C]
└─$ nano esempio1.c
```

```
  GNU nano 7.2                          esempio1.c *
#include <stdio.h>

int main()
{

int primo_numero;
int secondo_numero;
int moltiplicazione;

printf ("Inserisci primo numero:\n");
scanf ("%d", &primo_numero);

printf ("Inserisci secondo numero:\n");
scanf ("%d", &secondo_numero);

moltiplicazione = primo_numero * secondo_numero;

printf ("Risultato della moltiplicazione tra i due numeri: %d\n", moltiplicazione);

return 0;
}
```

Questo è il primo programma di moltiplicazione:
1) #include <stdio.h> facciamo riferimento alla libreria che ci interessa
2) int main { definiamo la nostra funzione che vogliamo far svolgere

Andiamo ad inserire le variabili che ci interessano, in questo caso i due numeri da inserire e la moltiplicazione.
Iniziamo a scrivere le operazioni che vogliamo vengano svolte:

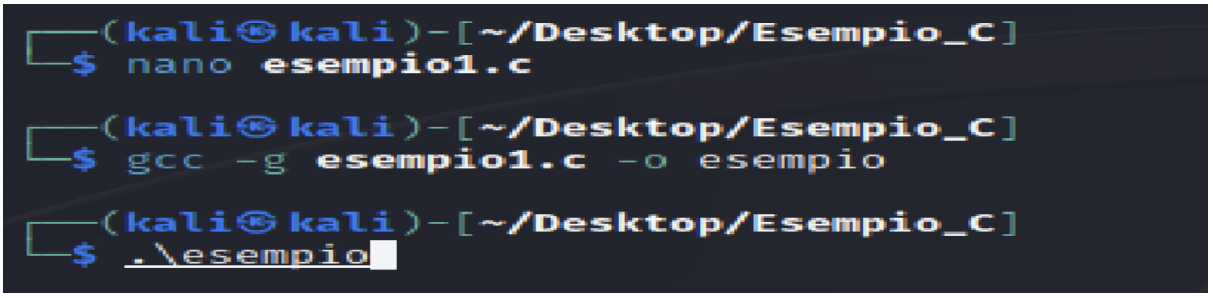
1) printf("...") e scriviamo quello che vogliamo quello che esce a schermo (in questo caso la richiesta del primo numero) e con \n diamo il comando di andare a capo.
2) Per leggere il numero inserito dall'utente bisogna utilizzare scanf (e andiamo a definire con "%d" il tipo di input (INT) , & seguito dal nome della variabile associata all'input.

Eseguiamo la stessa operazione anche per la richiesta della seconda variabile (secondo_numero).

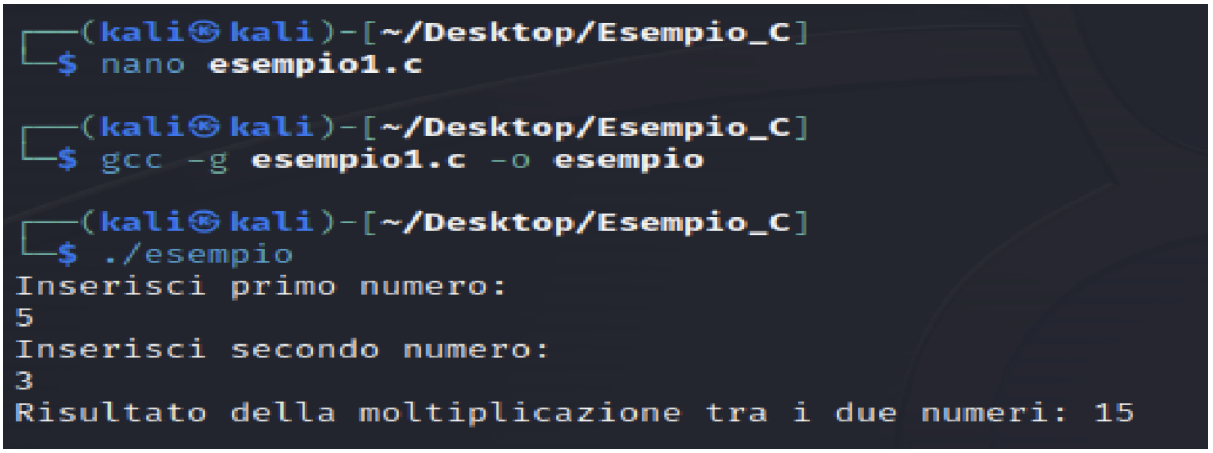
Definiamo cos'è la variabile moltiplicazione cioè la moltiplicazione, per l'appunto, tra primo_numero e secondo_numero.

Dobbiamo mandare a schermo il risultato della moltiplicazione:
1)printf (“...”) all’interno della parentesi scriviamo il testo che vogliamo che appaia a schermo e con %d definiamo che il risultato sia un numero intero , la variabile moltiplicazione.
Chiudiamo la funzione in modo che torni a 0 con returne 0, e chiudiamo la funzione con }.

Ora andiamo a configurare il nostro file in modo che possa essere eseguito con il comando: gcc -g esempio1.c -o esempio1.



Eseguiamo questo programma con ./esempio e verifichiamo che funzioni correttamente.



Ripetiamo gli stessi passaggi per creare il secondo esercizio e lo nominiamo esempio2.c e andiamo ad aprirlo per scriverci sopra.

```
  GNU nano 7.2                                esempio2.c *
#include <stdio.h>

int main()
{

int   primo_numero;
int   secondo_numero;
float   media;

printf ("Inserisci il primo numero:\n");
scanf ("%d", &primo_numero);

printf ("inserisci il secondo numero:\n");
scanf ("%d", &secondo_numero);

media = (float) (primo_numero + secondo_numero)/2;

printf ("Risultato della media: %2f\n", media);

return 0;

}
```

Qui ci viene richiesta la media: procediamo come prima inserendo le variabili e ricordandoci di definire la media come float, perchè la media potrebbe essere un numero reale.

Come nel primo esempio andiamo a dare i comandi per mandare a schermo richiesta e poi inserimento dei due valori e andiamo a definire la variabile media = scriviamo l'operazione di media scrivendoci prima (float), cioè andiamo a dire che il risultato dell'operazione di media dovrà essere un numero reale.

Ora scriviamo printf("...") il testo che ci interessi vada a schermo : %2f, cioè che il risultato sarà un numero reale(float) con 2 valori dopo la virgola (se ne avessimo voluti 5 avremmo scritto %5f).

Come prima chiudiamo la fuzione con returne 0 }.

Salviamo torniamo a terminale ripetiamo i passaggi dell'esempio 1 per eseguire il file (gcc -g esempio2.c -o esempio2), e verifichiamo che il programma funzioni correttamente.

```
┌──(kali㉿kali)-[~/Desktop/Esempio_C]
└─$ nano esempio2.c

┌──(kali㉿kali)-[~/Desktop/Esempio_C]
└─$ gcc -g esempio2.c -o esempio2

┌──(kali㉿kali)-[~/Desktop/Esempio_C]
└─$ ./esempio2
Inserisci il primo numero:
9
inserisci il secondo numero:
2
Risultato della media: 5.500000
```

PS: in linguaggio C ogni riga di codice deve terminare con ;