

Prima di iniziare a realizzare i due programmi in C richiesti dalla consegna, è necessario creare una cartella e al suo interno un file .c in cui poter svolgere l'esercizio. Ho creato la cartella "esempio" sul Desktop e al suo interno il file "esempio.c" in questo modo:

```
(kali@kali)-[~]
$ mkdir /home/kali/Desktop/esempio

(kali@kali)-[~]
$ cd /home/kali/Desktop/esempio

(kali@kali)-[~/Desktop/esempio]
$ nano esempio.c

(kali@kali)-[~/Desktop/esempio]
$ ls
esempio.c
```

Grazie al comando nano esempio.c si apre il file in cui scrivere il primo programma.

1 – Si scriva un programma che esegua l'operazione moltiplicazione tra due numeri inseriti dall'utente.

La libreria utilizzata è stdio.h che fornisce funzioni per l'input e l'output standard.
Funzione principale = int main ()

All'interno delle parentesi graffe { } viene inserito il codice del programma.

Successivamente bisogna:

- Chiedere all'utente di inserire il primo numero
- Chiedere all'utente di inserire il secondo numero
- Calcolare il prodotto dei due numeri
- Stampare

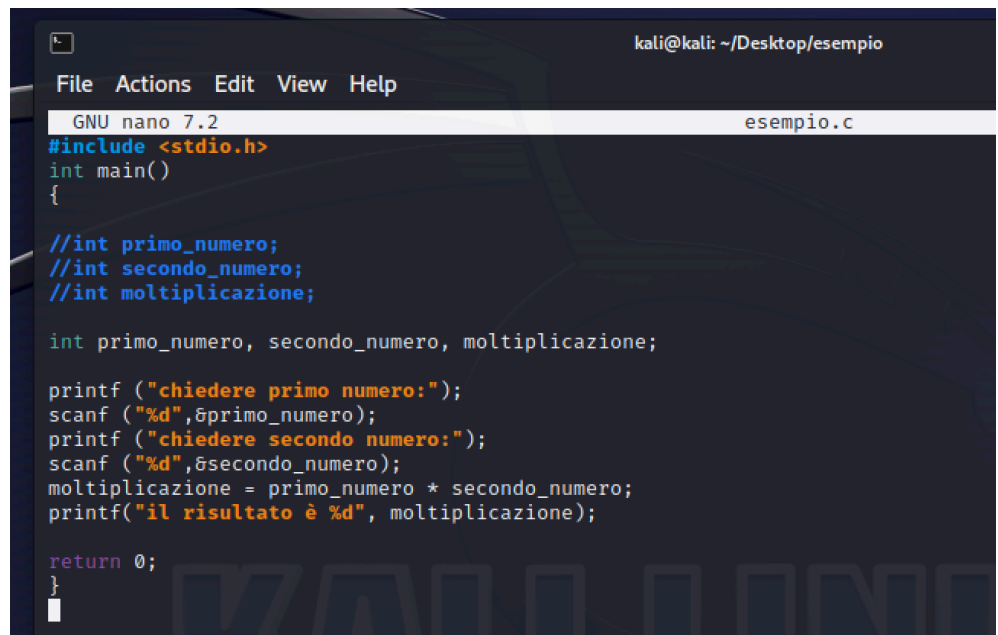
Quindi:

- Dichiarare le variabili 1 e 2 per memorizzare i numeri inseriti dall'utente e il prodotto per memorizzare il risultato della moltiplicazione
- Stampare un messaggio per informare l'utente sul calcolo che il programma eseguirà
- Chiedere all'utente di inserire il primo numero che verrà salvato nella variabile 1
- Chiedere all'utente di inserire il secondo numero che verrà salvato nella variabile 2
- Calcolare il prodotto dei due numeri utilizzando l'operazione di moltiplicazione (*) e memorizzare il risultato della variabile prodotto
- Stampare il risultato della moltiplicazione a schermo quindi il programma terminerà restituendo zero per indicare che il programma si sia chiuso correttamente (0).

scanf: dà la possibilità all'utente di digitare qualcosa, sospendendo il programma in attesa di un input

%d: indica che verrà preso in input un numero intero che verrà poi assegnato alla variabile scritta dopo il puntatore &

&: indica appunto in che variabile andrà l'input



```
kali@kali: ~/Desktop/esempio
File Actions Edit View Help
GNU nano 7.2 esempio.c
#include <stdio.h>
int main()
{
    //int primo_numero;
    //int secondo_numero;
    //int moltiplicazione;

    int primo_numero, secondo_numero, moltiplicazione;

    printf ("chiedere primo numero:");
    scanf ("%d",&primo_numero);
    printf ("chiedere secondo numero:");
    scanf ("%d",&secondo_numero);
    moltiplicazione = primo_numero * secondo_numero;
    printf("il risultato è %d", moltiplicazione);

    return 0;
}
```

gcc: è il compilatore

-o: serve per specificare il nome del file

Con il comando `./programma1` rendiamo quindi il primo programma eseguibile

```
(kali㉿kali)-[~/Desktop/esempio]
$ ls
esempio.c

(kali㉿kali)-[~/Desktop/esempio]
$ gcc -o programma1 esempio.c

(kali㉿kali)-[~/Desktop/esempio]
$ ./programma1
chiedere primo numero:4
chiedere secondo numero:3
il risultato è 12

(kali㉿kali)-[~/Desktop/esempio]
$
```

2- Si scriva un programma in linguaggio C che legga due valori interi e visualizzi la loro media aritmetica.

-L'inizio del programma è come nel programma precedente con la differenza che la media non restituirà un numero intero

- Dichiarare le variabili per i numeri inseriti e per la media
- Calcolare la media di due numeri
- Chiedere all'utente di inserire il primo numero e il secondo numero
- Calcolare la media dei due numeri e stampare il risultato
- Return zero

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

int main()
{
    int primo_numero, secondo_numero;
    float media;

    printf("chiedere primo numero:");
    scanf("%d",&primo_numero);
    printf("chiedere secondo numero:");
    scanf("%d",&secondo_numero);
    media=(float)(primo_numero + secondo_numero) /2;
    printf("la media è:%.2f",media);

    return 0;
}
```

%.2f: indica che prenderà in input un numero reale (in questo caso per due valori dopo la virgola)

Avendo un risultato non intero è necessario fare un'operazione di casting: la conversione esplicita di un tipo di dato in un altro tipo di dato. In questo caso abbiamo "castato" float in media.

Anche in questo caso rendo il programma eseguibile in questo modo:

```
File Actions Edit View Help
(kali㉿kali)-[~/Desktop/esempio]
$ ls
esempio.c  esempio2.c  programma1

(kali㉿kali)-[~/Desktop/esempio]
$ gcc -o programma2 esempio2.c

(kali㉿kali)-[~/Desktop/esempio]
$ ls
esempio.c  esempio2.c  programma1  programma2

(kali㉿kali)-[~/Desktop/esempio]
$ ./programma2
chiedere primo numero:30
chiedere secondo numero:27
la media è:28.50

(kali㉿kali)-[~/Desktop/esempio]
$
```