

W^oG

Linguaggio.c

Esercizi e Soluzioni



INDICE

- **Esercizi di Programmazione Base**

1. Stampa Hello World	1
2. Calcolo Somma di Due Numeri	2
3. Programma per Calcolare il Numero Precedente e Successivo	3
4. Verifica se un Numero è Pari o Dispari	4
5. Somma di 10 Numeri Interi	5
6. Calcolo Della Somma di N Numeri	6
7. Calcolo del Fattoriale di un Numero	7
8. Verifica se un Numero è Primo	8
9. Calcolo della Successione di Fibonacci	9
10. Risoluzione del Quadrato di Binomio	10
11. Risoluzione di un'Equazione di Secondo Grado	11
12. Calcolo delle Coordinate del Punto Medio di un Segmento	12
13. Calcolo del Tasso di Interesse su 25 Giorni	13
14. Calcolo dello Scorporo dell'IVA	14
15. Inserimento e Stampa di 10 Numeri in un Vettore	15
16. Stampa dei Numeri Pari in un Vettore	16
17. Conteggio di Numeri Pari e Dispari in un Vettore	17
18. Somma dei Numeri Pari e Dispari in un Vettore	18
19. Calcolo della Somma dei Numeri Pari in un Vettore	19
20. Calcolo della Media dei Numeri Pari in un Vettore	20
21. Calcolo Somma dei Prodotti tra due Vettori	21
22. Unione Alternata di due Vettori in un Terzo Vettore	22
23. Contatore Maggiorenni e Minorenni	23
24. Programma per Trovare il Numero Maggiore tra due Interi	24
25. Verifica della Validità di una Classe Scolastica	25-26
26. Identificazione del Tipo di Carattere	27
27. Verifica della Parità dei Numeri in un Vettore	28
28. Conteggio dei Numeri Pari e Dispari in un Vettore	29
29. Somma dei Numeri Positivi e Conteggio dei Negativi	30
30. Determinare il Mese Corrispondente a un Numero	31-32
31. Riconoscimento del Tipo di Carattere Inserito	33-34
32. Verifica di Caratteri Speciali in un Nome o Cognome	35

INDICE

- **Algoritmi di Ordinamento di Vettori**

1.Selection Sort.....	36
2.Bubble Sort	37
3.Merge Sort	38/39

- **Varianti di Esercizi con Funzioni**

1.Maggiore tra Due Numeri – Calcolo con Funzione	40
2.Numero Precedente e Successivo – Calcolo con Funzione	41
3.Calcolo della Somma tra Due Numeri – Calcolo con Funzione	42
4.Calcolo della Somma di Dieci Numeri – Calcolo con Funzione	43
5.Successione di Fibonacci – Calcolo con Funzione	44
6.Riconoscimento Carattere- Calcolo con Funzione	45/46

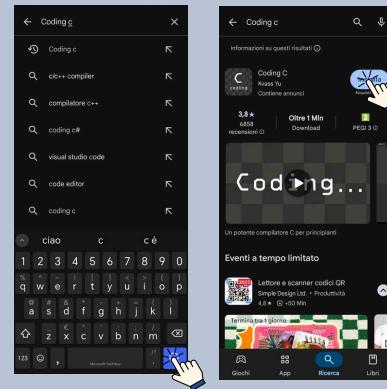
Installazione dell'ambiente di sviluppo

Per scrivere ed eseguire programmi in linguaggio C, è necessario un ambiente di sviluppo con un compilatore. Qui sotto trovi le istruzioni per diversi dispositivi: Android, Windows, Mac e anche un comodo compilatore online.



Android – Coding C

1. Apri il Play Store.
2. Cerca Coding C: C Compiler IDE.
3. Installa l'app.
4. Avvia la app e inizia a scrivere il tuo codice C.
5. Premi il tasto ► per eseguire il programma.



Windows – Dev-C++

1. Vai su:
2. [🔗 https://sourceforge.net/projects/orwelldevcpp/](https://sourceforge.net/projects/orwelldevcpp/)
3. Scarica Dev-C++ e segui l'installazione guidata.
4. Avvia il programma e vai su File > Nuovo > File sorgente.
5. Scrivi il tuo codice C e salvalo con estensione .c.
6. Premi F9 per compilare ed eseguire il programma.



Mac – Xcode

1. Apri l'App Store e cerca Xcode.
2. Installa Xcode (attenzione: occupa circa 10 GB).
3. Apri il Terminale e digita:

```
xcode-select --install
```

per installare gli strumenti da riga di comando.

4. Avvia Xcode.
5. Crea un nuovo progetto:
 - Seleziona "Command Line Tool".
 - Dai un nome al progetto.
 - Scegli C come linguaggio.



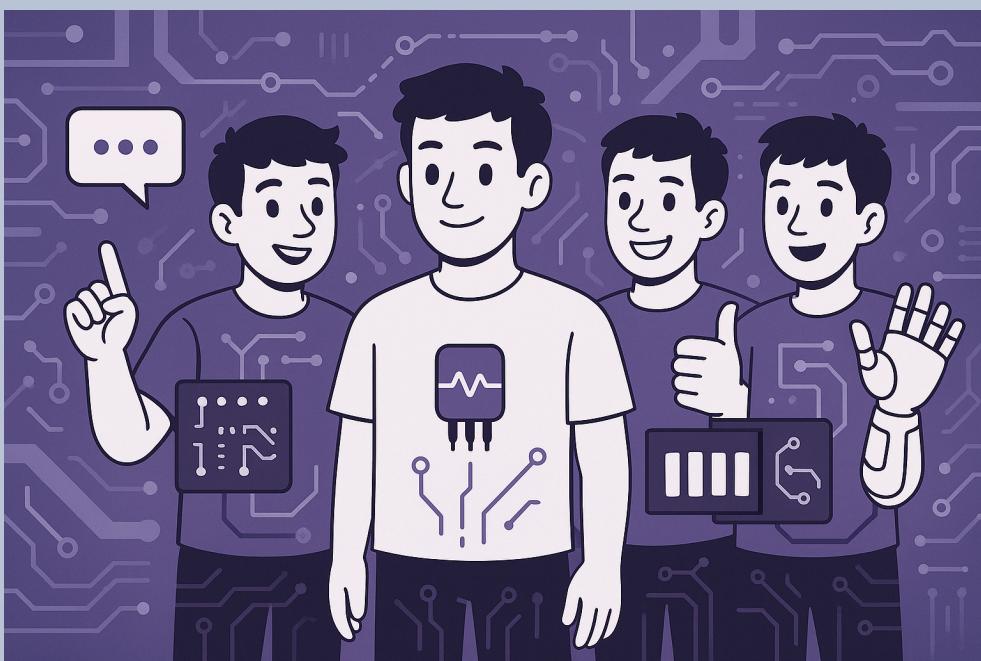
Compilatore Online – OnlineGDB

1. Vai su:
2. [🔗 https://www.onlinegdb.com/online_c_compiler](https://www.onlinegdb.com/online_c_compiler)
3. Seleziona "C" dal menu in alto a sinistra (di solito è già selezionato).

Esercizi di Programmazione Base

Questa raccolta di esercizi è pensata per aiutarti a sviluppare e consolidare le tue competenze nella programmazione in C. Attraverso una serie di problemi di difficoltà crescente, potrai mettere in pratica concetti fondamentali come l'uso di variabili, strutture di controllo, cicli, funzioni e vettori.

Gli esercizi proposti coprono una vasta gamma di argomenti, partendo dalle basi, come la stampa a schermo di un messaggio, fino ad arrivare a calcoli più complessi, come la risoluzione di equazioni e la gestione di strutture dati. Affronta ogni esercizio con curiosità e determinazione, sfruttando questa raccolta come un'opportunità per migliorare le tue capacità di programmazione.



Stampa Hello World

Questo programma in C stampa a video il messaggio "Hello World" utilizzando la funzione **printf()**. È un esempio base per introdurre la sintassi del linguaggio C.

```
/*
 * programma che stampa a video Hello World.
 *
 */
#include <stdio.h>
    int main ()  {

        printf("Hello World");

        return 0 ;
    }
```

Calcolo Somma di Due Numeri

Questo programma in C chiede all'utente di inserire due numeri interi, calcola la loro somma e visualizza il risultato. Utilizza la funzione **scanf** per leggere l'input e **printf** per stampare la somma.

```
/*
*stampare la somma di 2 numeri
*/
#include <stdio.h>
int main () {
    int num1,num2,somma;
    printf("inserisci il primo numero");
    scanf("%d",&num1);
    printf("inserisci il secondo numero");
    scanf("%d",&num2);
    somma=num1+num2;
    printf("la somma dei due numeri e' %d\n",somma);
}
```

Programma per Calcolare il Numero Precedente e Successivo

Questo programma in C permette all'utente di inserire un numero intero e restituisce in output sia il numero precedente che quello successivo. Utilizza la funzione **scanf** per acquisire l'input e **printf** per visualizzare i risultati.



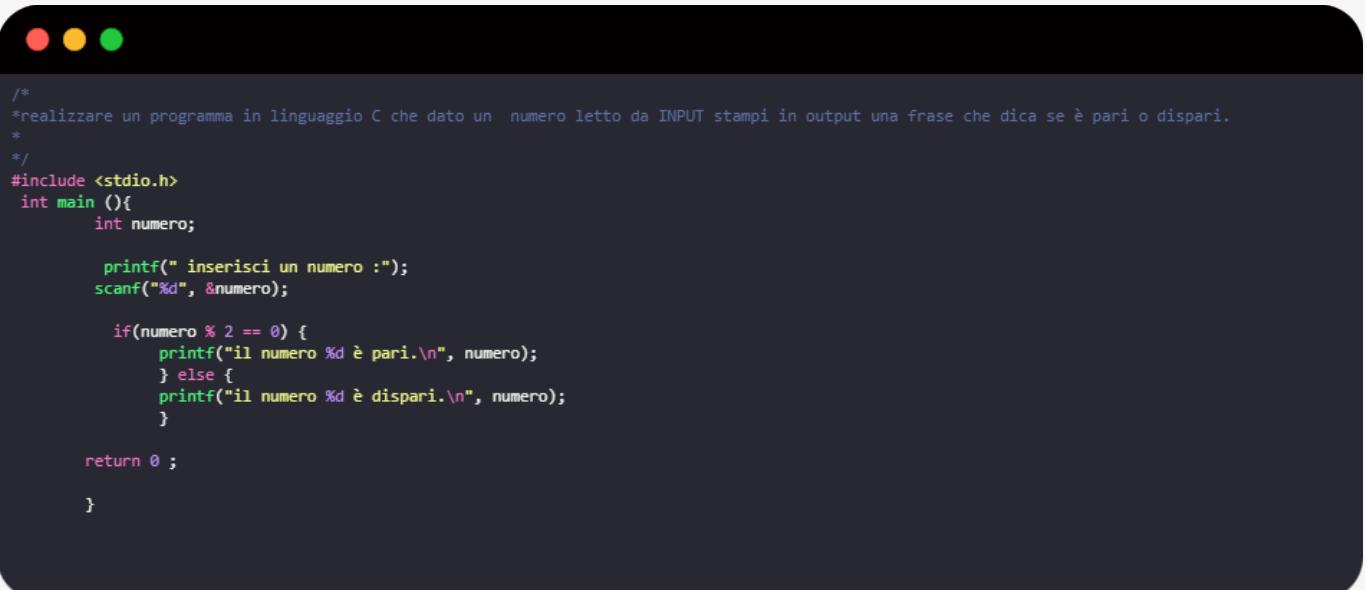
```
/*
*inserire un numero e stampare in output il numero precedente e quello successivo.
*/
#include<stdio.h>
main () {
    int numero;
    int precedente;          // variabile del numero precedente
    int successivo;         // variabile del numero successivo

        // input
    printf("inserisci il numero: ");
    scanf("%d",&numero);

        // calcolo numeri
    precedente=numero-1;
    successivo=numero+1;
        // output
    printf("il numero precedente e':%d\n",precedente);
    printf("il numero successivo e':%d",successivo);
}
```

Verifica se un Numero è Pari o Dispari

Questo programma in C legge un numero intero inserito dall'utente e verifica se è pari o dispari. Il controllo viene fatto utilizzando l'operatore modulo “%” e il risultato viene stampato a schermo.



```
/*
*realizzare un programma in linguaggio C che dato un numero letto da INPUT stampi in output una frase che dica se è pari o dispari.
*/
#include <stdio.h>
int main (){
    int numero;

    printf(" inserisci un numero :");
    scanf("%d", &numero);

    if(numero % 2 == 0) {
        printf("il numero %d è pari.\n", numero);
    } else {
        printf("il numero %d è dispari.\n", numero);
    }

    return 0 ;
}
```

Somma di 10 Numeri Interi

Questo programma in C legge 10 numeri interi inseriti dall'utente e ne calcola la somma utilizzando un ciclo for. Alla fine, stampa il risultato della somma.

```
/*
*scrivere un programma che legga 10 numeri interi num e ne calcola la somma tramite un ciclo for
*/
#include <stdio.h>
int main(){

int somma = 0 ;
int num,i ;

for ( i = 0 ; i < 10; i++){
    printf ("Inserisci un numero intero:");
    scanf ("%d",&num);
    somma = somma + num;
}

printf("La somma dei numeri è %d.", somma );
return 0 ;
}
```

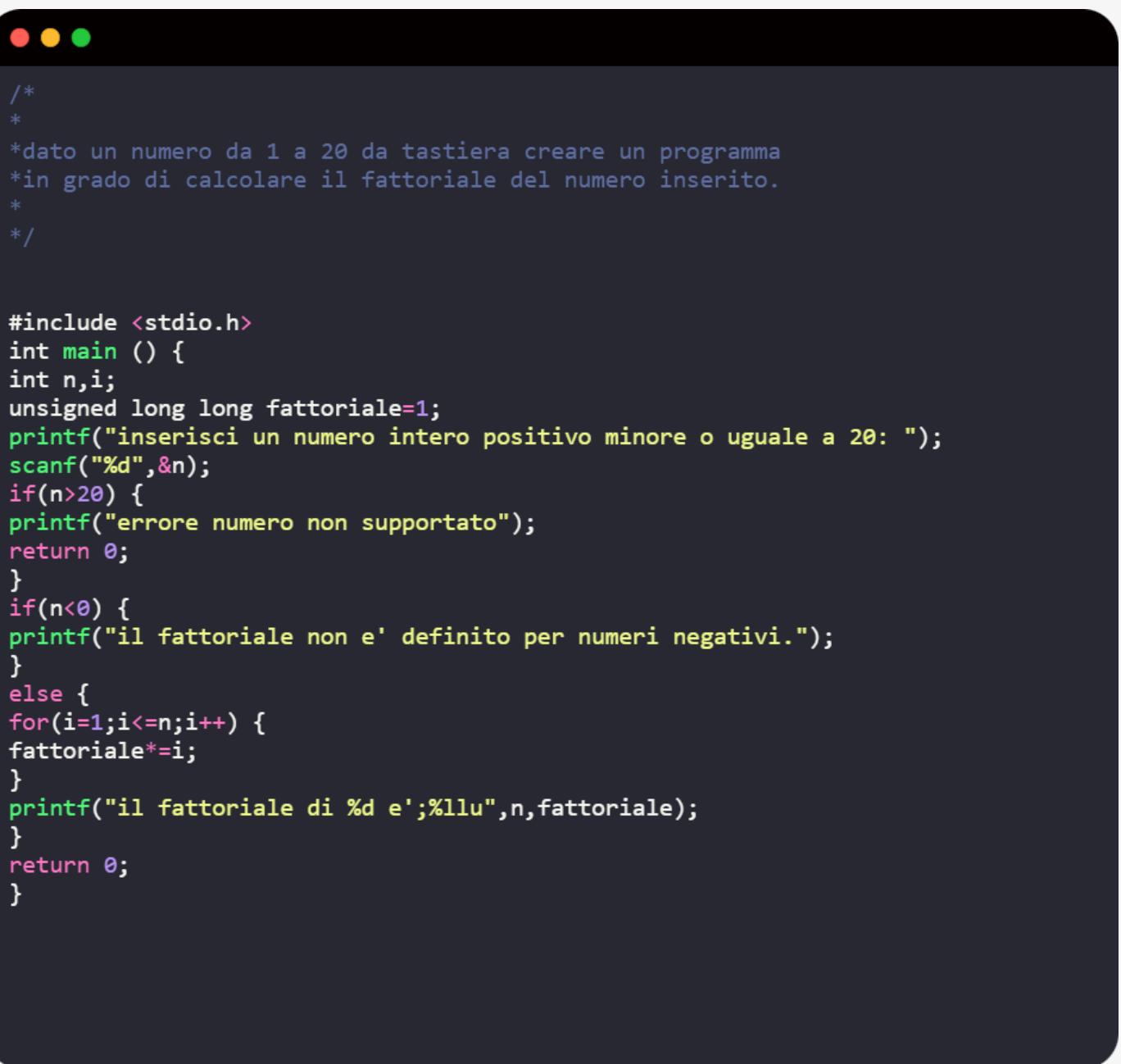
Calcolo Della Somma di N Numeri

Questo programma in C consente all'utente di sommare un numero arbitrario di interi. Dopo aver chiesto quanti numeri si desidera sommare, il programma utilizza un ciclo **for** per acquisire i numeri dall'utente e calcolare la loro somma. Infine, stampa il risultato della somma totale.

```
/*
*somma di n numeri
*
*/
#include <stdio.h>
int main () {
int n,i;
int somma=0;
int num;
printf("quanti numeri vuoi sommare?");
scanf("%d",&n);
for(i=1;i<=n;i++) {
printf("inserisci il %d numero:",i);
scanf("%d",&num);
somma+=num;
}
printf("la somma dei %d numeri è %d\n",n,somma);
return 0;
}
```

Calcolo del Fattoriale di un Numero

Questo programma in C calcola il fattoriale di un numero intero positivo compreso tra 1 e 20, inserito dall'utente. Il programma utilizza un ciclo **for** per moltiplicare progressivamente i numeri fino al valore inserito. Se l'utente inserisce un numero maggiore di 20, viene visualizzato un messaggio di errore, poiché il risultato potrebbe superare i limiti della variabile. Inoltre, viene gestito il caso di numeri negativi, per i quali il fattoriale non è definito.



```
/*
*
*dato un numero da 1 a 20 da tastiera creare un programma
*in grado di calcolare il fattoriale del numero inserito.
*/
#include <stdio.h>
int main () {
int n,i;
unsigned long long fattoriale=1;
printf("inserisci un numero intero positivo minore o uguale a 20: ");
scanf("%d",&n);
if(n>20) {
printf("errore numero non supportato");
return 0;
}
if(n<0) {
printf("il fattoriale non e' definito per numeri negativi.");
}
else {
for(i=1;i<=n;i++) {
fattoriale*=i;
}
printf("il fattoriale di %d e';%llu",n,fattoriale);
}
return 0;
}
```

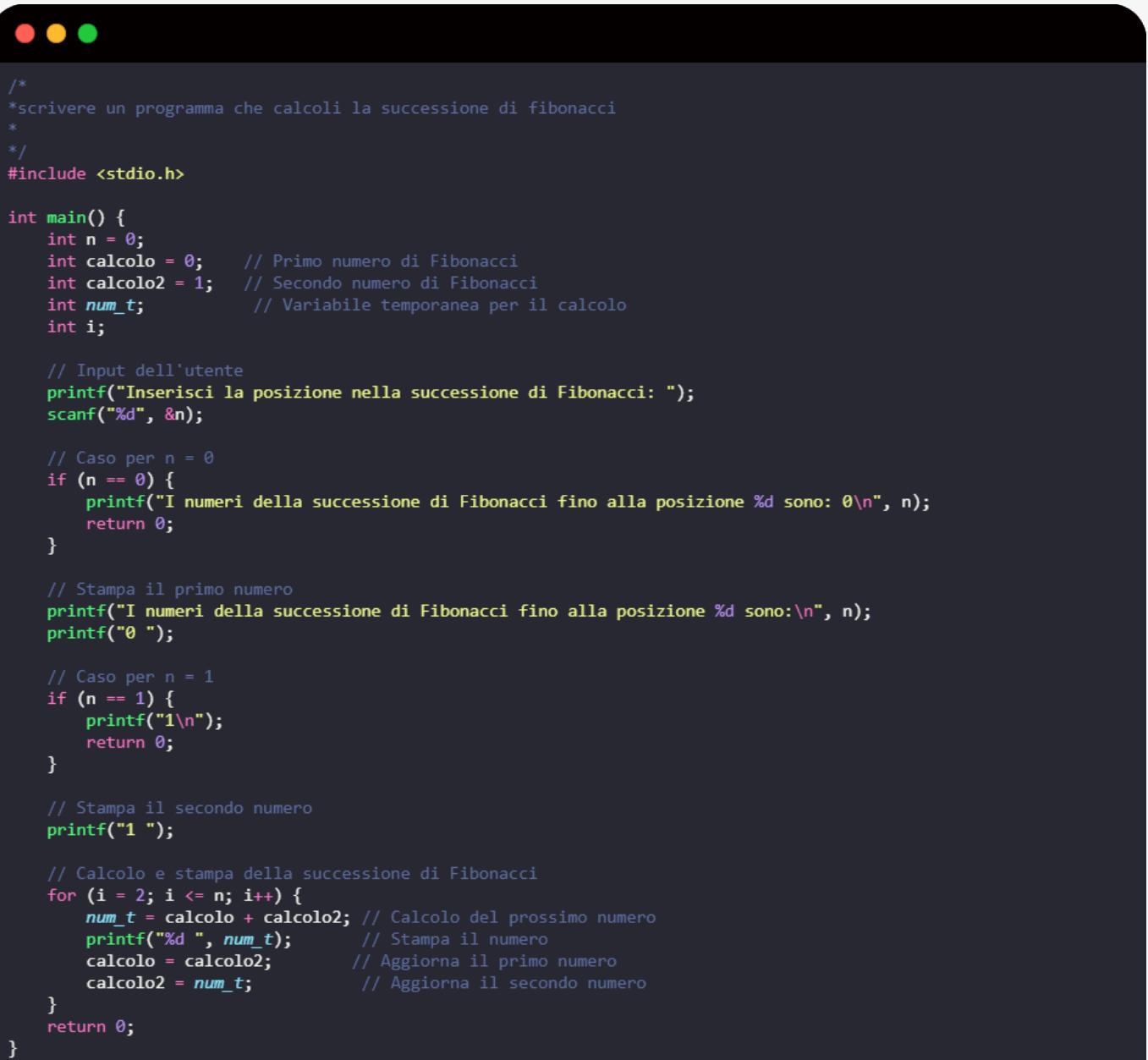
Verifica se un Numero è Primo

Questo programma in C verifica se un numero intero positivo inserito dall'utente è un numero primo. Dopo aver acquisito l'input, il programma utilizza un ciclo **for** per controllare se il numero ha divisori oltre a 1 e a se stesso. Se trova un divisore, il numero non è primo; altrimenti, è primo. Infine, stampa il risultato.

```
/*
*scrivere un programma che calcoli la verifica di un numero primo
*/
#include <stdio.h>
main () {
int numero,i,primo=1;
printf("inserisci un numero intero positivo:");
scanf("%d",&numero);
if(numero<=1) {
primo=0;
}
else {
for(i=2;i<=numero/2;i++) {
if(numero% i==0) {
primo=0;
break;
}
}
}
if(primo) {
printf("il numero %d e' primo.\n",numero);
}
else{
printf("il numero %d non e' primo.\n",numero);
}
return 0;
}
```

Calcolo della Successione di Fibonacci

Questo programma in C genera e stampa i numeri della successione di Fibonacci fino a una posizione specificata dall'utente. Dopo aver acquisito l'input, il programma gestisce i casi base (**n = 0** e **n = 1**) e utilizza un ciclo **for** per calcolare i successivi numeri della sequenza. Ogni numero viene ottenuto come somma dei due precedenti e viene stampato a schermo.



```
/*
 *scrivere un programma che calcoli la successione di fibonacci
 */
#include <stdio.h>

int main() {
    int n = 0;
    int calcolo = 0;      // Primo numero di Fibonacci
    int calcolo2 = 1;     // Secondo numero di Fibonacci
    int num_t;           // Variabile temporanea per il calcolo
    int i;

    // Input dell'utente
    printf("Inserisci la posizione nella successione di Fibonacci: ");
    scanf("%d", &n);

    // Caso per n = 0
    if (n == 0) {
        printf("I numeri della successione di Fibonacci fino alla posizione %d sono: 0\n", n);
        return 0;
    }

    // Stampa il primo numero
    printf("I numeri della successione di Fibonacci fino alla posizione %d sono:\n", n);
    printf("%d ", 0);

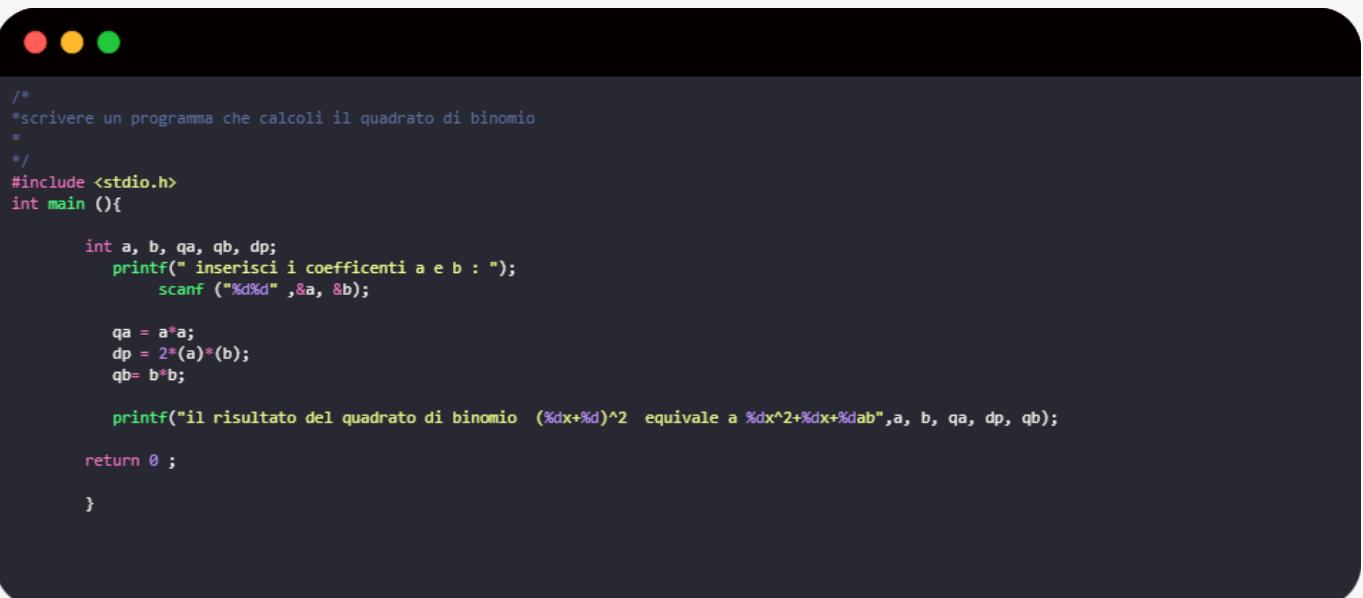
    // Caso per n = 1
    if (n == 1) {
        printf("1\n");
        return 0;
    }

    // Stampa il secondo numero
    printf("1 ");

    // Calcolo e stampa della successione di Fibonacci
    for (i = 2; i <= n; i++) {
        num_t = calcolo + calcolo2; // Calcolo del prossimo numero
        printf("%d ", num_t);       // Stampa il numero
        calcolo = calcolo2;        // Aggiorna il primo numero
        calcolo2 = num_t;          // Aggiorna il secondo numero
    }
    return 0;
}
```

Risoluzione del Quadrato di Binomio

Questo programma in C calcola il quadrato di un binomio della forma $(ax + b)^2$. Chiede all'utente di inserire i coefficienti a e b, e quindi calcola e stampa l'espansione del quadrato del binomio, utilizzando la formula $(a + b)^2 = a^2 + 2ab + b^2$.



```
/*
*scrivere un programma che calcoli il quadrato di binomio
*/
#include <stdio.h>
int main (){
    int a, b, qa, qb, dp;
    printf(" inserisci i coefficienti a e b : ");
    scanf ("%d%d" ,&a, &b);

    qa = a*a;
    dp = 2*(a)*(b);
    qb= b*b;

    printf("il risultato del quadrato di binomio (%dx+%d)^2 equivale a %dx^2+%dx+%dab",a, b, qa, dp, qb);

    return 0 ;
}
```

Risoluzione di un'Equazione di Secondo Grado

Questo programma in C calcola le soluzioni di un'equazione di secondo grado nella forma **$ax^2 + bx + c = 0$** . L'utente inserisce i coefficienti **a**, **b** e **c**, e il programma determina se l'equazione ha soluzioni reali o complesse, calcolando e visualizzando le soluzioni corrispondenti.



```
/*
 *scrivere un programma che calcoli l'equazione di 2 grado
 */
#include <stdio.h>
#include <math.h>

int main() {
    int a, b, c;

    printf("Inserisci i coefficienti a, b, c: ");
    scanf("%d %d %d", &a, &b, &c);

    if (a == 0) {
        printf("Non è un'equazione di secondo grado.\n");
    } else {
        int delta = b * b - 4 * a * c;
        if (delta < 0) {
            printf("L'equazione non ha soluzioni reali.\n");
        } else if (delta == 0) {
            int x = -b / (2 * a);
            printf("L'equazione ha una soluzione reale: x = %d\n", x);
        } else {
            int x1 = (-b + sqrt(delta)) / (2 * a);
            int x2 = (-b - sqrt(delta)) / (2 * a);
            printf("L'equazione ha due soluzioni reali: x1 = %d, x2 = %d\n", x1, x2);
        }
    }
    return 0;
}
```

Calcolo delle Coordinate del Punto Medio di un Segmento

Questo programma in C permette di calcolare le coordinate del punto medio di un segmento dati i suoi estremi nel piano cartesiano. L'utente inserisce le coordinate dei punti A e B, e il programma calcola e visualizza le coordinate del punto medio M utilizzando la formula:

$$M = \left(\frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2} \right)$$

```
/*
*scrivere un programma che calcoli le coordinate del punto medio
*/
#include <stdio.h>

int main (){
    float x1, y1;
    float x2,y2;
    float xm, ym ;

    printf("inserisci la cordinata del punto A =");
    scanf("%f%f",&x1,&y1);

    printf("inserisci la cordinata del punto B =");
    scanf("%f%f",&x2,&y2);

    xm = (x1+x2) / 2 ;
    ym = (y1+y2) / 2 ;

    printf("il punto medio di AB e' = (%.2f;.2f)",xm,ym);

    return 0 ;
}
```

Calcolo del Tasso di Interesse su 25 Giorni

Questo programma in C calcola l'interesse maturato su un saldo corrente in base a un tasso annuo fornito dall'utente. Dopo aver inserito il saldo e il tasso annuo, il programma calcola l'interesse per 25 giorni e restituisce il nuovo saldo aggiornato.



```
#include <stdio.h>

int main() {
    float tasso_annuo_percentuale;
    float tasso_annuo, tasso_mensile, tasso_giornaliero;
    float saldo_iniziale;
    float saldo_finale;           // saldo dopo 25 giorni con interesse
    float interesse_25_giorni;

    // Input del saldo iniziale
    printf("Inserisci il saldo corrente: ");
    scanf("%f", &saldo_iniziale);

    // Input del tasso di interesse annuo (in percentuale)
    printf("Inserisci il tasso annuo (in percentuale): ");
    scanf("%f", &tasso_annuo_percentuale);

    // Calcolo dell'interesse maturato in 25 giorni
    tasso_annuo = saldo_iniziale * tasso_annuo_percentuale / 100;
    tasso_mensile = tasso_annuo / 12;
    tasso_giornaliero = tasso_mensile / 31;
    interesse_25_giorni = tasso_giornaliero * 25;
    saldo_finale = saldo_iniziale + interesse_25_giorni;

    // Output del risultato
    printf("Interesse maturato in 25 giorni = %.2f\n", interesse_25_giorni);
    printf("Saldo totale dopo 25 giorni = %.2f\n", saldo_finale);

    return 0;
}
```

Calcolo dello Scorporo dell'IVA

Questo programma in C permette di calcolare lo scorporo dell'IVA a partire da un importo lordo inserito dall'utente. Utilizzando un'aliquota IVA del 22%, il programma calcola sia l'importo netto che la quota IVA separata. I risultati vengono poi visualizzati con due decimali per una maggiore precisione.



```
#include <stdio.h>

int main() {
    double iva_scoporata;      // IVA calcolata dal lordo
    double importo_lordo;     // Importo comprensivo di IVA
    double importo_netto;     // Importo senza IVA

    // Input dell'importo lordo
    printf("Inserisci l'importo desiderato: ");
    scanf("%lf", &importo_lordo); // Uso %lf per leggere un double

    // Calcolo dell'IVA e dell'importo netto
    iva_scoporata = importo_lordo * 0.22;
    importo_netto = importo_lordo - iva_scoporata;

    // Output dei risultati
    printf("Importo lordo: %.2f\n", importo_lordo);
    printf("IVA (22%): %.2f\n", iva_scoporata);
    printf("Importo netto: %.2f\n", importo_netto);

    return 0;
}
```

Inserimento e Stampa di 10 Numeri in un Vettore

Questo programma in C permette all'utente di inserire 10 numeri interi in un vettore utilizzando un ciclo **for**. Dopo l'inserimento, un secondo ciclo **for** stampa i numeri inseriti uno per uno. Questo programma illustra l'uso dei cicli per gestire e visualizzare dati in un array.

```
/*
*scrivere 10 numeri in un vettore usando un ciclo
*/
#include<stdio.h>
main() {
int v[10];
int i;
for(i=0;i<10;i++) {
printf("inserisci un numero ");
scanf("%d",&v[i]);
}
for(i=0;i<10;i++) {
printf("%d\n",v[i]);
}
}
```

Stampa dei Numeri Pari in un Vettore

Questo programma in C scansiona un vettore di numeri interi e stampa solo gli elementi pari. Utilizza un ciclo **for** per scorrere gli elementi e una condizione **if** per verificare la divisibilità per 2.

```
/*
*scrivere un programma che stampa solo i numeri pari in un vettore
*/
#include <stdio.h>

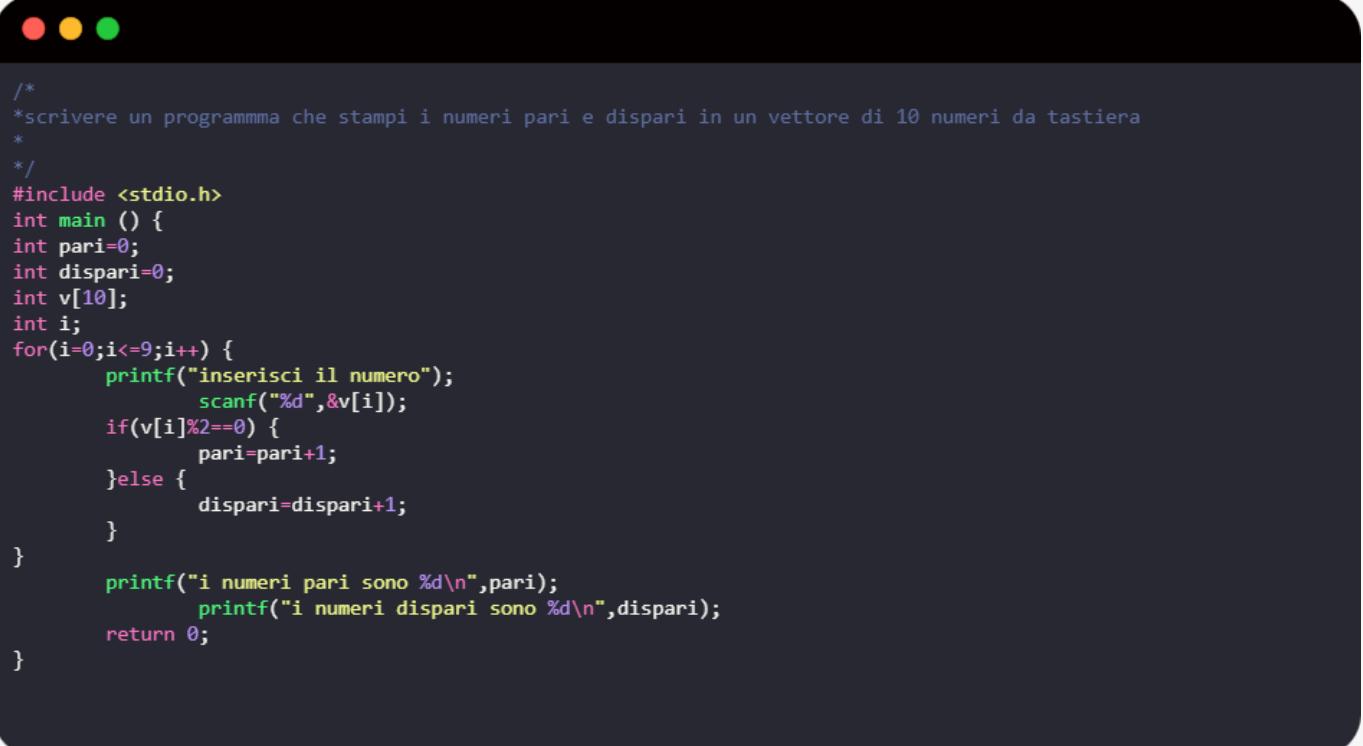
int main (){
    int v [4]={1, 2, 3, 4};

    int i;

    for(i=0;i<4;i++){
        if (v[i] % 2 == 0){
            printf(" %d ", v[i]);
        }
    }
    return 0 ;
}
```

Conteggio di Numeri Pari e Dispari in un Vettore

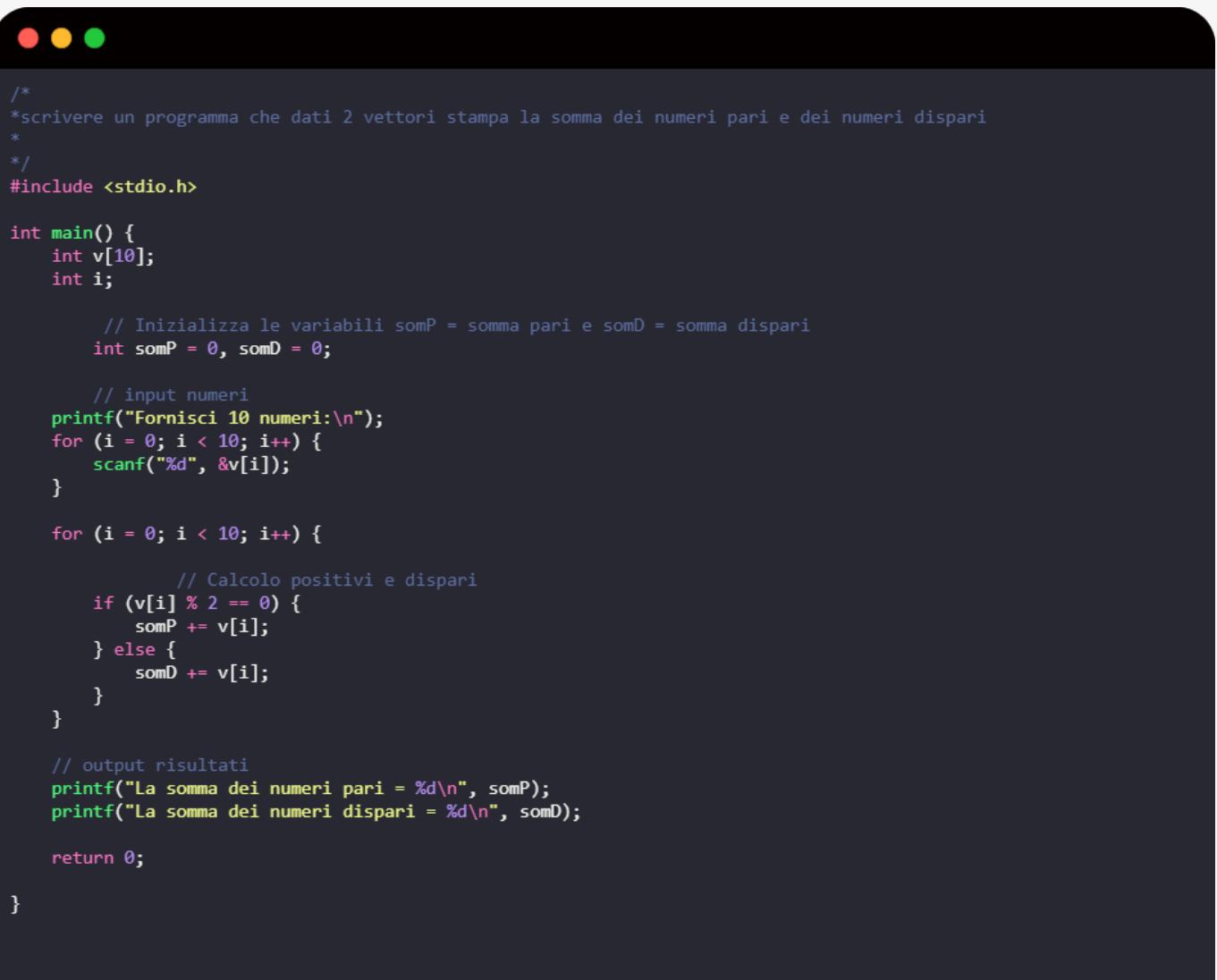
Questo programma in C permette all'utente di inserire 10 numeri interi da tastiera e conta quanti di essi sono pari e quanti sono dispari. Utilizza un ciclo **for** per leggere i numeri e una condizione **if** per verificare la parità. Infine, stampa il numero totale di valori pari e dispari inseriti.



```
/*
*scrivere un programma che stampi i numeri pari e dispari in un vettore di 10 numeri da tastiera
*/
#include <stdio.h>
int main () {
int pari=0;
int dispari=0;
int v[10];
int i;
for(i=0;i<=9;i++) {
    printf("inserisci il numero");
    scanf("%d",&v[i]);
    if(v[i]%2==0) {
        pari=pari+1;
    }else {
        dispari=dispari+1;
    }
}
printf("i numeri pari sono %d\n",pari);
printf("i numeri dispari sono %d\n",dispari);
return 0;
}
```

Somma dei Numeri Pari e Dispari in un Vettore

Questo programma in C permette all'utente di inserire 10 numeri interi in un vettore e calcola separatamente la somma dei numeri pari e quella dei numeri dispari. Utilizza un ciclo **for** per acquisire i dati e un altro ciclo per distinguere e sommare i valori pari e dispari. Infine, stampa i risultati.



```
/*
*scrivere un programma che dati 2 vettori stampa la somma dei numeri pari e dei numeri dispari
*/
#include <stdio.h>

int main() {
    int v[10];
    int i;

    // Inizializza le variabili somP = somma pari e somD = somma dispari
    int somP = 0, somD = 0;

    // input numeri
    printf("Fornisci 10 numeri:\n");
    for (i = 0; i < 10; i++) {
        scanf("%d", &v[i]);
    }

    for (i = 0; i < 10; i++) {

        // Calcolo positivi e dispari
        if (v[i] % 2 == 0) {
            somP += v[i];
        } else {
            somD += v[i];
        }
    }

    // output risultati
    printf("La somma dei numeri pari = %d\n", somP);
    printf("La somma dei numeri dispari = %d\n", somD);

    return 0;
}
```

Calcolo della Somma dei Numeri Pari in un Vettore

Questo programma in C permette all'utente di inserire 10 numeri interi in un vettore e calcola la somma di quelli pari. Utilizza un ciclo **for** per leggere i numeri e un altro ciclo per sommare solo i valori divisibili per 2. Infine, stampa il risultato.



```
/*
*scrivere un programma che calcola la somma dei numeri pari nel vettore
*/
#include <stdio.h>

int main (){
    int v[10];
    int i;
    int som = 0;

    // Inserisci 10 numeri
    printf("Inserisci 10 numeri:\n");
    for (i = 0; i < 10; i++) {
        scanf("%d", &v[i]);
    }

    // Calcolare la somma dei numeri pari
    for (i = 0; i < 10; i++) {
        if (v[i] % 2 == 0) {
            som += v[i];
        }
    }

    printf("La somma dei numeri pari e': %d\n", som);

    return 0;
}
```

Calcolo della Media dei Numeri Pari in un Vettore

Questo programma in C consente all'utente di inserire 10 numeri interi e calcola la media dei numeri pari. Utilizza un ciclo for per acquisire i numeri e verifica la parità. Se trova numeri pari, ne accumula la somma e tiene traccia della loro quantità. Alla fine, calcola e stampa la media, oppure informa l'utente se non sono presenti numeri pari.

```
/*
*scrivere un programma che calcola la media dei numeri pari in un vettore di 10 numeri da tastiera
*/
#include<stdio.h>
main() {
    int v [10];
    int media=0;
    int i;
    int somma=0;
    int numeri;
    int pari=0;
    for(i=0;i<=9;i++) {
        printf("inserisci i numeri");
        scanf("%d",&v[i]);
        if(v[i]%2==0) {
            somma=somma+v[i];
            pari++;
        }
    }
    if(pari==0) {
        printf("non e' presente nessun numero pari");
        return 0;
    }
    else {
        media=somma/pari;
        printf("la media e' %d",media);
    }
    return 0;
}
```

Calcolo Somma dei Prodotti tra due Vettori

Questo programma in linguaggio C permette di calcolare la somma dei prodotti tra gli elementi di due vettori inseriti dall'utente.

Funzionamento:

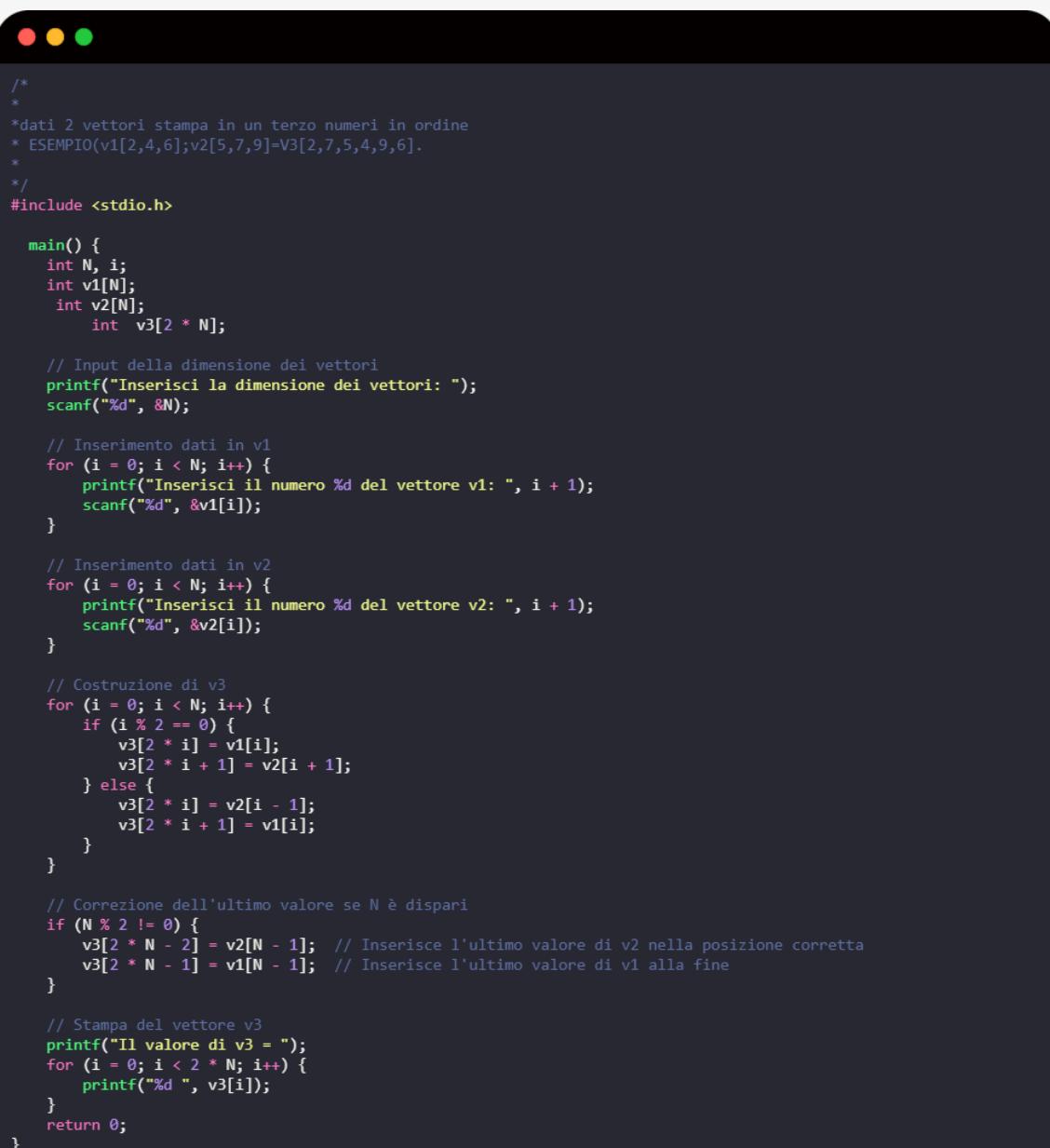
1. L'utente inserisce la dimensione dei vettori (**n**).
2. Il programma chiede all'utente di inserire gli elementi del primo vettore.
3. Successivamente, chiede di inserire gli elementi del secondo vettore.
4. Per ogni posizione **i**, il programma calcola il prodotto **v1[i] * v2[i]** e lo somma al risultato totale.
5. Infine, viene stampata la somma dei prodotti.



```
/*
* restituire la somma dei prodotti in un vettore tramite un ciclo
*/
#include <stdio.h>
int main() {
    int n;
    int i;
printf("Inserisci la dimensione dei vettori: ");
scanf("%d", &n);
int v1[n], v2[n];
int somma_prodotti = 0;
for (i = 0; i < n; i++) {
printf("inserisci i valori del primo vettore: ");
scanf("%d", &v1[i]);
}
for (i = 0; i < n; i++) {
printf("inserisci i valori del secondo vettore: ");
scanf("%d", &v2[i]);
}
for (i = 0; i < n; i++) {
    somma_prodotti += v1[i] * v2[i];
}
printf("La somma dei prodotti e': %d\n", somma_prodotti);
return 0;
}
```

Unione Alternata di Due Vettori in un Terzo Vettore

Questo programma in C prende in input due vettori di dimensione specificata dall'utente e genera un terzo vettore alternando gli elementi dei primi due. Dopo aver richiesto la lunghezza dei vettori e i valori da inserire, il programma costruisce il nuovo vettore disponendo in ordine alternato un elemento del primo vettore e uno del secondo. Se la dimensione è dispari, l'ultimo elemento del secondo vettore viene inserito prima dell'ultimo elemento del primo. Infine, il programma stampa il contenuto del vettore risultante.



```
/*
*
*dati 2 vettori stampa in un terzo numeri in ordine
* ESEMPIO:v1[2,4,6];v2[5,7,9]=v3[2,7,5,4,9,6].
*/
#include <stdio.h>

main() {
    int N, i;
    int v1[N];
    int v2[N];
    int v3[2 * N];

    // Input della dimensione dei vettori
    printf("Inserisci la dimensione dei vettori: ");
    scanf("%d", &N);

    // Inserimento dati in v1
    for (i = 0; i < N; i++) {
        printf("Inserisci il numero %d del vettore v1: ", i + 1);
        scanf("%d", &v1[i]);
    }

    // Inserimento dati in v2
    for (i = 0; i < N; i++) {
        printf("Inserisci il numero %d del vettore v2: ", i + 1);
        scanf("%d", &v2[i]);
    }

    // Costruzione di v3
    for (i = 0; i < N; i++) {
        if (i % 2 == 0) {
            v3[2 * i] = v1[i];
            v3[2 * i + 1] = v2[i + 1];
        } else {
            v3[2 * i] = v2[i - 1];
            v3[2 * i + 1] = v1[i];
        }
    }

    // Correzione dell'ultimo valore se N è dispari
    if (N % 2 != 0) {
        v3[2 * N - 2] = v2[N - 1]; // Inserisce l'ultimo valore di v2 nella posizione corretta
        v3[2 * N - 1] = v1[N - 1]; // Inserisce l'ultimo valore di v1 alla fine
    }

    // Stampa del vettore v3
    printf("Il valore di v3 = ");
    for (i = 0; i < 2 * N; i++) {
        printf("%d ", v3[i]);
    }
    return 0;
}
```

Contatore Maggiorenni e Minorenni

Questo programma in C utilizza un ciclo **do-while** per consentire all'utente di inserire ripetutamente il nome e l'età di diverse persone. Dopo ogni inserimento, il programma determina se la persona è maggiorenne (età maggiore o uguale a 18) o minorenne (età inferiore a 18) e aggiorna i rispettivi contatori. L'utente può decidere se continuare o terminare l'inserimento dei dati rispondendo a una specifica domanda. Al termine, il programma stampa il numero totale di maggiorenni e minorenni registrati.



```
/*
*dato un programma stampare quanti maggiorenni e quanti minorenni ci sono
*/
#include <stdio.h>

int main(){
    /* input */
    char nome[20];
    int eta;
    int risp;

    /* output */
    int conta_maggiorenni = 0, conta_minorenni = 0;

    do {
        printf("Nome: ");
        scanf("%s", nome);
        printf("Eta: ");
        scanf("%d", &eta);

        if (eta >= 18){
            conta_maggiorenni++; // Incrementa i maggiorenni
        } else {
            conta_minorenni++; // Incrementa i minorenni
        }

        printf("Elenco finito: 0=no, 1=sì? ");
        scanf("%d", &risp);
    } while (risp == 1); // Continua se risp è 1, altrimenti termina

    printf("I maggiorenni sono: %d\n", conta_maggiorenni);
    printf("I minorenni sono: %d\n", conta_minorenni);

    return 0;
}
```

Programma per Trovare il Numero Maggiore tra due Interi

Questo programma in linguaggio C legge due numeri interi inseriti dall'utente e determina quale dei due è maggiore. Se i numeri sono uguali, lo segnala all'utente. Il confronto viene effettuato utilizzando strutture condizionali (**if, else if, else**), e il risultato viene stampato a schermo.

```
/*
*realizzare un programma in linguaggio C che legga da input due numeri
*interi e stampi in output il maggiore tra i 2 numeri
*/
#include <stdio.h>

int main() {
    int numero_1, numero_2;

    // Input dei due numeri
    printf("Inserisci il primo numero: ");
    scanf("%d", &numero_1);

    printf("Inserisci il secondo numero: ");
    scanf("%d", &numero_2);

    // Confronto tra i due numeri
    if (numero_1 > numero_2) {
        printf("Il numero maggiore è: %d\n", numero_1);
    } else if (numero_1 < numero_2) {
        printf("Il numero maggiore è: %d\n", numero_2);
    } else {
        printf("I numeri sono uguali.\n");
    }

    return 0;
}
```

1) Verifica della Validità di una Classe Scolastica

Questo programma in C legge una sequenza di due caratteri inseriti dall'utente e verifica se rappresentano il nome di una classe di scuola superiore.

- Il primo carattere deve essere un numero compreso tra 1 e 5 (indicante l'anno scolastico).
- Il secondo carattere deve essere una lettera maiuscola dell'alfabeto (indicante la sezione).
- Se l'input rispetta queste regole, il programma conferma che la sequenza è valida; altrimenti, restituisce un messaggio di errore.

Il programma utilizza **getchar()** per acquisire i caratteri .
"L'esercizio riportato qui sotto si suddivide in 2 pagine."



```
/* carattere5.c
*
* Data una sequenza di due caratteri (da leggere uno alla volta con la
* funzione getchar), stabilire se la sequenza rappresenta il nome di una
* classe di alunni della scuola superiore (esempi: 3C è una sequenza che
* indica una classe; C3 oppure 6A non sono sequenze valide).
*/
#include <stdio.h>

int main(void) {
    char cla, sez;

    printf("Inserire classe e sezione: ");
    cla = getchar();
    sez = getchar();

    if (sez >= 'a' && sez <= 'z') {
        sez -= 32;
    }

    switch (cla) {
        case '1':
        case '2':
        case '3':
        case '4':
        case '5':
```

2) Verifica della Validità di una Classe Scolastica

```
switch (sez) {  
    case 'A':  
    case 'B':  
    case 'C':  
    case 'D':  
    case 'E':  
    case 'F':  
    case 'G':  
    case 'H':  
    case 'I':  
    case 'J':  
    case 'K':  
    case 'L':  
    case 'M':  
    case 'N':  
    case 'O':  
    case 'P':  
    case 'Q':  
    case 'R':  
    case 'S':  
    case 'T':  
    case 'U':  
    case 'V':  
    case 'W':  
    case 'X':  
    case 'Y':  
    case 'Z':  
        printf("La sequenza rappresenta una classe.\n");  
        break;  
    default:  
        printf("Sequenza non valida.\n");  
    }  
    break;  
default:  
    printf("Sequenza non valida.\n");  
}  
  
return 0;  
}
```

Identificazione del Tipo di Carattere

Questo programma in C riceve in ingresso un carattere e determina se si tratta di una lettera maiuscola, una lettera minuscola, una cifra numerica oppure un altro carattere speciale. Utilizza un'istruzione **switch** per classificare il carattere e stampare il risultato corrispondente.

```
/*
*
* Dato in ingresso un carattere, visualizzare a video se si tratta di una
* cifra numerica, una lettera minuscola, una lettera maiuscola,
* oppure un altro carattere utilizzando uno switch.
*/
#include <stdio.h>
main() {
char carattere [20];
printf("inserisci il carattere:");
scanf("%s",carattere);
switch(carattere[0]) {
case 'A' ... 'Z':
printf("il carattere e' una lettera maiuscola.\n");
break;

case 'a' ... 'z':
printf("il carattere e' una lettera minuscola.\n");
break;

case '0' ... '9':
printf("il carattere e' un numero.\n");
break;

default:
printf("il carattere e' speciale..\n");
}
return 0;
}
```

Verifica della Parità dei Numeri in un Vettore

Il programma richiede all'utente di inserire 10 numeri interi, che vengono memorizzati in un array. Per ogni numero inserito, il programma verifica se è maggiore o minore di zero e stampa un messaggio corrispondente.

```
/*
*in un vettore di 10 numeri stampare se i numeri sono maggiore o minore di 0
*/
#include <stdio.h>
main () {
int v[10];
int i;
for(i=0;i<=9;i++) {
printf("inserisci un numero:");
scanf("%d",&v[i]);

if(v[i]>0) {
printf("il numero e' maggiore di 0\n");
}
else{
printf("il numero e' minore di 0\n");
}
}
}
```

Conteggio dei Numeri Pari e Dispari in un Vettore

Questo programma in C richiede all'utente di inserire 10 numeri interi, memorizzandoli in un array. Durante l'inserimento, il programma conta quanti di questi numeri sono pari e quanti sono dispari. Al termine, visualizza il totale dei numeri pari e dispari inseriti.



```
/*
*scrivere un programma che conti i numeri pari e i numeri dispari nel vettore
*/
#include <stdio.h>

int main (){
    int pari = 0 ;
    int dispari =0;
    int v [10];
    int i ;

    for (i =0;i <=9;i++){
        printf("inserisci il numero ");
        scanf("%d",&v[i]);

        if (v[i]%2 == 0 ){
            pari++;
        }else {
            dispari++;
        }
    }

    printf("i numeri pari sono %d e i numeri dispari sono %d ",pari , dispari );

    return 0 ;
}
```

Somma dei Numeri Positivi e Conteggio dei Negativi

Questo programma in C richiede all'utente di inserire una serie di numeri, terminando l'input con il numero 0. Durante l'esecuzione, il programma somma solo i numeri positivi inseriti e conta quanti numeri negativi sono stati forniti. Alla fine, il programma stampa la somma dei numeri positivi e il conteggio dei numeri negativi.



```
#include <stdio.h>

int main() {
    int num;
    int somma_positivi = 0;
    int conta_negativi = 0;

    printf("Inserisci una serie di numeri e termina con 0:\n");
    scanf("%d", &num);

    while (num != 0) {
        if (num > 0) {
            somma_positivi += num;
        } else if (num < 0) {
            conta_negativi++;
        }

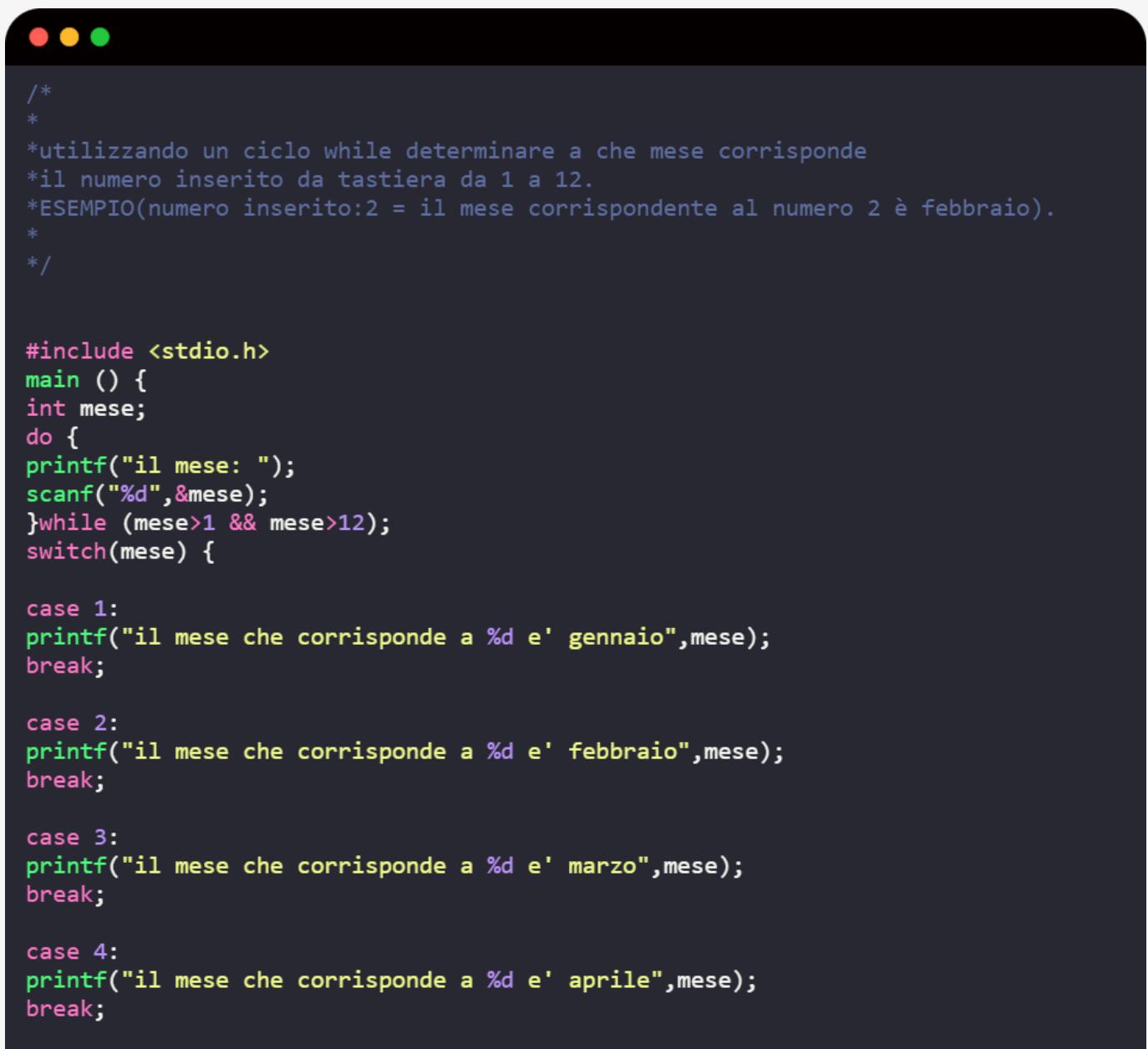
        scanf("%d", &num);
    }

    printf("Somma dei numeri positivi: %d\n", somma_positivi);
    printf("Quantità dei numeri negativi: %d\n", conta_negativi);

    return 0;
}
```

1) Determinare il Mese Corrispondente a un Numero

Questo programma in C chiede all'utente di inserire un numero intero compreso tra 1 e 12 e determina a quale mese dell'anno corrisponde il numero inserito. Se l'utente inserisce un numero al di fuori di questo intervallo, il programma richiede nuovamente l'inserimento fino a quando viene fornito un numero valido. Una volta ottenuto un numero valido, il programma utilizza una struttura di controllo **switch** per associare il numero al mese corrispondente e visualizzare il risultato.



```
/*
*
*utilizzando un ciclo while determinare a che mese corrisponde
*il numero inserito da tastiera da 1 a 12.
*ESEMPIO(numero inserito:2 = il mese corrispondente al numero 2 è febbraio).
*/
#include <stdio.h>
main () {
int mese;
do {
printf("il mese: ");
scanf("%d",&mese);
}while (mese>1 && mese>12);
switch(mese) {

case 1:
printf("il mese che corrisponde a %d e' gennaio",mese);
break;

case 2:
printf("il mese che corrisponde a %d e' febbraio",mese);
break;

case 3:
printf("il mese che corrisponde a %d e' marzo",mese);
break;

case 4:
printf("il mese che corrisponde a %d e' aprile",mese);
break;
}
}
```

2) Determinare il Mese Corrispondente a un Numero

```
case 5:  
printf("il mese che corrisponde a %d e' maggio",mese);  
break;  
  
case 6:  
printf("il mese che corrisponde a %d e' giugno",mese);  
break;  
  
case 7:  
printf("il mese che corrisponde a %d e' luglio",mese);  
break;  
  
case 8:  
printf("il mese che corrisponde a %d e' agosto",mese);  
break;  
  
case 9:  
printf("il mese che corrisponde a %d e' settembre",mese);  
break;  
  
case 10:  
printf("il mese che corrisponde a %d e' ottobre",mese);  
break;  
  
case 11:  
printf("il mese che corrisponde a %d e' novembre",mese);  
break;  
  
case 12:  
printf("il mese che corrisponde a %d e' dicembre",mese);  
break;  
  
default:  
printf("il numero non corrisponde a nessun      mese");  
}  
}
```

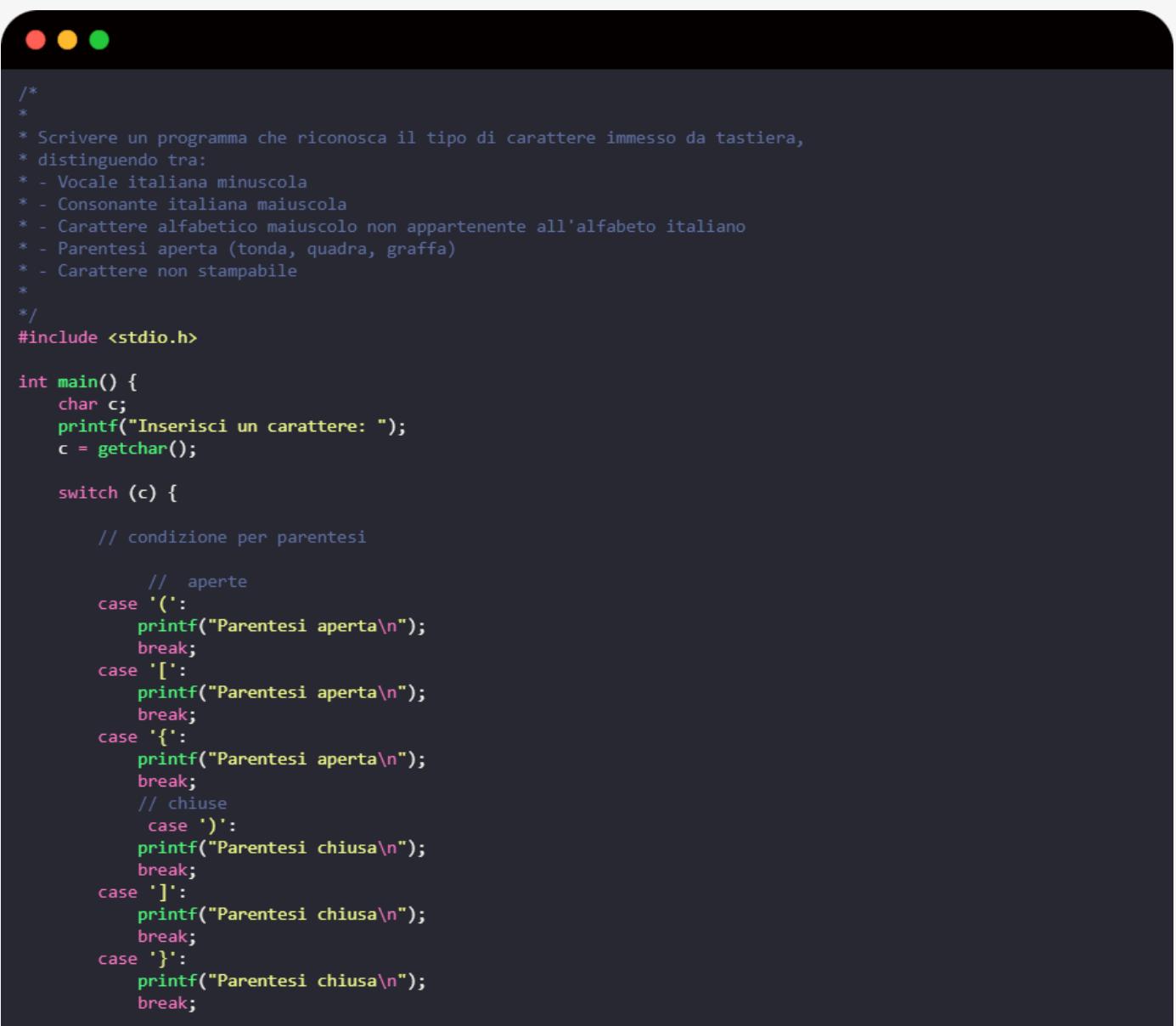
1) Riconoscimento del Tipo di Carattere Inserito

Questo programma in C identifica il tipo di carattere inserito dall'utente e lo classifica nelle seguenti categorie:

- Vocale italiana minuscola (a, e, i, o, u)
- Vocale italiana maiuscola (A, E, I, O, U)
- Consonante italiana maiuscola (tutte le lettere maiuscole dell'alfabeto italiano escluse le vocali)
- Carattere alfabetico maiuscolo non appartenente all'alfabeto italiano (J, K, W, X, Y)
- Parentesi aperta o chiusa (tonda (), quadra [], graffa {})
- Altro carattere (qualsiasi altro carattere non compreso nelle categorie sopra)

Il programma utilizza un'istruzione switch per la classificazione e gestisce le condizioni con un controllo if nel default per le consonanti italiane maiuscole.

"L'esercizio riportato qui sotto si suddivide in 2 pagine."



```
/*
 *
 * Scrivere un programma che riconosca il tipo di carattere immesso da tastiera,
 * distinguendo tra:
 * - Vocale italiana minuscola
 * - Consonante italiana maiuscola
 * - Carattere alfabetico maiuscolo non appartenente all'alfabeto italiano
 * - Parentesi aperta (tonda, quadra, graffa)
 * - Carattere non stampabile
 */
#include <stdio.h>

int main() {
    char c;
    printf("Inserisci un carattere: ");
    c = getchar();

    switch (c) {
        // condizione per parentesi
        // aperte
        case '(':
            printf("Parentesi aperta\n");
            break;
        case '[':
            printf("Parentesi aperta\n");
            break;
        case '{':
            printf("Parentesi aperta\n");
            break;
        // chiuse
        case ')':
            printf("Parentesi chiusa\n");
            break;
        case ']':
            printf("Parentesi chiusa\n");
            break;
        case '}':
            printf("Parentesi chiusa\n");
            break;
    }
}
```

2) Riconoscimento del Tipo di Carattere Inserito

```
case 'a':
    printf("Vocale italiana minuscola\n");
    break;
case 'e':
    printf("Vocale italiana minuscola\n");
    break;
case 'i':
    printf("Vocale italiana minuscola\n");
    break;
case 'o':
    printf("Vocale italiana minuscola\n");
    break;
case 'u':
    printf("Vocale italiana minuscola\n");

    break;
case 'A':
    printf("Vocale italiana maiuscola\n");
    break;
case 'E':
    printf("Vocale italiana maiuscola\n");
    break;
case 'I':
    printf("Vocale italiana maiuscola\n");
    break;
case 'O':
    printf("Vocale italiana maiuscola\n");
    break;
case 'U':
    printf("Vocale italiana maiuscola\n");
    break;

    // condizione vocali maiuscolo non italiane
case 'J':
    printf("Carattere alfabetico maiuscolo non appartenente all'alfabeto italiano\n");
    break;
case 'K':
    printf("Carattere alfabetico maiuscolo non appartenente all'alfabeto italiano\n");
    break;
case 'W':
    printf("Carattere alfabetico maiuscolo non appartenente all'alfabeto italiano\n");
    break;
case 'X':
    printf("Carattere alfabetico maiuscolo non appartenente all'alfabeto italiano\n");
    break;
case 'Y':
    printf("Carattere alfabetico maiuscolo non appartenente all'alfabeto italiano\n");
    break;
default:

    if (c >= 'A' && c <= 'Z') { // condizione lettere alfabeto
        printf("Consonante italiana maiuscola\n");
    } else {
        printf("Altro carattere\n"); // altro carattere
    }
    break;
}

return 0;
}
```

Verifica di Caratteri Speciali in un Nome o Cognome

Questo programma in C permette all'utente di inserire un nome o un cognome e verifica se contiene caratteri speciali. Se l'input contiene solo lettere dell'alfabeto, il programma conferma la validità del nome; in caso contrario, chiede di reinserirlo. L'implementazione utilizza un ciclo **do-while** per garantire che l'utente inserisca un nome corretto prima di terminare l'esecuzione.

```
#include <stdio.h>
#include <ctype.h> // per isalpha()

int main() {
    char nome[20];
    int valido, i;

    do {
        valido = 1;

        printf("Inserisci il tuo nome o cognome: ");
        scanf("%s", nome);

        for (i = 0; nome[i] != '\0'; i++) {
            if (!isalpha(nome[i])) {
                valido = 0;
                printf("Errore: inserire solo lettere dell'alfabeto.\n");
                break;
            }
        }
    } while (!valido);

    printf("Il nome \"%s\" non contiene caratteri speciali.\n", nome);
    return 0;
}
```

Algoritmi di Ordinamento di Vettori

Un algoritmo di ordinamento è un metodo per disporre gli elementi di un vettore in ordine crescente o decrescente. Tra i più comuni ci sono Bubble Sort, Selection Sort e Merge Sort, ognuno con un funzionamento diverso.

Bubble Sort

È uno degli algoritmi più semplici. Funziona confrontando coppie di elementi vicini e scambiandoli se non sono nell'ordine giusto. Questo processo viene ripetuto più volte fino a quando il vettore è completamente ordinato. Il nome deriva dal fatto che i numeri più grandi "risalgono" lentamente verso la fine, come bolle d'aria in un liquido.

Selection Sort

Questo algoritmo cerca il valore più piccolo in tutto il vettore e lo sposta nella prima posizione. Poi cerca il secondo più piccolo e lo sposta al secondo posto, e così via fino alla fine.

Merge Sort

È un algoritmo più avanzato. Divide il vettore a metà, ordina separatamente le due parti e poi le unisce in modo ordinato. Questo metodo è chiamato "divide et impera" perché il problema viene diviso in sotto-problemi più piccoli e poi risolto.

In generale, Bubble Sort e Selection Sort sono adatti per piccole liste, mentre Merge Sort è una scelta migliore per liste più grandi o quando serve un ordinamento efficiente.

Selection Sort – Ordinamento per Selezione

Questo programma implementa l'algoritmo di Selection Sort per ordinare un array di 6 numeri inseriti dall'utente. L'algoritmo seleziona l'elemento più piccolo in ogni iterazione e lo scambia con il primo elemento non ordinato, ripetendo il processo fino a ottenere un array ordinato in ordine crescente.



```
#include <stdio.h>

int main() {
    int v[6], i, j, min, temp;

    // Input: lettura del vettore
    printf("Inserisci 6 numeri:\n");
    for (i = 0; i < 6; i++) {
        scanf("%d", &v[i]);
    }

    // Algoritmo Selection Sort
    for (i = 0; i < 5; i++) {
        min = i; // Assume che l'elemento in posizione sia il minimo
        for (j = i + 1; j < 6; j++) {
            if (v[j] < v[min]) { // Trova il minimo nel resto dell'array
                min = j;
            }
        }
        // Scambia l'elemento minimo trovato con il primo non ordinato
        temp = v[i];
        v[i] = v[min];
        v[min] = temp;
    }

    // Output: stampa il vettore ordinato
    printf("Il vettore in ordine è: ");
    for (i = 0; i < 6; i++) {
        printf("%d ", v[i]);
    }
    printf("\n");

    return 0;
}
```

Bubble Sort – Ordinamento per Scambio Successivo

Questo programma utilizza l'algoritmo di Bubble Sort per ordinare un array di 5 numeri in ordine crescente. Ad ogni passaggio, gli elementi vengono confrontati a coppie e scambiati se necessario, facendo emergere gradualmente il valore più grande alla fine dell'array. Alla fine, viene stampato il numero totale di scambi effettuati.

```
#include <stdio.h>

int cont = 0; // Contatore degli scambi

void bubble_sort(int arr[], int Lunghezza) {
    int i, j, tmp;
    for (i = 0; i < lunghezza - 1; i++) { // Passaggi esterni
        for (j = 0; j < lunghezza - i - 1; j++) { // Confronti e scambi interni
            if (arr[j] > arr[j + 1]) {
                tmp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = tmp;
                cont++; // Incrementa il contatore
            }
        }
    }
    // Stampa il risultato
    printf("Array ordinato: ");
    for (i = 0; i < lunghezza; i++) {
        printf("%d ", arr[i]);
    }
    printf("\nNumero di scambi: %d\n", cont);
}

int main() {
    int arr[] = {21, 10, 18, 7, 5};
    int lunghezza = 5;
    bubble_sort(arr, lunghezza); // Ordinamento
    return 0;
}
```

1) Merge Sort - Ordinamento per Fusione

Questo programma implementa l'algoritmo Merge Sort, un algoritmo di ordinamento basato sulla strategia "divide et impera". Divide ricorsivamente l'array in due metà, le ordina separatamente e poi le unisce (fusionando) in un array ordinato. Durante il processo, viene conteggiato il numero totale di operazioni di fusione.

```
#include <stdio.h>

int cont = 0; // Contatore delle operazioni

// Funzione per unire due sotto-array ordinati
void unisci(int arr[], int sx, int m, int dx) {
    int j, k;
    int lunghezza1 = m - sx + 1; // Lunghezza primo sotto-array
    int lunghezza2 = dx - m; // Lunghezza secondo sotto-array

    int SX[lunghezza1], DX[lunghezza2];

    // Copia i dati nei sotto-array
    for (i = 0; i < lunghezza1; i++) {
        SX[i] = arr[sx + i];
    }
    for (j = 0; j < lunghezza2; j++) {
        DX[j] = arr[m + 1 + j];
    }

    // Unisce i sotto-array ordinati
    i = 0; j = 0; k = sx;
    while (i < lunghezza1 && j < lunghezza2) {
        if (SX[i] <= DX[j]) {
            arr[k] = SX[i];
            i++;
        } else {
            arr[k] = DX[j];
            j++;
        }
        cont++; // Conta l'operazione di fusione
        k++;
    }

    // Gestisce gli eventuali resti
    while (i < lunghezza1) {
        arr[k] = SX[i];
        i++; k++;
        cont++;
    }
    while (j < lunghezza2) {
        arr[k] = DX[j];
        j++; k++;
        cont++;
    }
}
```

2) Merge Sort - Ordinamento per Fusione

```
// Funzione ricorsiva per il Merge Sort
void merge_sort(int arr[], int sx, int dx) {
    if (sx < dx) {
        int m = (sx + dx) / 2; // Trova il punto medio
        merge_sort(arr, sx, m); // Ordina la prima metà
        merge_sort(arr, m + 1, dx); // Ordina la seconda metà
        unisci(arr, sx, m, dx); // Unisci le due metà
    }

    // Stampa l'array ordinato
    printf("Array ordinato: ");
    for (i = 0; i <= dx; i++) {
        printf("%d ", arr[i]);
    }
    printf("\nNumero di operazioni: %d\n", cont); // Stampa il numero di operazioni
}

int main() {
    int arr[] = {21, 10, 18, 7, 5}; // Array da ordinare
    int lunghezza = 5; // Lunghezza dell'array

    merge_sort(arr, 0, lunghezza - 1); // Chiamata alla funzione merge_sort
    return 0;
}
```

Varianti di Esercizi con Funzioni

Cos'è una Funzione e Come Funziona

Una funzione è come una scatola che fa un lavoro specifico. Può ricevere degli ingredienti (i parametri), fare qualcosa con questi e poi restituire un risultato (un valore).

Ad esempio, una funzione che somma due numeri potrebbe prendere i numeri, fare la somma e restituire il risultato.

Come si Scrive una Funzione

Ecco un esempio di come scrivere una funzione in modo semplice:

```
int somma(int a, int b) {  
    // La logica successiva implementa il calcolo della somma  
}
```

- **int**: significa che la funzione restituisce un numero intero.
- **somma**: è il nome della funzione.
- **(int a, int b)**: sono i numeri che la funzione usa per fare il suo lavoro (i parametri).

Cos'è una Chiamata di Funzione

Per usare la funzione, dobbiamo chiamarla nel programma. Una chiamata di funzione è come chiedere alla scatola di fare il suo lavoro.

Ecco come si chiama la funzione somma:

```
int risultato = somma(5, 3);  
printf("La somma è: %d", risultato);
```

In questo caso:

1. `somma(5, 3)` è la chiamata alla funzione.
2. La funzione somma i numeri ($5 + 3$) e restituisce il risultato (8).
3. Il risultato (8) viene mostrato sullo schermo con `printf`.

Quindi, una funzione è semplicemente un modo per fare cose ripetitive senza dover scrivere lo stesso codice ogni volta!

Maggiore tra Due Numeri – Calcolo con Funzione

Questo programma in C utilizza una funzione chiamata **maggiore** per confrontare due numeri inseriti dall'utente e determinare quale dei due è il maggiore. La funzione riceve i due numeri come input, esegue il confronto e stampa il numero maggiore. L'uso delle funzioni rende il codice più modulare e facilmente riutilizzabile, separando la logica del calcolo dalla parte di interazione con l'utente.



```
#include <stdio.h>

// Funzione che calcola il numero maggiore tra due numeri
void maggiore(int A, int B) {
    if (A > B) {
        printf("Il numero maggiore è: %d\n", A);
    } else {
        printf("Il numero maggiore è: %d\n", B);
    }
}

int main() {
    int numero_1, numero_2;

    // Input dei due numeri
    printf("Inserisci il primo numero: ");
    scanf("%d", &numero_1);

    printf("Inserisci il secondo numero: ");
    scanf("%d", &numero_2);

    // Chiamata alla funzione per determinare il numero maggiore
    maggiore(numero_1, numero_2);

    return 0;
}
```

Numero Precedente e Successivo – Calcolo con Funzione

Questo programma in C utilizza una funzione per calcolare il numero precedente e successivo a un numero fornito dall'utente. La funzione **calcolaPrecedenteSuccessivo** prende in input un numero intero e, all'interno della funzione, calcola sia il numero che lo precede (**numero - 1**) sia il numero che lo segue (**numero + 1**). I risultati vengono poi stampati a schermo.
L'uso di funzioni consente una struttura più modulare e riutilizzabile del codice, separando la logica di calcolo dalla parte di interazione con l'utente.

```
#include <stdio.h>

int precedente; // Variabile globale per il numero precedente
int successivo; // Variabile globale per il numero successivo

// Funzione per calcolare il numero precedente e successivo
void calcolaPrecedenteSuccessivo(int numero) {
    precedente = numero - 1; // Calcola il numero precedente
    successivo = numero + 1; // Calcola il numero successivo

    // Stampa i risultati
    printf("Numero inserito: %d\n", numero);
    printf("Numero precedente: %d\n", precedente);
    printf("Numero successivo: %d\n", successivo);
}

int main() {
    int numero; // Variabile per memorizzare il numero inserito dall'utente

    // Chiede all'utente di inserire un numero
    printf("Inserisci un numero: ");
    scanf("%d", &numero);

    // Chiama la funzione per calcolare e stampare il numero precedente e successivo
    calcolaPrecedenteSuccessivo(numero);

    return 0; // Fine del programma
}
```

Calcolo della Somma tra Due Numeri – Calcolo con Funzione

In questo programma, l'utente inserisce due numeri interi e, tramite una funzione chiamata **somma**, viene calcolata la somma tra i due numeri. La funzione riceve i due numeri come parametri, calcola il risultato della somma e lo stampa a schermo. Questo esempio illustra come utilizzare una funzione in C per eseguire un'operazione (**somma**) separando la logica di calcolo dalla parte di **input/output**, migliorando la modularità e la leggibilità del programma. L'uso di funzioni permette anche di riutilizzare facilmente la stessa logica in altri contesti.



```
#include <stdio.h>

int num1, num2; // Variabili globali per i numeri

// Funzione che calcola la somma di due numeri
void somma(int a, int b) {
    int risultato = 0; // Variabile per il risultato della somma
    risultato = a + b; // Somma i due numeri
    printf("%d", risultato); // Stampa il risultato
}

int main() {
    // Input dei numeri
    printf("inserisci il primo numero: ");
    scanf("%d", &num1); // Legge il primo numero

    printf("inserisci il secondo numero: ");
    scanf("%d", &num2); // Legge il secondo numero

    somma(num1, num2); // Chiama la funzione per calcolare e stampare la somma
    return 0; // Termina il programma
}
```

Calcolo della Somma di Dieci Numeri – Calcolo con Funzione

In questo programma, l'utente inserisce dieci numeri interi, che vengono memorizzati in un array. La funzione **sommaDieciNumeri** riceve l'array di numeri come parametro, calcola la loro somma e la stampa. Questo esempio mostra come utilizzare una funzione per eseguire il calcolo separatamente dalla parte di **input**, migliorando la modularità e la gestione del codice. La struttura del programma consente di gestire più numeri con una singola funzione, semplificando il codice e la sua manutenzione.

```
#include <stdio.h>

int i; // Variabile globale utilizzata nei cicli for

// Funzione che calcola la somma di 10 numeri
void sommaDieciNumeri(int numeri[10]) {
    int somma = 0; // Variabile per memorizzare la somma dei numeri

    // Ciclo per sommare tutti i numeri nell'array
    for (i = 0; i < 10; i++) {
        somma += numeri[i]; // Somma ogni numero all'array
    }

    printf("La somma dei numeri è: %d\n", somma); // Stampa il risultato
}

int main() {
    int numeri[10]; // Array per memorizzare i 10 numeri

    printf("Inserisci 10 numeri interi:\n");
    // Ciclo per leggere i 10 numeri dall'utente
    for (i = 0; i < 10; i++) {
        printf("Numero %d: ", i + 1); // Chiede il numero all'utente
        scanf("%d", &numeri[i]); // Legge il numero
    }

    sommaDieciNumeri(numeri); // Chiama la funzione per calcolare e stampare la somma

    return 0; // Termina il programma
}
```

Successione di Fibonacci – Calcolo con Funzione

Questo programma in C calcola e stampa i numeri della successione di Fibonacci fino alla posizione inserita dall'utente. La funzione **Fibonacci** riceve un numero intero che rappresenta la posizione desiderata nella successione e calcola i numeri precedenti utilizzando la formula ricorsiva di Fibonacci. Vengono stampati tutti i numeri fino alla posizione richiesta, gestendo i casi per **n = 0** e **n = 1** separatamente. La struttura del programma è modulare, separando la logica di calcolo dalla parte di **input/output**.

```
#include <stdio.h>

// Funzione per calcolare e stampare la successione di Fibonacci fino alla posizione n
void fibonacci(int n) {
    int a = 0, b = 1, next, i;

    if (n == 0) {
        printf("I numeri della successione di Fibonacci fino alla posizione %d sono: 0\n", n);
        return;
    }

    printf("I numeri della successione di Fibonacci fino alla posizione %d sono:\n", n);
    printf("%d ", 0);

    if (n == 1) {
        printf("\n");
        return;
    }

    printf("1 ");

    for (i = 2; i <= n; i++) {
        next = a + b;
        printf("%d ", next);
        a = b;
        b = next;
    }

    printf("\n");
}

int main() {
    int numero;

    // Input dell'utente
    printf("Inserisci la posizione nella successione di Fibonacci: ");
    scanf("%d", &numero);

    // Calcolo e stampa della successione
    fibonacci(numero);

    return 0;
}
```

1) Riconoscimento Carattere- Calcolo con Funzione

Questo programma in C utilizza una funzione **carattere** per classificare un carattere inserito dall'utente. La funzione esamina il carattere attraverso un'istruzione **switch** e stampa una descrizione in base alla sua categoria: parentesi, vocali italiane (sia minuscole che maiuscole), vocali maiuscole non italiane, consonanti italiane, o altri caratteri. La gestione delle condizioni è eseguita con specifici casi per ogni tipo di carattere, e se il carattere non rientra in nessuna delle categorie definite, viene classificato come "Altro carattere".



```
#include <stdio.h>

// Funzione che classifica il carattere inserito
void carattere(int carattere_1) {
    switch (carattere_1) {
        // Condizioni per parentesi
        case '(':
        case '[':
        case '{':
            printf("Parentesi aperta\n");
            break;
        case ')':
        case ']':
        case '}':
            printf("Parentesi chiusa\n");
            break;

        // Condizioni per vocali italiane minuscole
        case 'a':
        case 'e':
        case 'i':
        case 'o':
        case 'u':
            printf("Vocale italiana minuscola\n");
            break;

        // Condizioni per vocali italiane maiuscole
        case 'A':
        case 'E':
        case 'I':
        case 'O':
        case 'U':
            printf("Vocale italiana maiuscola\n");
            break;

        // Condizioni per vocali maiuscole non italiane
        case 'J':
        case 'K':
        case 'W':
        case 'X':
        case 'Y':
            printf("Carattere alfabetico maiuscolo non appartenente all'alfabeto italiano\n");
            break;

        // Consonanti italiane maiuscole
        default:
            if (carattere_1 >= 'A' && carattere_1 <= 'Z') {
                printf("Consonante italiana maiuscola\n");
            } else {
                printf("Altro carattere\n"); // Altri caratteri
            }
            break;
    }
}
```

2) Riconoscimento Carattere- Calcolo con Funzione

```
int main() {
    char carattere_1;
    // Input dell'utente
    printf("Inserisci un carattere: ");
    carattere_1 = getchar(); // Legge un carattere

    // Chiamata alla funzione per classificare il carattere
    carattere(carattere_1);

    return 0;
}
```

Note Legali e Riconoscimenti

Copyright © 2025

Daniel Pio D'Ambrosio, Samuel Esposito, Silvio De Luca, Luigi Bevilacqua, Emilio Marturano.

Tutti i diritti riservati.

Nessuna parte di questo libro può essere riprodotta, archiviata in un sistema di recupero o trasmessa in qualsiasi forma o con qualsiasi mezzo – elettronico, meccanico, fotocopia, registrazione o altro – senza il permesso scritto degli autori, tranne nei casi previsti dalla legge sul diritto d'autore.

Prima edizione: 2025

L'elenco degli autori è riportato in ordine alfabetico e non riflette alcuna gerarchia o importanza.

Riconoscimenti

Questo libro è il frutto di un lavoro di gruppo realizzato con impegno e dedizione dagli studenti della classe 3G Informatica sotto la supervisione del professore Emilio Malizia.

- Daniel Pio D'Ambrosio: ideazione grafica, organizzazione del layout e finalizzazione del documento.
- Samuel Esposito e Silvio De Luca: selezione e raccolta degli esercizi svolti in classe.
- Luigi Bevilacqua e Emilio Marturano: revisione del documento.

Abbiamo creato questo materiale con l'obiettivo di fornire un utile strumento di studio e approfondimento, raccogliendo gli esercizi affrontati durante l'anno scolastico per facilitare il ripasso e la comprensione degli argomenti trattati.

Versione: 0.0.3