TROVA L’ERRORE

#include <stdio.h>

int main() {

int a, b;

printf("Inserisci due numeri: ");

scanf("%d, %d", &a, &b);

int somma = a + b;

printf("La somma è: %d\n", somma);

return 0;

}

Troppe virgole

scanf("%d %d", &a, &b);

#include <stdio.h>

int main() {

int numeri[5];

for (int i = 1; i <= 5; i++) {

printf("Inserisci numero %d: ", i);

scanf("%d", &numeri[i]);

}

printf("Primo numero: %d\n", numeri[0]);

return 0;

}

L’array parte da 0

for (int i = 0; i < 5; i++) {

printf("Inserisci numero %d: ", i + 1);

scanf("%d", &numeri[i]);

}

#include <stdio.h>

int main() {

int n, i, somma = 0;

printf("Inserisci un numero: ");

scanf("%d", &n);

for (i = 1; i <= n; i++); {

somma += i;

}

printf("La somma dei numeri da 1 a %d è: %d\n", n, somma);

return 0;

}

; dopo il for

for (i = 1; i <= n; i++); // <-- questo ';' termina subito il ciclo!

#include <stdio.h>

int main() {

int numeri[5] = {2, 4, 6, 8, 10};

int max = 0;

for (int i = 0; i < 5; i++) {

if (numeri[i] > max);

max = numeri[i];

}

printf("Il valore massimo è: %d\n", max);

return 0;

}

; dopo if

if (numeri[i] > max); // <-- questo ';' fa sì che il blocco dopo si esegua sempre

max = numeri[i]; // eseguito ogni volta → max finisce con l'ultimo valore dell'array

#include <stdio.h>

int fattoriale(int n) {

if (n == 0)

return 0;

else

return n \* fattoriale(n - 1);

}

int main() {

int n;

printf("Inserisci un numero: ");

scanf("%d", &n);

printf("Fattoriale di %d = %d\n", n, fattoriale(n));

return 0;

}

Il caso base del fattoriale è **sbagliato**: fattoriale(0) deve restituire 1, non 0.  
Con l’errore attuale, qualunque chiamata ricorsiva restituisce **0**.

int fattoriale(int n) {

if (n == 0)

return 1;

else

return n \* fattoriale(n - 1);

}

SCRIVI L BINARIO

int a = 5;

// binario: 00000101

int b = 12;

// binario: 00001100

int c = 255;

// binario:0 11111111

int d = 1024;

// binario: 010000000000

int e = -1;

// binario: 11111111

int f = -5;

// binario: 11111011

int g = -128;

// binario: 10000000

unsigned char uc = 200;

// binario: 11001000

unsigned int ui = 1023;

// binario: 00000011 11111111

short s = 3;

00000011

// binario (16 bit): 000000000000011

long l =2;

// binario (32 bit): 00000000 00000000 00000000 00000010