

**LABORATORIO DI PROGRAMMAZIONE**  
**CORSO DI LAUREA IN SICUREZZA INFORMATICA**  
**UNIVERSITÀ DEGLI STUDI DI MILANO**  
**2025-2026**

INDICE

<b>Parte 1. Una rudimentale calcolatrice</b>	<b>3</b>
Esercizio 1	3
<i>Calcolatrice usa e getta</i>	3
Tempo: 20 min.	3
Esercizio 2	3
<i>Calcolatrice usa e getta, gestione degli errori</i>	3
Tempo: 10 min.	3
Esercizio 3	3
<i>Calcolatrice riutilizzabile</i>	3
Tempo: 15 min.	3
Esercizio 4	3
<i>Calcolatrice riutilizzabile con pulsante Off</i>	3
Tempo: 10 min.	3
 <b>Parte 2. Numeri Primi</b>	 <b>4</b>
Esercizio 5	4
<i>Trova l'errore</i>	4
Tempo: 30 min.	4
 <b>Parte 3. Il M.C.D. con le divisioni: Euclide, Fibonacci e Lamé</b>	 <b>5</b>
Esercizio 6	5
<i>L'algoritmo euclideo con le divisioni</i>	5
Tempo: 20 min.	5
Esercizio 7	5
<i>L'algoritmo euclideo: input dell'utente</i>	5
Tempo: 10 min.	5
Esercizio 8	5
<i>L'algoritmo euclideo: validazione dell'input</i>	5
Tempo: 10 min.	6
Esercizio 9	6
<i>L'algoritmo euclideo: conteggio delle divisioni</i>	6
Tempo: 10 min.	6
Esercizio 10	6
<i>La successione di Fibonacci (versione iterativa)</i>	6

---

Tratti dagli esercizi del corso del prof. Vincenzo Marra.  
Ultima revisione: 13 ottobre 2025.

Tempo: 15 min.	6
Esercizio 11	7
<i>Verifica sperimentale della proposizione di Lamé</i>	7
Tempo: 15 min.	7
<b>Parte 4. Selezioni multiple annidate</b>	8
Esercizio 12	8
<i>Il signor Pignolino</i>	8
Tempo: 35 min.	8
Esercizio 13	9
<i>Il signor Pignolino interrogato con insistenza</i>	9
Tempo: 15 min.	9

**Parte 1. Una rudimentale calcolatrice****ESERCIZIO 1***Calcolatrice usa e getta.*

Tempo: 20 min.

Scrivete un programma secondo le specifiche seguenti.

- (1) All'avvio, il programma chiede all'utente di inserire due valori reali (tipo `double`).
- (2) Acquisiti i due operandi, il programma chiede all'utente se voglia eseguire una somma, una sottrazione, una moltiplicazione, o una divisione.
- (3) L'utente inserisce uno fra gli interi 1, 2, 3, e 4 per indicare la sua scelta.
- (4) Acquisita la scelta dell'utente, il programma esegue l'operazione richiesta, visualizza il risultato, e termina.

*Osservate:* l'analisi dei dati inseriti dall'utente si basa sul costrutto selezione. È necessario usare il costrutto iterazione in questo programma?

**ESERCIZIO 2***Calcolatrice usa e getta, gestione degli errori.*

Tempo: 10 min.

Modificate la calcolatrice scritta per l'Esercizio 1 di modo che:

- (1) Se la scelta dell'utente non è uno degli interi 1, 2, 3, e 4, il programma termina visualizzando un appropriato messaggio d'errore.
- (2) Se l'utente sceglie di eseguire una divisione, ma inserisce valore zero per il divisore, il programma termina visualizzando un appropriato messaggio d'errore.

**ESERCIZIO 3***Calcolatrice riutilizzabile.*

Tempo: 15 min.

Modificate la calcolatrice con gestione degli errori scritta per l'Esercizio 2 di modo che, in caso di visualizzazione di un messaggio d'errore, il programma riprenda l'esecuzione dall'inizio, proponendo all'utente la scelta dell'operazione da compiere. Analogamente, fate sì che quando una operazione è stata eseguita con successo, il programma ritorni all'inizio.

*Osservate:* “riutilizzare la calcolatrice” vorrà quindi dire “ripetere (l'esecuzione di un brano di codice) tramite iterazione”

(Suggerimento. Usate un'istruzione `while` con condizione sempre verificata per realizzare un ciclo infinito.)

La versione più semplice di una tale iterazione è:

```
while (1)
    istruzione
```

**ESERCIZIO 4***Calcolatrice riutilizzabile con pulsante Off.*

Tempo: 10 min.

Modificate la calcolatrice riutilizzabile scritta per l'Esercizio 3 di modo che l'utente abbia anche la scelta (oltre a 1–4) di uscire dal programma.

## Parte 2. Numeri Primi

```
----- primiErrori.c -----
1  #include <stdio.h>
2
3  int main() {
4  int i,j;
5  int n;
6  int flag = 0;
7  do {
8  printf("Inserisci una cifra massima per calcolare i numeri primi: ");
9  scanf("%d", &n);
10 } while (n<=0);
11
12 printf("Numeri primi fino a %d: 2 ", n);
13 for (i = 3; i <= n; i+=2); {
14 for (j = 2; j * j <= i; j++)
15 if (n % j == 0)
16 flag = 0; // Il numero non è primo se è divisibile per i
17
18 if (flag) {
19 printf("%d ", i);
20 }
21 }
22
23 printf("\n");
24
25 return 0;
26 }
```

FIGURA 1. Un programma che vorrebbe calcolare i numeri primi fino a  $n$ .

### ESERCIZIO 5

*Trova l'errore.*

Tempo: 30 min.

In Figura 1 trovate un programma che *dovrebbe* calcolare i numeri primi fino ad  $n$ , con  $n$  inserito dall'utente. Purtroppo, nello scriverlo, il programmatore è stato disattento e ha commesso degli errori, almeno 3. Inoltre il codice è scritto male, e questo rende il compito di trovare l'errore più difficile. Sistemate il programma in modo tale da renderlo leggibile. Cercate di correggere il programma e farlo funzionare.

**Parte 3. Il M.C.D. con le divisioni: Euclide, Fibonacci e Lamé**

FIGURA 2. Una statua di Leonardo Pisano Fibonacci.

**ESERCIZIO 6**

*L'algoritmo euclideo con le divisioni.*

Tempo: 20 min.

La Figura 3 mostra lo pseudocodice dell'algoritmo euclideo basato sulle divisioni successive. Scrivete il codice C corrispondente, assumendo che **a** e **b** siano variabili intere il cui valore è impostato all'inizio del programma tramite istruzioni di assegnamento; per esempio, `int a=100;` e `int b=46;`.

```
Input : a, b > 0
Output : m.c.d.(a, b)
while (b ≠ 0)
    aux := b
    b := a mod b
    a := aux
return a
```

FIGURA 3. L'algoritmo euclideo delle divisioni successive in pseudocodice.

**ESERCIZIO 7**

*L'algoritmo euclideo: input dell'utente.*

Tempo: 10 min.

Modificate la vostra soluzione all'Esercizio 6 in modo che sia l'utente a inserire da terminale i valori delle variabili **a** e **b**, all'inizio dell'esecuzione.

**ESERCIZIO 8**

*L'algoritmo euclideo: validazione dell'input.*

Tempo: 10 min.

Modificate la vostra soluzione all'Esercizio 7 in modo che il programma si assicuri che i valori di **a** e **b** inseriti dall'utente soddisfino le precondizioni: **a, b > 0**. Nel caso ciò non avvenga, visualizzate un appropriato messaggio d'errore per l'utente e terminate immediatamente l'esecuzione con l'istruzione **return -1**.

Come abbiamo già visto, restituire il valore **-1** segnala convenzionalmente una condizione eccezionale di terminazione.

#### ESERCIZIO 9

*L'algoritmo euclideo: conteggio delle divisioni.*

Tempo: 10 min.

Modificate la vostra soluzione all'Esercizio 8 in modo che, al termine del calcolo del M.C.D., il programma visualizzi il numero di operazioni di divisione che ha eseguito. Usate una variabile intera per tenere traccia del numero di divisioni. Incrementate di una unità il valore della variabile ad ogni successiva divisione.

#### ESERCIZIO 10

*La successione di Fibonacci (versione iterativa).*

Tempo: 15 min.

La Figura 4 mostra il sorgente parziale di un programma in C che calcola l'*n*-esimo termine della successione di Fibonacci. Manca una sola riga di codice, come indicato in figura. Aggiungetela, compilate ed eseguite il programma, ed assicuratevi con qualche esperimento del suo corretto funzionamento.

#### La successione di Fibonacci

Si consideri la successione di numeri naturali induttivamente definita da:

$$\begin{aligned} F_1 &:= 1, \\ F_2 &:= 1, \\ F_n &:= F_{n-1} + F_{n-2} \text{ per } n \geq 3. \end{aligned} \quad (*)$$

La successione  $(*)$  è nota come *successione di Fibonacci*.<sup>a</sup> In Tabella 1 sono tabulati i primi termini della successione di Fibonacci.

<sup>a</sup>Dal nome di Leonardo Pisano Fibonacci (ca. 1170–ca. 1250), così detto perché figlio del mercante pisano Guglielmo dei Bonacci. Leonardo fu un importante matematico medievale. Nel 1202 pubblicò il *Liber Abaci*, che svolse un ruolo fondamentale nella diffusione in Occidente del sistema di numerazione indiano basato sulle cifre 0–9, comunemente dette “arabe”. La successione di Fibonacci compare in un esempio del *Liber Abaci*.

<i>n</i>	1	2	3	4	5	6	7	8	9	10	11	12	13
<i>F<sub>n</sub></i>	1	1	2	3	5	8	13	21	34	55	89	144	233

TABELLA 1. I primi termini della successione di Fibonacci

$a \setminus b$	1	2	3	4	5	6	7	8
1	1							
2	1	1						
3	1	2	1					
4	1	1	2	1				
5	1	2	3	2	1			
6	1	1	1	2	2	1		
7	1	2	2	3	3	2	1	
8	1	1	3	1	4	2	2	1

TABELLA 2. Il numero di divisioni eseguito dall'algoritmo euclideo della Figura 3 per  $a \geq b > 0$ .

---

```

1  #include <stdio.h>
2
3  int main(void)
4  {
5      int n;          //il programma calcola F(n)
6
7      do
8      {
9          printf("Inserisci un intero positivo: ");
10         scanf("%d",&n);
11     } while ( (n<=0) ); //sono ammessi solo interi positivi
12
13     printf("F(%d)=",n);
14
15     int i;           //contatore del ciclo for
16     int f_i=1,f_prec=1; //valori iniziali F(2)=F(1)=1
17
18     for (i=3;i<=n;i++) //il ciclo comincia da i=3
19     {
20         int aux=f_i; //variabile ausiliaria aux
21         ...;         //somma ad f_i il valore di f_prec
22         f_prec=aux;  //f_prec = f_i prima dell'iterazione
23     }
24     printf("%d\n",f_i);
25 }

```

---

FIGURA 4. Una implementazione iterativa in C della successione di Fibonacci. L'implementazione è completa, eccezion fatta per la riga 21.

#### ESERCIZIO 11

*Verifica sperimentale della proposizione di Lamé.*

Tempo: 15 min.

Il matematico francese Gabriel Lamé (1795–1870) dimostrò il fatto seguente.

### Proposizione

Sia  $n > 0$  un intero, e siano  $a > b > 0$  interi tali che l'algoritmo euclideo della Figura 3 applicato ad  $a$  e  $b$  esegua esattamente  $n$  divisioni. Se  $a$  è il minimo intero che soddisfa tali condizioni, si ha

$$a = F_{n+2}$$

$$b = F_{n+1},$$

dove  $F_n$  denota la successione di Fibonacci.

Usando i programmi che avete sviluppato negli Esercizi 10 e 9, verificate la proposizione di Lamé in qualche caso concreto. Per esempio, la Tabella 2 riporta il numero di divisioni eseguito dall'algoritmo euclideo per  $8 \geq a \geq b > 0$ . La vostra soluzione dell'Esercizio 9 dovrebbe produrre valori identici. Ora notate che la prima coppia  $(a, b)$  che richieda 4 divisioni è  $(8, 5)$  — e infatti, dalla Tabella 1 si vede che  $F_6 = 8$  ed  $F_5 = 5$ , in accordo con la proposizione di Lamé. Provate con almeno altri due valori di  $n$ , uno più piccolo e uno più grande di 4.

## Parte 4. Selezioni multiple annidate

### ESERCIZIO 12

*Il signor Pignolino.*

Tempo: 35 min.

Il signor Pignolino è pensionato. Egli esce di casa ogni mattina, ma solo a condizione che non piovga e che le previsioni metereologiche per la giornata siano buone. Nel caso in cui decida di uscire, se il giorno del mese è pari (ad esempio, è il 20 di febbraio, o di marzo, etc.), si reca ai giardini pubblici, portando con sé un libro; altrimenti, si reca al Caffé, portandosi dietro il suo diario personale invece di un libro. Quanto detto vale a meno che non sia domenica, giorno in cui il signor Pignolino va invariabilmente a far visita alla sua cara amica signora Precisina, avendo l'accortezza di prendere un ombrello nel caso in cui piovga o le condizioni metereologiche non siano buone, o un parasole altrimenti.

Si scriva un programma che rivolga al signor Pignolino un certo numero di domande, tenga traccia delle sue risposte per mezzo di variabili intere, e produca in uscita una descrizione accurata del comportamento del signor Pignolino per la giornata. Il signor Pignolino è impersonato dall'utente. Si assuma che le risposte del signor Pignolino siano solo del tipo Sì/No, codificate dai caratteri **S** e **N**, rispettivamente. Se la risposta a una qualsiasi delle domande non è **S** o **N**, il programma termina con un messaggio d'errore. Se invece tutte le risposte del signor Pignolino sono state acquisite correttamente, il programma deve produrre in uscita, prima di terminare, una e una sola fra le frasi seguenti, quella che riflette esattamente le risposte fornite da Pignolino:

- Il signor Pignolino oggi non e' uscito.



- Il signor Pignolino oggi e' uscito per recarsi ai giardini pubblici. Ha portato con se' un libro.
- Il signor Pignolino oggi e' uscito per recarsi al Caffe'. Ha portato con se' il suo diario.
- Il signor Pignolino oggi e' uscito per recarsi dalla signora Precisina. Ha portato con se' un ombrello.
- Il signor Pignolino oggi e' uscito per recarsi dalla signora Precisina. Ha portato con se' un parasole.

## ESERCIZIO 13

*Il signor Pignolino interrogato con insistenza.*

Tempo: 15 min.

Modificate il programma che avete scritto per risolvere l'Esercizio 12 di modo che, nel caso in cui Pignolino risponda a un domanda con un carattere diverso da S e N, il programma, invece di terminare con errore, ponga nuovamente la stessa domanda fino ad ottenere una risposta nel formato corretto. Per il resto il comportamento del programma è invariato.

DIPARTIMENTO DI INFORMATICA, UNIVERSITÀ DEGLI STUDI DI MILANO,