

Olimpiadi Italiane di Informatica 2024/2025 - Selezione scolastica

Cognome

Nome

Classe

Sezione

Data di nascita

Codice prova

Ciao! Ecco le informazioni essenziali sulla prova che stai per svolgere. Ti consigliamo di leggerle attentamente.

Regole di base

Ti è permesso:

- avere a disposizione una calcolatrice – va bene qualunque tipo di calcolatrice, comprese quelle grafiche e la calcolatrice di sistema dell'eventuale dispositivo elettronico che stai usando (PC o dispositivo mobile);
- avere con te dei fogli bianchi (eventualmente a righe o quadretti);
- avere con te del materiale di cancelleria (penne, matite, gomma, etc.);
- andare in bagno in qualunque momento – in tal caso, dovrai lasciare nell'aula dispositivi elettronici, il testo della prova, e altro materiale cartaceo;
- comunicare con il docente sorvegliante in caso di problemi tecnici.

Non ti è permesso:

- navigare su internet, se non all'indirizzo del sito della prova <https://scolastiche2024.olinfo.it>;
- comunicare con i tuoi compagni;
- comunicare con il docente sorvegliante sul contenuto della prova;
- diffondere il testo della prova, o parte di esso, prima delle 20:00 del giorno della prova (12 dicembre).

Formato della prova

La prova contiene *10 problemi* da risolvere in *90 minuti*, ed è divisa in due parti:

- sette problemi di **pensiero logico-algoritmico**, e
- tre problemi di interpretazione di **procedimenti procedurali** come programmi a blocchi.

In entrambe le parti i problemi sono **in ordine casuale**, e quindi non in ordine di difficoltà. Ogni problema comprende *due domande*, valutate separatamente. La prima domanda è **sempre più semplice**, e può aiutare a rispondere alla seconda domanda (che in genere è da considerarsi difficile).

Punteggio

Le domande possono essere a scelta multipla oppure a risposta aperta numerica. Tutte le domande a *scelta multipla* presentano 5 opzioni, di cui **solo una** è corretta. Il punteggio assegnato per tali domande è:

- 5 punti per una risposta *corretta*;
- 1 punto per una risposta *in bianco*;
- 0 punti per una risposta *errata*.

Ogni *domanda aperta numerica* richiede come risposta un numero intero (eventualmente negativo). Il punteggio assegnato per tali domande è:

- 5 punti per una risposta *corretta*;
- 0 punti per una risposta *errata o in bianco*.

Quesiti di programmazione e pseudocodice

I quesiti di programmazione presentano semplici programmi scritti in *pseudocodice*. Qui sotto puoi trovare un riassunto della sintassi dello pseudocodice, oppure [scarica la guida completa](#) in PDF.

Pseudocodice	Descrizione	Esempio
■ Variabili e tipi		
variable <i>i</i> : <i>integer</i>	Dichiarazione della variabile di tipo intero chiamata <i>i</i>	
variable <i>arr</i> : <i>integer[]</i>	Dichiarazione di una variabile di tipo array di interi chiamata <i>arr</i>	
<i>{var}</i> ← <i>{espr}</i>	Assegnamento del valore dell'espressione <i>{espr}</i> alla variabile <i>{var}</i>	<i>i</i> ← 1 <i>a</i> ← 3 × <i>i</i> + 5 <i>arr</i> ← [3, 5/ <i>a</i> , 2]
<i>arr</i> [<i>{espr}</i>]	Variabile corrispondente all'elemento dell'array <i>arr</i> di indice <i>{espr}</i>	<i>a</i> ← <i>arr</i> [3] + 1 <i>arr</i> [<i>x</i> + 1] ← 2
(<i>a</i> , <i>b</i>) ← (<i>b</i> , <i>a</i>)	Scambio del valore delle variabili <i>a</i> e <i>b</i>	
■ Operatori		
+, −, ×, /, mod	Aritmetica: addizione, sottrazione (o negazione), moltiplicazione, divisione intera, resto della divisione intera (modulo)	<i>a</i> + <i>b</i> − <i>a</i> <i>i</i> ← <i>a mod</i> 10
==, ≠, <, ≤, >, ≥	Confronto: uguale, diverso, minore, minore o uguale, maggiore, maggiore o uguale	<i>a</i> == 7 2 × (<i>x</i> + 1) ≤ <i>y</i>
and , or , not	Operatori logici: e, o, non	<i>a</i> > <i>b</i> and <i>b</i> ≠ −1 not (<i>a</i> > 2 or <i>a</i> == 0)
■ Strutture di controllo		
if {condizione} then {corpo if} else {corpo else} end if	Struttura condizionale if ... else : se {condizione} è vera viene eseguito {corpo if}, altrimenti viene eseguito {corpo else}. La parte else può essere omessa	if <i>n mod</i> 2 == 0 then <i>i</i> ← 0 else <i>i</i> ← <i>n</i> − 1 end if
while {condizione} do {corpo} end while	Ciclo while : il blocco {corpo} viene ripetuto fintanto che {condizione} è vera	while <i>i</i> < <i>n</i> do <i>sum</i> ← <i>sum</i> + <i>i</i> <i>i</i> ← <i>i</i> + 7 end while
for {indice} in {intervallo} do {corpo} end for	Ciclo for : il blocco {corpo} viene eseguito mentre la variabile {indice} itera sui valori in {intervallo}, specificato come [<i>a</i> ... <i>b</i>], che significa "tutti i numeri da <i>a</i> (incluso) fino a <i>b</i> (escluso)"	for <i>i</i> in [0 ... <i>n</i>) do <i>arr</i> [<i>i</i>] ← −1 end for (assegna −1 a tutti gli elementi di un array <i>arr</i> di lunghezza <i>n</i>)
■ Funzioni		
function <i>fun</i> (<i>var1</i> : <i>tipo1</i> , <i>var2</i> : <i>tipo2</i> , ...) → <i>ritorno</i> {corpo} end function	Funzione con parametri <i>var1</i> , <i>var2</i> , etc. Il tipo di ritorno → <i>ritorno</i> può essere omesso. Il valore viene restituito tramite la parola chiave return	function <i>add</i> (<i>a</i> : <i>integer</i> , <i>b</i> : <i>integer</i>) → <i>integer</i> return <i>a</i> + <i>b</i> end function
<i>fun</i> () <i>fun</i> (<i>arg1</i> , <i>arg2</i> , ...)	Chiamata alla funzione <i>fun</i> (rispettivamente senza argomenti e con argomenti). La funzione output stampa il valore di una variabile oppure una stringa fissata. Le funzioni min e max restituiscono risp. il minimo e il massimo di due interi	return <i>add</i> (<i>a</i> , <i>b</i>) <i>m</i> ← <i>very_big_integer</i> () output (<i>x</i>) output ("string") <i>min</i> (<i>a</i> , <i>b</i>) <i>max</i> (<i>a</i> , <i>b</i>)

Sezione 1: Esercizi logico-algoritmici

Domanda 1.1

Hai due numeri a e b , puoi fare due tipi di operazioni:

- diminuire a di 10^k , per un qualche k intero non negativo.
- diminuire a di b .

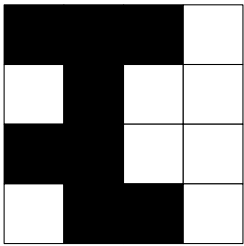
Se $a = 5075$ e $b = 1893$, quante mosse servono al minimo per rendere $a = 0$?

Domanda 1.2

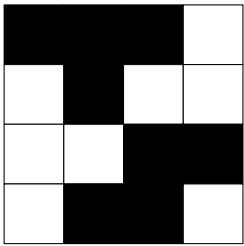
Se invece $a = 5075$ e $b = 18$, quante mosse servono al minimo per rendere $a = 0$?

Domanda 2.1

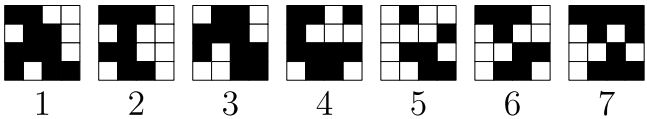
Mentre andava a scuola, Francesco è inciampato su una griglia magica 4x4 formata da caselle bianche e nere, rappresentata in figura.



Provando a giocare si è accorto che può invertire tutte le celle in una riga, facendo sì che le caselle nere diventino bianche e viceversa. Ad esempio, se applica l'operazione alla terza riga ottiene la griglia seguente.










Quali delle seguenti griglie può ottenere se applica questa operazione in modo opportuno? Inserire la lista di griglie che può ottenere, in ordine e non separate da spazi. Per esempio se può ottenere le griglie 2, 4 e 5, inserire 245.



Domanda 2.2

Il giorno dopo Francesco si accorge che può anche invertire una colonna della griglia. Sapendo quindi che può invertire sia righe che colonne, quali delle seguenti griglie può ottenere? Inserire la risposta nello stesso formato della domanda precedente.

1234567

Domanda 3.1

Carlo la cavalletta ama saltare sulla retta dei numeri interi. Inizialmente si trova in 0. Successivamente, per ogni $k=0, 1, 2, 3, \dots, 16$ sceglie se saltare in avanti di 2^k unità, saltare indietro di 2^k unità oppure stare fermo. In quanti interi x tali che $-1024 \leq x \leq 2024$ può trovarsi **alla fine** di questo processo?
Attenzione: prima di finire, Carlo può anche trovarsi in posizioni x maggiori di 2024 o minori di -1024 .

- ☐ A) 3048

☐ B) 3049

☐ C) 1016

☐ D) 1523

☐ E) 1524

Domanda 3.2

Se invece Carlo non potesse mai stare fermo, quanti ne potrebbe raggiungere?

- ☐ A) 1016

☐ B) 1523

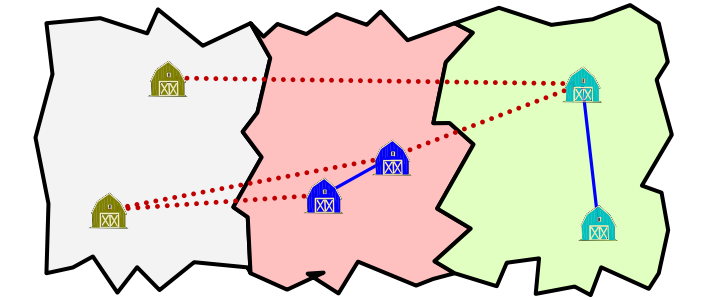
☐ C) 3048

☐ D) 3049

☐ E) 1524

Domanda 4.1

A Olinfolandia ci sono 100 stati e ognuno di essi ha al suo interno 10 città. Valerio è stato incaricato di costruire delle strade **bidirezionali** per connettere le città: può costruire una strada che connette due città dello stesso stato con costo 1 e città di stati diversi con costo 2. Per esempio, se ci fossero 3 stati con 2 città ciascuno, si potrebbero collegare come in figura. Questo piano costa $2 \times 4 + 1 \times 2 = 10$:

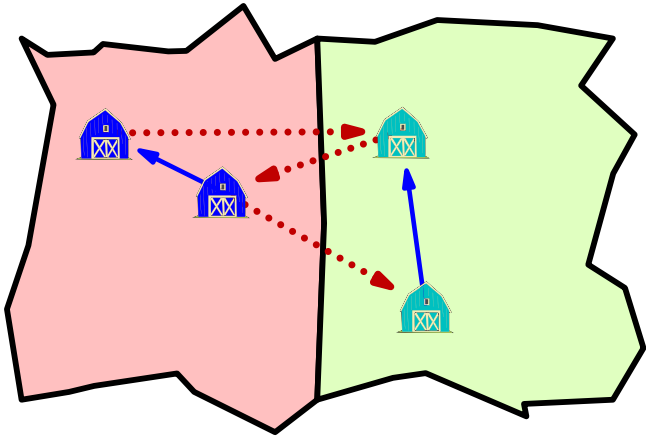


In questo esempio, ci sono anche altri piani validi, ciascuno con un costo di 7 o più.

Ricordando che a Olinfolandia ci sono 100 stati con 10 città, se Valerio costruisce strade **bidirezionali**, quanto deve spendere al minimo per fare in modo che da ogni città si possa raggiungere ogni altra città?

Domanda 4.2

Mettiamo ora che le strade che Valerio costruisce siano **unidirezionali**, come in questo esempio:



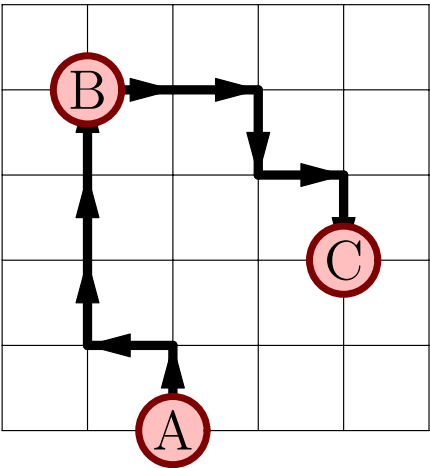
Questo piano costa $3 \times 2 + 2 \times 1 = 8$, e consente di poter andare da ogni città ad ogni altra rispettando i versi delle strade. Come prima, ci sono anche altri piani validi, ciascuno con un costo di 6 o più.

Ricordando che a *Olinfolandia* ci sono 100 stati con 10 città, se Valerio costruisce strade **unidirezionali**, quanto deve spendere al minimo per fare in modo che da ogni città si possa raggiungere ogni altra città?

Domanda 5.1

Valerio sceglie tre punti **distinti** A , B e C su una griglia 1932×1932 . Davide deve scegliere il percorso più corto possibile che parte dal punto A , attraversa il punto B e termina nel punto C , composto solo da righe orizzontali e verticali.

Ad esempio, questo è uno dei percorsi più brevi (lungo 10 unità) che potrebbe scegliere Davide su una griglia 5×5 , dati i punti A , B e C :



Valerio è malvagio, e sceglie i punti A , B e C per rendere il percorso il più lungo possibile. Qual'è la lunghezza del percorso più lungo che potrebbe dover scegliere Davide su una griglia 1932×1932 ?

- ☐ A) 3864

☐ B) 5152

☐ C) 7727

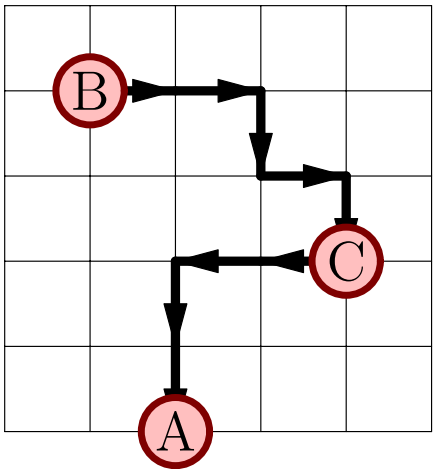
☐ D) 7728

☐ E) 4830

Domanda 5.2

Valerio ora permette a Davide di scegliere l'ordine in cui attraversare i punti, che lo sceglie in modo che il percorso risultante sia il più corto possibile.

Ad esempio, sulla griglia precedente, questo è uno dei percorsi più brevi (lungo 9 unità) che potrebbe scegliere Davide:

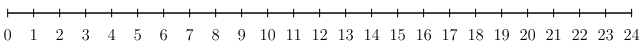


Qual è la lunghezza del percorso più lungo che potrebbe dover scegliere Davide su una griglia 1932 x 1932, scegliendo in modo ottimale l'ordine in cui attraversare i punti?

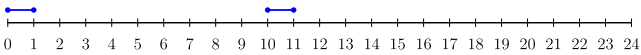
- ☐ A) 7727
- ☐ B) 7728
- ☐ C) 4830
- ☐ D) 5152
- ☐ E) 3864

Domanda 6.1

Hai davanti a te una linea di coordinate da 0 a 24.



Vuoi disegnare 2 segmenti lunghi 1 (a coordinate intere) su questa linea, in modo che non si sovrappongano (ma possono toccarsi alle estremità). Ad esempio questo è un modo valido di scegliere i segmenti.

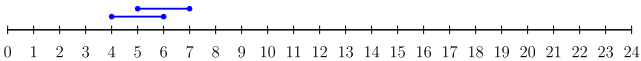


In quanti modi puoi farlo?

- ☐ A) 288
- ☐ B) 276
- ☐ C) 300
- ☐ D) 576
- ☐ E) 48

Domanda 6.2

In quanti modi puoi disegnare 2 segmenti lunghi 2 (a coordinate intere) su questa linea, in modo che non si sovrappongano (ma possono toccarsi alle estremità)? Ad esempio questo è un modo **non** valido dato che i segmenti si sovrappongono.



- ☐ A) 288
- ☐ B) 48
- ☐ C) 300
- ☐ D) 231
- ☐ E) 210

Domanda 7.1

Elia è nel suo negozio di quaderni preferito. Davanti a sé ha una fila di quaderni, e ha assegnato ad ognuno di essi un valore in base a quanto gli piace. Vuole comprare alcuni di essi, ma ha deciso di non comprare 2 quaderni che si trovano vicini. Qual è la massima somma dei valori dei quaderni che può comprare? Per esempio se i valori fossero questi:

5	10	0	2	6
---	----	---	---	---

la somma massima ottenibile sarebbe 16.

Rispondi considerando che questa è la lista dei valori dei quaderni, in ordine da sinistra a destra.

0	0	0	1	1	1	0	1	0	1	1	1	1	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---

Domanda 7.2

Quale sarebbe invece la massima somma se i valori fossero questi?

10	9	0	18	19	6	2	6	3	17	14	12	2	3
----	---	---	----	----	---	---	---	---	----	----	----	---	---

Sezione 2: Esercizi di programmazione

Domanda 8.1

Considera la seguente funzione, che prende come parametri due interi **positivi** l , r , e restituisce un intero:

```
function calcola(l: integer, r: integer) → integer
  variable a: integer[]
  variable sum: integer

  for i in [0...r) do
    a[i] ← 0
  end for

  for i in [l...r) do
    variable j: integer
    j ← i
    while j ≥ 1 do
      a[j] ← 1
      j ← j / 2
    end while
  end for

  sum ← 0
  for i in [0...r) do
    sum ← sum + a[i]
  end for
  return sum
end function
```

Che valore viene restituito da `calcola(8, 13)`?

☐ A) 10

☐ B) 11

☐ C) 8

☐ D) 13

☐ E) 5

Domanda 8.2

Che valore viene restituito da `calcola(512, 875)`?

☐ A) 693

☐ B) 363

☐ C) 875

☐ D) 512

☐ E) 729

Domanda 9.1

Considera la seguente funzione, che prende come parametri un array di interi *a* e un intero *n* che indica la sua lunghezza.

```
function conta(a: integer[], n:integer) → integer
  variable ans: integer
  for i in [0...n-1) do
    if a[i] == a[i + 1] then
      ans ← ans + 1
    else
      ans ← ans + a[i + 1] - a[i]
    end if
  end for
  return ans
end function
```

Che valore viene restituito da `conta([6,6,0,4,8,7,7,6,4,7], 10)`? (l'array *a* è anche riportato sotto)

6	6	0	4	8	7	7	6	4	7
---	---	---	---	---	---	---	---	---	---

Domanda 9.2

Quanti valori distinti può ritornare la funzione `conta` se *n* vale 13 e l'array *a* è composto da 13 valori tra 0 e 9?

Domanda 10.1

Considera la seguente funzione, che prende come parametri un intero **positivo** n , e restituisce un intero:

```
function f(n: integer) → integer
    variable arr: integer[]
    variable i: integer
    variable sum: integer
    for i in [1..n) do
        arr[i] ← 0
    end for
    i ← 0
    while i < n do
        arr[i] ← i
        i ← i + 2
    end while
    i ← 0
    while i < n do
        arr[i] ← i
        i ← i + 3
    end while
    for i in [1..n) do
        arr[i] ← max(arr[i-1], arr[i])
    end for
    sum ← 0
    for i in [0..n) do
        sum ← sum + i - arr[i]
    end for
    return sum
end function
```

Cosa viene restituito da $f(9)$?

- ☐ A) 36 ☐ B) 4 ☐ C) 3
- ☐ D) 1 ☐ E) 9

Domanda 10.2

Cosa viene restituito da $f(912)$?

- ☐ A) 456 ☐ B) 912 ☐ C) 304
- ☐ D) 152 ☐ E) 415416

