AICA
Associazione Italiana per l'Informatica
ed il Calcolo Automatico

Ministero della
Pubblica Istruzione

OLIMPIADI ITALIANE DI INFORMATICA

Materiale didattico per la preparazione alla selezione scolastica (I° livello)

a cura del Comitato Olimpico

hanno collaborato:
G. Callegarin
W. Fasoli
A. Germini

Milano, ottobre 2008

Nota per i Docenti

1. Premessa

Le ***Olimpiadi Internazionali di Informatica***, che si sono svolte per la prima volta nel 1989 su iniziativa e con il patrocinio dell'Unesco, hanno l'obiettivo di mettere in competizione i giovani di maggior talento in Informatica di tutto il mondo, ma anche di promuovere l'amicizia tra studenti e docenti di diversa cultura.

Le Olimpiadi si svolgono ogni anno in paesi diversi (dalla Bulgaria all'Olanda, dal Sudafrica agli USA, dalla Cina al Messico) e vi partecipano oltre 80 nazioni, ciascuna con una squadra di 4 studenti di età non superiore a 20 anni. L'Italia ha iniziato la sua partecipazione nell'anno 2000 (Olimpiadi di Pechino)e da allora è sempre stata presente con risultati che si possono definire lusinghieri. Fino ad oggi la squadra italiana ha conquistato una medaglia d'oro, 6 medaglie d'argento e 12 medaglie di bronzo.

La partecipazione italiana alle Olimpiadi Internazionali è organizzata da un ***Comitato Olimpico*** composto da rappresentanti del Ministero della Pubblica Istruzione e dell'Associazione Italiana per l'Informatica ed il Calcolo Automatico (Aica). Il Comitato Olimpico è affiancato dal ***Gruppo di Allenatori Nazionali***, che ha il compito di curare la selezione della squadra italiana e la preparazione scientifica dei partecipanti. La selezione avviene in quattro fasi.

La prima è la ***selezione scolastica*** che si svolge in novembre/dicembre presso le scuole aderenti all'iniziativa (circa 500 con circa 15.000 studenti) ed è gestita localmente dagli insegnanti a cui vengono inviati i testi dei problemi via Internet. Lo studente deve risolvere semplici quesiti logico-matematici e di programmazione (in Pascal o C/C++ che sono i linguaggi di programmazione ufficiali della gara internazionale) senza l'uso del calcolatore.

Secondo la nuova procedura adottata nel 2008 dal Comitato Olimpico, i migliori classificati (un migliaio) vengono ammessi alla ***selezione territoriale*** che si svolge orientativamente nei successivi mesi di marzo o aprile. Agli studenti viene richiesto di progettare gli algoritmi risolutori di alcuni problemi (ancora di tipo logico-matematico) e di redigere i corrispondenti programmi (sempre in Pascal o C/C++). I testi dei problemi pervengono ai server delle scuole-sedi territoriali tramite un server centrale che li trasmette in forma criptata. Lo stesso server centrale provvede alla correzione automatica degli elaborati.

Gli studenti che superano la selezione territoriale (circa 80) partecipano l'anno scolastico successivo alla ***selezione nazionale*** denominata ***Gara Olimpica Nazionale.*** Si svolge orientativamente nel mese di febbraio in un'unica sede che cambia di anno in anno e presenta caratteristiche simili alla ***Gara Olimpica Internazionale*** anche se i problemi, pur essendo impegnativi, presentano difficoltà minori. A conclusione, vengono assegnate 5 medaglie d'oro, 10 di argento e 20 di bronzo.

I primi 15 classificati sono i ***probabili Olimpici*** e vengono ammessi alla quarta fase di selezione che prevede alcuni "stage" intensivi curati nei mesi successivi dal Gruppo di Allenatori Nazionali. Negli ultimi anni gli allenamenti si sono svolti presso l'Università di Pisa. Alla fine degli "stage"vengono scelti, solitamente nel mese di maggio, i quattro "atleti" titolari della squadra che parteciperà alle Olimpiadi Internazionali, oltre a due riserve. Titolari e riserve proseguono gli allenamenti, anche con il supporto di strumenti telematici, fino alla gara internazionale, generalmente collocata in agosto.

2. Finalità del materiale didattico

Le osservazioni e le riflessioni effettuate durante i sette anni in cui l'Italia ha partecipato alle gare hanno evidenziato la limitata preparazione in Informatica posseduta dagli studenti delle scuole secondarie italiane. Situazione del resto ben nota, poiché l'insegnamento di questa materia è presente soltanto in alcuni indirizzi degli Istituti Tecnici e presso alcuni Licei sperimentali.

I risultati soddisfacenti della nostra squadra olimpica sono quindi dovuti sia al forte interesse di molti giovani per l'Informatica (spesso veri e propri autodidatti) sia alla qualità degli allenamenti, ma soprattutto al talento dei membri della squadra olimpica che riescono ad acquisire in breve tempo le competenze di cui sono carenti. Vale la pena di sottolineare quanto accade alle Olimpiadi Internazionali: le nazioni che ottengono il maggior numero di medaglie d'oro e d'argento sono quelle dell'Estremo Oriente e dell'Europa orientale, ma l'Italia è tra le prime nell' Europa occidentale.

Per migliorare ulteriormente questi esiti, il Comitato Olimpico, d'intesa con il Ministero della Pubblica Istruzione, sta promovendo azioni volte ad accrescere il numero di scuole, e quindi degli studenti, che partecipano al primo livello di selezione. In questo quadro si inserisce l'iniziativa di produrre alcuni materiali didattici da mettere a disposizione dei docenti che, auspicabilmente, intendono organizzare attività preparatorie con cui aiutare gli studenti ad affrontare la gara. Il materiale potrebbe risultare utile anche per la preparazione degli studenti ammessi alle selezioni regionali, poiché contiene riferimenti ad una vasta gamma di concetti di base e alle tecniche per la progettazione di algoritmi, oltre che alla scrittura di programmi corretti ed efficienti.

3. Caratteristiche del materiale didattico

Il gruppo di lavoro incaricato di predisporre i materiali, di cui fanno parte anche docenti di informatica da anni impegnati nella ricerca didattica di questa disciplina, ha ritenuto di prendere le mosse dai test sottoposti agli studenti negli ultimi cinque anni ed ha proceduto come segue:

- ne ha esaminato i contenuti e definito le tipologie aggregandole per macro-aree e relative sotto-aree;
- ha elencato per ciascuna sotto-area le fondamentali abilità e conoscenze necessarie per poter comprendere e risolvere i problemi presentati;
- ha cercato, sempre per ciascuna sotto-area e con riferimento alle abilità e alle conoscenze ad essa associate, di segnalare una limitata ed essenziale bibliografia/sitografia commentata da utilizzare per aiutare gli studenti ad acquisire le indispensabili conoscenze di base (che in parte maggiore o minore potrebbero anche già possederle);
- ha individuato alcuni esempi di problemi riferibili ai vari contenuti delle sub-aree scegliendoli fra quelli assegnati nelle precedenti selezioni scolastiche;
- ha presentato la soluzione commentata dei problemi scelti.

Il risultato di tale lavoro si trova negli allegati che sono così connotati:

Allegato 1 - Quadro generale di riferimento
Contiene in forma tabellare la sintesi del materiale didattico. Nella prima colonna sono riportate le tipologie di problemi raggruppati in due macro-aree: area **logico-matematica** e area **informatica/ tecniche di programmazione.** La prima è suddivisa in sei sotto-aree e la seconda in quattro. Ovviamente gli argomenti elencati in ciascuna sotto-area e le stesse sotto-aree non esauriscono tutti gli argomenti oggetto delle domande che possono essere formulate in occasione di un test di

selezione Tuttavia, forniscono una buona copertura essendo il risultato di un'analisi dei problemi effettivamente posti nelle recenti selezioni.
Nella seconda colonna sono elencate le abilità o le conoscenze necessarie per rispondere correttamente ai problemi appartenenti alle sub-aree indicate nella prima colonna.
Nella terza colonna sono indicati i codici identificativi dei problemi riportati e commentati nell'Allegato 3. Ciascun codice è costituito da un numero progressivo, dall'anno in cui il problema è stato posto, dal numero d'ordine del problema nell'ambito del test di quell'anno e dal grado di difficoltà (F=facile, M=medio, D=difficile).

Allegato 2 – Bibliografia e sitografia
Contiene un breve commento relativo alle caratteristiche delle diverse tipologie di problemi seguito da alcune indicazioni su testi o siti nei quali è possibile trovare la trattazione di contenuti ed esercizi svolti riguardanti le tematiche presenti nei problemi scelti come esempi.

Allegato 3 - Problemi e relativa soluzione commentata
Contiene i testi dei problemi elencati nella terza colonna dell'allegato 1 e per ciascun problema riporta la risposta seguita da un commento.

4. Suggerimenti per l'uso del materiale didattico

Le Olimpiadi internazionali, ed in misura inferiore le Olimpiadi nazionali, le selezioni regionali e quelle scolastiche, sono gare che si svolgono in una situazione fortemente competitiva in cui i concorrenti hanno a disposizione un tempo limitato.
La correzione delle prove olimpiche, in particolare, avviene con strumenti automatici che non solo verificano la correttezza delle risposte sottoponendo ai programmi realizzati dai concorrenti un certo numero di casi-prova (senza entrare nell'analisi del programma realizzato), ma controllano anche, per escludere soluzioni banali ed inefficienti, i tempi di esecuzione che non devono superare certi limiti prefissati.
Gli esempi qui forniti si riferiscono al primo livello di selezione e quindi vanno visti come preparazione a prove relativamente semplici. Si suggerisce tuttavia ai docenti di praticare modalità didattiche tali da favorire il conseguimento di una preparazione funzionale a quanto serve per poter rispondere ai quesiti della gara. Tali modalità dovrebbero tendere a improntare la gestione dell'aula con attività basate su approcci laboratoriali, apprendimento cooperativo e coinvolgimento diretto. In altri termini, assumere in classe linee di condotta così connotate:

- partire sempre dai problemi per ricercarne la soluzione e discuterla risalendo poi agli aspetti concettuali o alle tecniche operative che ne costituiscono le fondamenta;
- favorire in ogni modo la discussione e il confronto fra gli studenti comparando e analizzando le diverse condotte risolutive adottate da ciascuno;
- creare frequentemente un "clima di gara" che spinga gli studenti ad assumere un atteggiamento competitivo;
- far acquisire la consapevolezza che la probabilità di successo è direttamente proporzionale al numero di problemi analizzati, svolti, confrontati e commentati.

Allegato 1 - Quadro generale di riferimento

	Tipologia di problemi	Abilità/conoscenze sottostanti	Codici identif. dei problemi
		Area Logico-Matematica	
A1	Problemi di ottimizzazione	- Problem solving - Elaborazione e confronto di strategie - Analisi dei casi possibili	1.2006.12.D 2.2005.1.M 3.2004.5.F
A2	Problemi di deduzione logica	- Calcolo proposizionale - Regole d'inferenza - Principio di non contraddizione - Proprietà dei quantificatori (esistenziale ed universale)	4.2003.1.D 5.2003.6.M 6.2007.4.F 7.2004.8.F
A3	Problemi di calcolo	- Proporzioni - Equazioni di 1° grado - Sostituzioni - Sistemi - Calcolo mentale	8.2006.11.D 9.2006.9.M 10.2006.7.F 11.2007.1.F
A4	Problemi di insiemistica e calcolo combinatorio	- Calcolo di cardinalità degli insiemi - Principio di inclusione ed esclusione - Disposizioni, combinazioni e permutazioni	12.2002.10.D 13.2002.11.D 14.2002.9.M 15.2006.3.F
A5	Problemi di geometria	- Intersezione e inclusione di piani e solidi - Superfici e volumi	16.2004.11.D 17.2004.10.M 18.2006.6.F
A6	Problemi sui sistemi di numerazione	- Rappresentazione posizionale di numeri - Calcoli in basi diverse da 10	19.2005.1.D 20.2004.12.M 21.2005.12.F
		Area Informatica / Tecniche di Programmazione	
E1	Concetti di base	- Variabili - Variabili strutturate (array, matrici) - Funzioni - Procedure - Passaggio dei parametri (riferimento, valore) - Campo di azione di un identificatore - Programmazione strutturata - Algoritmi	22.2002.8.D 23.2006.2.M 24.2005.6.F
E2	Esecuzione manuale di codice non ricorsivo	- Tecniche di tracing - Tecniche di debugging - Tavole di traccia	25.2006.5.D 26.2006.2.M 27.2003.1bis.F
E3	Esecuzione manuale di codice ricorsivo	- Ricorsione semplice - Mutua ricorsione - Tecniche di tracing per la ricorsione	28.2007.7.D 29.2006.8.M 30.2007.6.F
E4	Riconoscimento del problema risolto da una porzione di codice	- Processo di astrazione e generalizzazione - Riconoscimento della soluzione - Conoscenza degli algoritmi di base	31.2005.3.D 32.2003.7.M 33.2004.3.F

Allegato 2 - Bibliografia e sitografia

1. Problemi logico-matematici

Lo scopo sottinteso di questo tipo di problemi in una gara di Informatica è quello di misurare indirettamente due tipi di capacità:
a) la corretta interpretazione del testo di un problema;
b) la risoluzione rapida di un problema che richiede conoscenze di base piuttosto elementari, ma anche una certa creatività nell'individuarla velocemente.

I due tipi di abilità sono legati: una cattiva interpretazione vanifica gli esiti della seconda. D'altronde, per dare una buona interpretazione, sono richieste conoscenze che si ritrovano spesso in persone che hanno buone capacità risolutive. Queste abilità sono in realtà pertinenti anche ad altri tipi di competizioni scientifiche come le Olimpiadi di Fisica, di Chimica e, soprattutto, le Olimpiadi Matematica. Molti dei problemi che vengono assegnati alle Olimpiadi di Informatica potrebbero benissimo essere assegnati alle Olimpiadi di Matematica e viceversa. Quasi tutti assumono l'aspetto del "rompicapo" (in inglese "puzzle") come si può trovare anche nei settimanali di enigmistica o nella letteratura di Matematica ricreativa, nella quale il mondo anglosassone vanta una lunga tradizione.

La domanda da porsi è dunque: "Come si possono preparare i ragazzi a questo tipo di gara?". A questo riguardo è utile riportare l'opinione di un grande studioso inglese della Matematica ricreativa:
… *sebbene la conoscenza della matematica e della logica sia sovente di grande aiuto nella ricerca della soluzione, accade talvolta che una certa astuzia e un certo acume naturale siano di una grande importanza nell'affrontare questo genere di problemi.*

H. E. Dudeney (1857 - 1930)

Si forniscono allora dei riferimenti, specie nell'ambito della Matematica, su quelle conoscenze che possono essere sovente di aiuto nella ricerca della soluzione, senza illuderci però che sia una condizione sufficiente per raggiungere l'obiettivo. In generale possono essere molto utili libri su "rompicapo matematici", "matematica divertente", ecc. che riportano e commentano le soluzioni.

Un celebre esempio è:
Gardner M., Enigmi e giochi matematici, BUR Biblioteca Univ. Rizzoli , 2001

In rete si possono trovare incredibili risorse, come il libro più prestigioso del già citato Henry Ernest Dudeney:
http://www.ebook.com/eBooks/Education/Amusements_in_Mathematics/inbrowser

Dal 2002, il Politecnico di Torino, nell'ambito del progetto Polymath mantiene un sito con una rubrica mensile di problemi alcuni dei quali possono essere rappresentativi di quelli assegnati alle gare di informatica/matematica:
http://www2.polito.it/didattica/polymath/htmlS/probegio/Prob/

Sul sito delle Olimpiadi di Matematica si possono trovare esercizi e soluzioni che potrebbero comparire benissimo in una gara di informatica:

http://olimpiadi.ing.unipi.it/

Di seguito si forniscono ulteriori informazioni rispetto a ciascuna delle categorie in cui sono stati classificati i problemi assegnati nelle edizioni scolastiche delle dal 2002 al 2007.

A1 - Problemi di ottimizzazione

I problemi di questo tipo richiedono indirettamente la scoperta di procedimenti ed il loro confronto sul piano delle "prestazioni". La risoluzione del problema impone di rispettare qualche vincolo di ottimizzazione (tipicamente massimizzare o minimizzare qualcosa).
Un esempio banale è quello del noto problema Lupo-Capra-Cavolo. La domanda potrebbe essere quella di scrivere il minor numero di mosse e non di elencare le mosse. Il metodo serve appunto indirettamente ed è significativo per la scienza informatica.
La casistica è naturalmente molto vasta e tocca in generale ambiti scientifici molto impegnativi come la Ricerca Operativa e la Teoria della Complessità Computazionale. Su questi temi la letteratura anche di semplice introduzione è ampia e in continua crescita. La loro lettura sarebbe però poco utile, almeno per la preparazione alla selezione scolastica, sia perché troppo specialistici sia perché il tempo richiesto per risolvere questi problemi obbliga il risolutore ad intuire la soluzione e ad applicarla senza formalizzarla. Lo spazio delle soluzioni è infatti relativamente modesto da essere dominato " a mente" e/o drasticamente riducibile con buone intuizioni.
Più utili possono essere invece le tecniche di rappresentazione delle strategie come gli alberi di decisione o sue varianti. A titolo di esempio si può fare riferimento alla soluzione del problema delle dodici monete descritto nel libro:

Luccio F. ,"La struttura degli algoritmi", Boringhieri, 1982 (cap. 1)

A2 - Problemi di deduzione logica

Chi è chiamato a risolvere rapidamente questi problemi lo fa con tecniche logiche "informali" o con l'intuito. Molti di questi problemi si potrebbero trovare anche su comuni settimanali di enigmistica dai cui lettori non ci si aspetta di certo conoscenze di logica formale.
La conoscenza di alcuni principi generali di logica può tuttavia costituire un utile strumento per riconoscere o applicare rapidamente alcuni schemi deduttivi. Gli elementi teorici sottostanti alla risoluzione di questo tipo di problemi si trovano spesso nel primo capitolo di molti libri di testo di Matematica del biennio.
Per la soluzione di molti problemi di logica con quantificatori si possono usare strumenti di uso più elementare come i diagrammi di Venn, certamente più familiari ai ragazzi delle regole di inferenza. Gli esempi di soluzione proposti in allegato ne sono una dimostrazione.
Per qualche approfondimento ci si può riferire ad uno dei tanti libri di introduzione alla logica di cui ogni biblioteca scolastica è solitamente fornita.
Ad esempio:

Lolli, G.: Introduzione alla logica formale, Il Mulino, 1991 (primi cinque capitoli)

Anche in rete si possono reperire diversi materiali accessibili e di qualità, completi di utili esercizi, come:
Lolli, G. : http://www2.dm.unito.it/paginepersonali/lolli/appunti/Cap0.05.pdf
Palladino, D. : http://www.dif.unige.it/epi/hp/pal/dispense.htm

A3 - Problemi di calcolo

Con questa categoria di problemi si vuole indicare quelli che richiedono abilità elementari di calcolo. La soluzione è un numero che si ottiene solitamente da una espressione con le quattro operazioni, per lo più su numeri piccoli, per i quali non è richiesta la calcolatrice. A volte sono utili nozioni su proporzioni ed equazioni di primo grado, tutte riscontrabili sui libri di testo di Matematica dei primi anni.
La difficoltà maggiore è piuttosto quella della corretta interpretazione del testo del problema.

Come risorse in rete si cita ancora il progetto Polymath del politecnico di Torino:
http://www2.polito.it/didattica/polymath/htmlS/probegio/Prob/

Lo studio delle soluzioni riportate può costituire un prezioso contributo alla preparazione dei ragazzi.

A4 - Problemi di insiemistica e di calcolo combinatorio

I problemi di questo tipo hanno a che fare con l'"arte del contare", un aspetto creativo della matematica di crescente interesse. I più facili riguardano conteggi di cardinalità di insiemi o di loro partizioni. I più difficili sono spesso problemi "atipici" di calcolo combinatorio.
Lo studio dei problemi di calcolo combinatorio nella scuola è tipicamente da associarsi all'esigenza di fornire strumenti per il calcolo delle probabilità. Diversi libri di testo di matematica o di statistica/calcolo delle probabilità prevedono sezioni con nozioni di questo genere.

In rete si trovano diverse risorse, come:

- Voce della wikipedia come base per ulteriori riferimenti in rete.

http://it.wikipedia.org/wiki/Calcolo_combinatorio

- Sezione dedicata al calcolo combinatorio di un "Laboratorio virtuale di probabilità e statistica" dell'Università di Firenze

http://www.ds.unifi.it/VL/VL_IT/comb/combl.html

- Sezione di un sito (in inglese) contenente una rassegna di problemi avanzati di calcolo combinatorio

http://www.mathpages.com/home/ic

A5 - Problemi di geometria

Per i problemi di geometria sono sufficienti le nozioni di geometria piana e solida che si trovano nei libri di scuola media. Sono infatti richieste conoscenza piuttosto elementari come il calcolo della superficie o del volume di parallelepipedi. Più utile è verificare le conoscenze sui rapporti tra volumi di due solidi aventi la stessa forma (ad es. due cubi).
Spesso si tratta di ancora di applicare l'"arte del contare", come nelle domande del tipo "quanti volumi di una certa forma possono essere contenuti in un volume con un'altra forma? "
Anche qui si può fare riferimento ai già citati siti sui rompicapo matematici.

A6 - Problemi sui sistemi di numerazione

Su questo tema molti libri di testo, anche delle scuole medie, forniscono le nozioni necessarie alla risoluzione di problemi di calcolo in base diversa da 10.

Il problema non è quasi mai diretto (ad es. convertire in base 3 un dato numero); piuttosto si chiede di riconoscere quando il problema può essere risolto facendo riferimento a qualche particolare sistema di numerazione. Ad esempio, nel test: "Se 15+12 = 30 quanto fa 6+6 ?" si dovrebbe intuire che il sistema di numerazione sottointeso è quello a base 7.
Oltre che in molti libri di testo, anche in rete si trovano riferimenti ai sistemi di numerazione.
Ad esempio:
http://it.wikipedia.org/wiki/Sistema_di_numerazione

2. Informatica e Tecniche di programmazione

Le competenze di programmazione richieste per svolgere questo tipo di esercizi sono più che altro di "comprensione" del codice, specie nel corso delle ultime edizioni. Non si chiede quindi di risolvere problemi in senso tradizionale quanto piuttosto di capire cosa fa un certo frammento di codice più o meno complesso, che va dalla semplice espressione ad un ciclo o nido di cicli, dalla ricorsione semplice alla mutua ricorsione. Dalle ultime edizioni le difficoltà sono le stesse per le versioni C e Pascal perché le peculiarità di ciascun linguaggio non cambiano la sostanza dei problemi proposti.

Le quattro categorie individuate rappresentano appunto, approssimativamente, livelli diversi di difficoltà di comprensione del codice. Anche qui sono in teoria sufficienti libri di testo di matematica e/o informatica che usino il C o il Pascal come linguaggio per la codifica. L'importante è che coprano i seguenti argomenti:

- variabili, tipi ed espressioni
- campo d'azione di un identificatore
- costrutti di controllo: if, while, for, e repeat-until(Pascal) e do-while (C)
- tavole di traccia
- vettori e matrici
- traccia di funzioni ricorsive
- procedure e funzioni
- passaggio per indirizzo e per valore
- istruzioni per lettura e la scrittura dei dati

In alternativa ai libri di testo si possono utilizzare dispense e manuali di libera diffusione che si trovano facilmente e numerosi in rete (di varia qualità) , specie in lingua inglese:

Ad esempio, sul C un manuale di riferimento di comoda consultazione è:
http://www.acm.uiuc.edu/webmonkeys/book/c_guide/

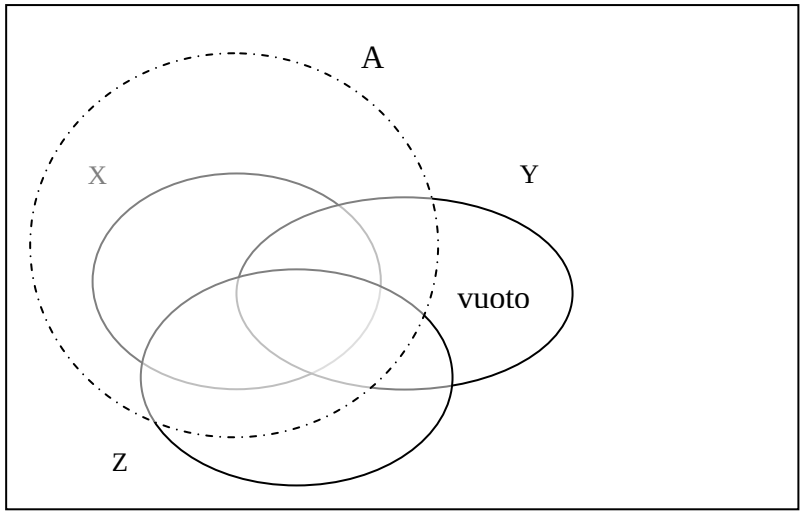
oppure, sotto l'aspetto di corso:
http://www.macs.hw.ac.uk/~rjp/Coursewww/Cwww/index.html

Un discreto tutorial sul Pascal è:
http://www.taoyue.com/tutorials/pascal/

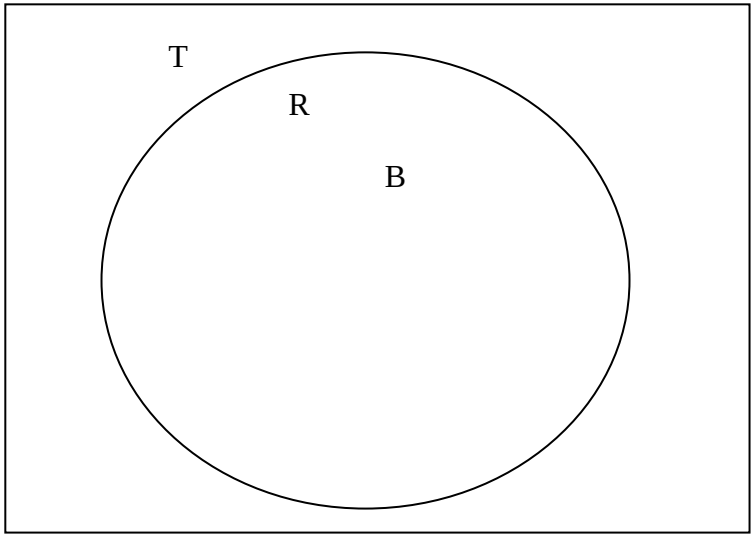
Meno facile è trovare in rete esempi di esercizi della difficoltà di quelli assegnati alle ultime edizioni scolastiche.

Allegato 3 –Problemi e relativa soluzione commentata

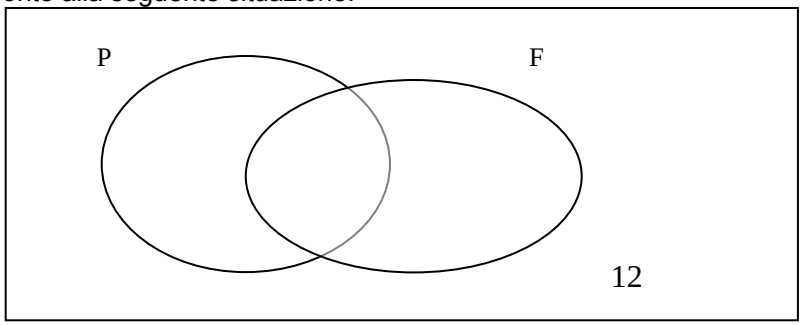
Codice ident.	Domanda	Risposta/ commento
1.2006.12.D	In un allevamento di bovini bisogna selezionare il più leggero fra 4 capi, avendo a disposizione un unico tipo di bilancia che, date due coppie di bovini, indica la coppia più leggera (si assuma che non esistano due coppie di bovini dello stesso identico peso). Nota bene: la bilancia non permette di confrontare il peso di due bovini fra loro e non fornisce il peso di una coppia di bovini. Dire quale delle seguenti affermazioni è vera: a) 2 pesate sono sempre sufficienti b) 2 pesate non sono sempre sufficienti e 3 pesate sono sempre sufficienti c) ci sono casi in cui questo tipo di bilancia non permette di trovare il bovino più leggero d) nessuna delle precedenti	**Risposta:** C Chiamiamo i quattro bovini B(1),B(2),B(3),B(4) e per ogni bovino B(i) indichiamo con P(B(i)) il suo peso. Per ipotesi del problema nessuno ha peso uguale ad un altro possiamo quindi assumere senza perdere di generalità che P(B(1)) > P(B(2)) > P(B(3)) > P(B(4)). L'allevatore dovrà confrontare i pesi delle varie coppie. Immaginiamo ora che il peso del bovino più grande si decisamente più grande rispetto agli altri, ad esempio che pesi 30Kg, mentre gli altri pesano rispettivamente 11kg,10Kg e 9Kg. Visto che P(B(1))>P(B(i))+P(B(j)) per ogni i,j con 2<= i,j<= 4 avremo che in qualsiasi pesata la qualunque sia il bovino che non mettiamo B1 avremo che il peso di quella coppia sarà maggiore di qualsiasi altra. Avremo quindi che P(B(1))+P(B(i))>P(B(j))+P(B(q)) per ogni 2<= i,j,k<=4 Non possiamo quindi in questo caso sapere qual è il bovino più leggero, ma solo sapere il bovino più pesante
2.2005.1.M (medio per ottimizzazio-ne, difficile per sistemi di numerazione)	Mario ha 83 macchinine che deve riporre in sacchetti. La suddivisione in sacchetti deve essere fatta in modo tale che quando i compagni di gioco di Mario gli chiederanno un numero qualsiasi di macchinine (compreso fra 1 e 83), Mario sarà in grado di consegnare il numero giusto di macchinine porgendo un certo numero di sacchetti senza aprirli per modificarne il contenuto. Quale è il numero minimo di sacchetti che Mario deve usare per riporre le sue macchinine? *Risposte:* *a) 7* *b) 8* *c) 21* *d) nessuna delle precedenti*	**Risposta:** A Possiamo fare riferimento alla rappresentazione binaria dei numeri da 1 a 63: con 6 sacchetti possiamo coprire qualsiasi numero da 1 a 63 (1+2+4+8+16+32). Ad esempio, nel caso in cui volessimo consegnare 10 macchinine, useremmo un sacchetto da 8 e un sacchetto da 2. Per i numeri *n* compresi fra 64 e 83 possiamo procedere così: riempiamo un ulteriore sacchetto con le restanti 20. La differenza *n*-20 sarà compresa tra 1 e 63 e la copriamo con il metodo precedentemente indicato. In questo modo potremo avere una soluzione per tutti i casi possibili.

3.2004.5.F	Durante una serata particolarmente limpida, con un bel cielo stellato, Antonio si concentra a guardare una porzione di cielo in cui sono visibili esattamente 99 stelle. Antonio si chiede quale fra queste stelle sia la più vicina. Ha a disposizione uno speciale misuratore che ha la capacità di confrontare fra loro 3 stelle ed indicarne la più vicina. Quale è il numero minimo di misurazioni che Antonio deve compiere per scoprire la stella più vicina? *Risposte:* *a) 33* *b) 49* *c) 99* *d) nessuna delle precedenti*	**Risposta:** B Numeriamo le stelle da 1 a 99 e subito cerchiamo il minimo tra le stelle 1,2,3. Prendiamo il minimo e lo confrontiamo con le stelle 4,5 e troviamo quindi il nuovo minimo. Iteriamo quindi questo procedimento finché non arriviamo alle stelle 98 e 99. All'inizio noi consideriamo le stelle 1,2,3 per cui da un totale di 99 stelle ne togliamo 3 e quindi abbiamo 96 stelle. In ogni altro passaggio consideriamo invece il minimo precedente e aggiungiamo due stelle quindi al totale togliamo due stelle. I passaggi saranno quindi 1 + (99-3)/2 = 49
4.2003.1.D	Alcune interessanti scoperte genetiche sui moscerini: - i moscerini con una mutazione del gene X sono sempre albini; - i moscerini non albini non hanno alcuna mutazione del gene Y, oppure hanno una mutazione sia del gene Y che del gene Z. Dite quale delle seguenti affermazioni e' sicuramente vera: a) se un moscerino e' albino, ha una mutazione del gene X; b) se un moscerino non ha una mutazione ne' nel gene Y ne' nel gene Z, allora e' albino; c) se un moscerino ha una mutazione nel gene Y ma non nel gene Z, allora e' albino; d) non esistono moscerini albini.	**Risposta:** C Rappresentiamo la situazione come segue (l'insieme A degli albini è tratteggiato). Dalla seconda condizione risulta vuoto l'insieme $Y - A - (Y \cap Z)$: non esistono moscerini non albini con la sola mutazione del gene Y. 

		Sottoponiamo ad esame le affermazioni: • a) è falsa, in quanto possono esistere moscerini facenti parte di A ma non di X • b) è falsa poiché possono esistere moscerini al di fuori dell'insieme $X \cup Y$ e pure al di fuori di A • c) risulta vera, in quanto gli elementi dell'insieme $Y - Z$ sono necessariamente dentro A • d) è falsa, in quanto l'esistenza di elementi in A è possibile in coerenza con tutte le condizioni.
5.2003.6.M	Nel Mercante di Venezia di Shakespeare, Porzia aveva tre scrigni, uno d'oro, uno d'argento e uno di piombo e in uno c'era il suo ritratto. Porzia voleva scegliere il suo sposo sulla base della sua intelligenza e fece incidere le seguenti iscrizioni sugli scrigni: Scrigno d'oro: il ritratto e' in questo scrigno Scrigno d'argento: il ritratto non e' in questo scrigno Scrigno di piombo: il ritratto non e' nello scrigno d'oro Porzia spiegò al suo pretendente che di queste tre affermazioni al massimo una era vera. Quale scrigno contiene il ritratto di Porzia?	**Risposta:** scrigno d'argento Conviene affrontare nell'ordine le quattro ipotesi: • Supponiamo che sia vera la frase riportata sullo scrigno d'oro: allora la scritta su quello d'argento sarebbe pure vera, il che contraddice la condizione che sia vera al massimo una affermazione. • Se fosse vera la frase sullo scrigno d'argento, allora dovrebbe essere falsa anche la frase su quello d'oro, e quindi il ritratto dovrebbe essere in quello di piombo, ma allora dovrebbe essere vera anche la frase su quest'ultimo, contraddicendo la condizione già citata. • Supponiamo sia vera la frase sullo scrigno di piombo: il ritratto non è in quello d'oro, pertanto quanto scritto su quest'ultimo risulta falso, coerentemente con le condizioni; deve essere falso anche quanto scritto sullo scrigno d'argento, e quindi il ritratto deve trovarsi in tale scrigno. • Supponiamo infine che siano false tutte e tre le affermazioni: quelle riportate sugli scrigni d'oro e di piombo sono una la negazione dell'altra, pertanto non possono essere entrambe false!
6.2007.4.F	La famiglia di Caterina consiste di Daniele, Maria, Gioia e Enzo. Essi sono la madre, il padre, il fratello e la sorella di Caterina. Trova il nome del padre, della madre, del fratello e della sorella di Caterina sapendo che 1. Daniele non ha sorelle 2. Maria non è la madre di Caterina Risposta aperta Madre …..… Padre …..… Sorella …..… Fratello …..…	**Risposta:** Madre Gioia; Padre Daniele; Sorella Maria; Fratello Enzo Applicazione del principio di non contraddizione. Visto che Daniele non ha sorelle mentre il fratello di Caterina ha ovviamente almeno una sorella, Daniele sarà di conseguenza il padre. Da cui Enzo sarà il fratello. Visto che Maria non è la madre, allora la madre sarà ovviamente Gioia , da cui Maria sarà la sorella.

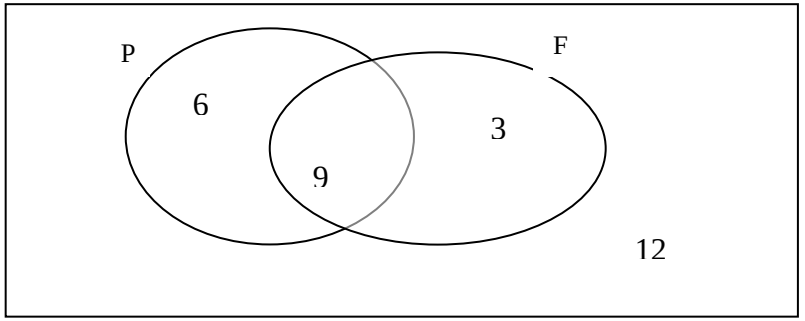
7.2004.8.M	Se tutti i belli sono ricchi e tutti i ricchi sono tristi, quale fra le seguenti frasi è corretta? *Risposte:* *a) alcuni tristi sono belli* *b) tutti i ricchi sono belli* *c) alcuni belli non sono tristi* *d) nessuna delle precedenti*	**Risposta:** A La situazione è rappresentabile con i tre insiemi B, R e T (gli insiemi di verità delle tre proposizioni) uno incluso nell'altro:  Verifichiamo le affermazioni proposte: L'affermazione a) è corretta, in quanto, se B non è vuoto, necessariamente degli elementi di T fanno anche parte di B. L'affermazione b) è scorretta perché è evidente che un elemento può appartenere ad R senza appartenere a B. L'affermazione c) è scorretta perché nel caso particolare in cui i tre insiemi B, R e T coincidono (situazione coerente con le premesse) allora tutti i belli sono necessariamente tristi. Essendo corretta a), di conseguenza è scorretta d).
8.2006.11.D	Un compito in classe inizia quando le lancette dell'orologio sono sovrapposte fra le 8 e le 9 e termina quando sono sovrapposte fra le 10 e le 11. Quanti minuti dura il compito? Risposte: a) esattamente 120 b) fra 120 e 124 c) fra 124 e 128 d) nessuna delle precedenti	**Risposta:** D Le lancette si sovrappongono ad intervalli regolari 11 volte ogni 12 ore, quindi ogni 12/11 di ora. Pertanto il compito dura 2*12/11=2,181818… ore. Si noti che 128 minuti corrispondono a 2.1333…ore.

9.2006.9.M	Il direttore di un ristorante con capienza massima 150 posti non ricorda quante erano le persone da lui servite in occasione dello scorso cenone di fine anno. Ricorda però che volendo sistemare tutte le persone servite in tavoli da 3 ne restava fuori esattamente una; inoltre, la stessa cosa succedeva sistemando tutte le persone in tavoli da 5 o tutte in tavoli da 7. Quante erano le persone servite in occasione dello scorso cenone di fine anno?	**Risposta:** 106 - Posto X il numero di invitati avremo: X ≤ 150 X= 3*K +1 X= 5* M +1 X= 7* N +1 Quindi X-1 è multiplo sia di 7 di 3 che di 5. Il più piccolo multiplo di 7, 3 e 5 è il minimo comune multiplo 3×5×7 (perché 3, 5, 7 sono numeri primi). Quindi X = 3×5×7 +1 = 106
10.2006.7.F	A Policrate che gli domandava quanti erano i suoi allievi, così rispose Pitagora: "I miei allievi possono essere suddivisi in insiemi disgiunti; in particolare - la metà coltiva la matematica - la quarta parte si dedica allo studio della natura - la settima parte ascolta con religioso silenzio le mie parole - inoltre ci sono tre allievi che non fanno nessuna delle cose precedenti" Quanti erano gli allievi di Pitagora? *Risposta aperta*	**Risposta:** 28 Gli insiemi sono disgiunti, nessuno coltiva due interessi contemporaneamente. Sia x il numero degli allievi. Allora: x/2 indica il numero di matematici x/4 indica il numero di naturalisti x/7 indica il numero di religiosi Quindi: x/2 + x/4 + x/7 +3 = x Risolvendo l'equazione si ottiene x=28
11.2007.1.F	Serena, Bruno e Roberto hanno complessivamente 100 cioccolatini. Serena e Bruno ne hanno più di 20 a testa e Roberto meno di 20. Bruno ne ha cinque in meno di Serena. Serena ne possiede una quantità pari ad un multiplo di 8. Quanti cioccolatini ha ciascuno di loro? Risposta aperta Serena ……. Roberto ……. Bruno …….	**Risposta:** Serena 48, Roberto 9, Bruno 43 Posto che Br = Cioccolatini di Bruno Ser= Cioccolatini di Serena Rob= Cioccolatini di Roberto I dati che la domanda ci fornisce sono: 100=Br+Ser+Rob Br=Ser-5 Ser= 8, 16, 24, 32, 40, 48, …., 96 (multiplo di 8) Rob<20 Dobbiamo trovare il minimo numero di cioccolatini in possesso di Serena e Bruno, per cui Roberto

		sia minore di 20 Al massimo Bruno + Serena possiedono 99 cioccolatini (Roberto ne ha 1) Al minimo Bruno e Serena ne hanno 81 (Roberto ne ha 19) Al massimo Serena ne può avere: Br+Ser<=99 (Ser-5)+Ser<=99 Ser<=52 Al minimo Serena ne può avere: Br+Ser>=81 (Ser-5)+Ser>=81 Ser>=43 L'unico numero multiplo di 8 compreso tra 43 e 52 è 48. Quindi Serena ne ha 48 Bruno ne ha 48-5=43 Roberto ne ha 100-(48+43)=9
12.2002.10.D	Bruno è una persona prudente, ha trascorso l'intero mese di settembre in montagna ma ha fatto escursioni solo 12 giorni, ogni volta che il tempo era perfetto, cioè ne' pioveva ne' era molto freddo! Al ritorno racconta che nel 50% delle giornate ha piovuto e nel 40% faceva molto freddo. Quanti giorni ha contemporaneamente fatto freddo e ha piovuto?	**Risposta:** 9 Settembre ha 30 giorni, pertanto ha piovuto 0,5*30 =15 giorni e ha fatto freddo 0,4*30 = 12 giorni. Indichiamo con P l'insieme dei giorni in cui ha piovuto e con F quello dei giorni in cui ha fatto freddo; dal testo sappiamo che in 12 giorni né ha piovuto, né ha fatto freddo. Pertanto ci troviamo di fronte alla seguente situazione:  Conveniamo di indicare con #S il numero di elementi di un insieme S.. Quindi in 18 giorni ha piovuto oppure ha fatto freddo, cioè $\#(P \cup F) = 30 - 12 = 18$. Applicando il Principio di Inclusione-Esclusione per due insiemi, si ha: $\#(P \cup F) = \#P + \#F - \#(P \cap F)$ in quanto non bisogna contare due volte gli elementi dell'intersezione. Nel nostra caso :

<table>
<tr>
<td></td>
<td></td>
<td>18 = 15 + 12 — # (P ∩ F)

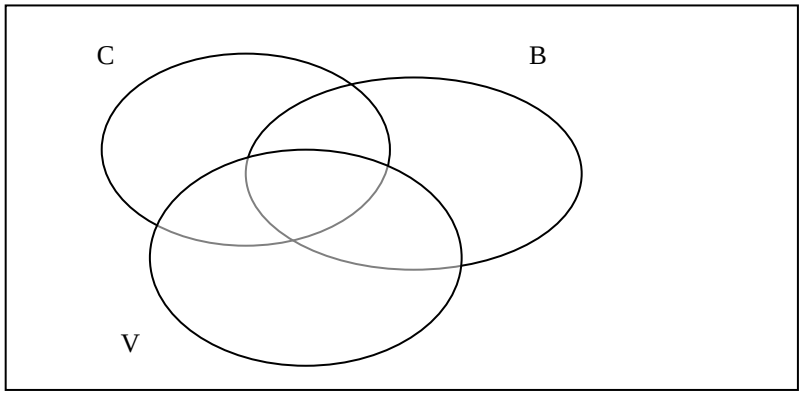
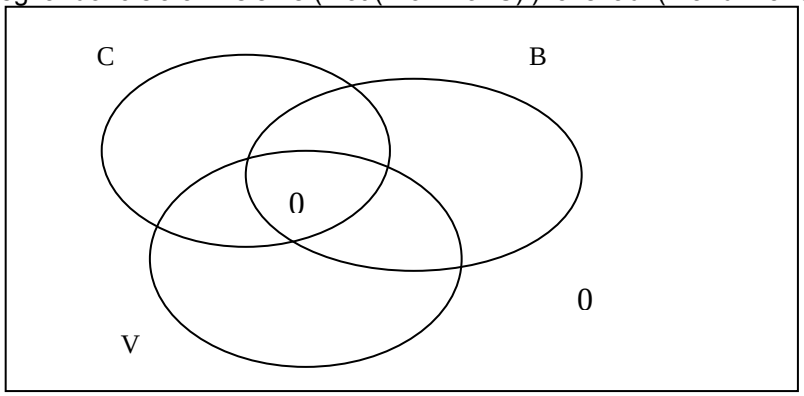
e quindi # (P ∩ F) = 9 come risulta dalla rappresentazione seguente:

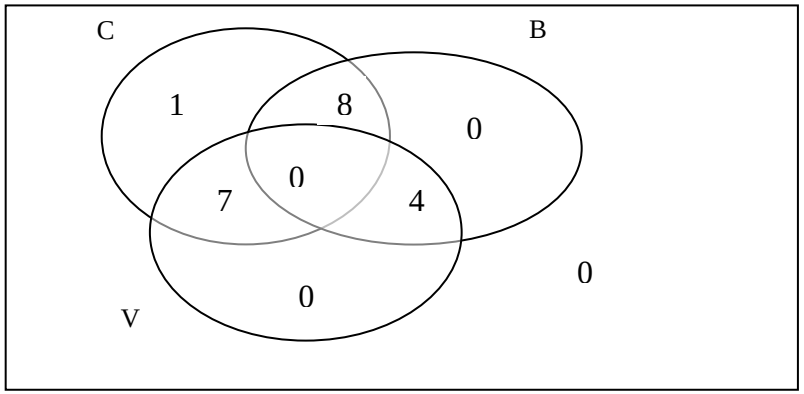
 
</td>
</tr>
<tr>
<td>13.2002.11.D</td>
<td>In quanti modi possiamo distribuire 10 caramelle a 3 bambini Aldo, Beatrice e Carla, in modo tale che ogni bambino riceva almeno due caramelle?
Nota bene: le soluzioni
Aldo = 3, Beatrice = 3, Carla=4
e
Aldo = 3,Beatrice = 4, Carla = 3
Sono due soluzioni diverse.</td>
<td>Risposta: 15
Se vogliamo che ciascun bambino riceva almeno due caramelle, troviamo, senza considerare diverse due soluzioni come quelle della nota, solo quattro possibilità:

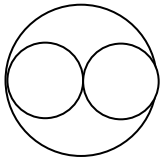
<table>
<tr><td></td><td>Aldo</td><td>Beatrice</td><td>Carla</td></tr>
<tr><td>I</td><td>2</td><td>2</td><td>6</td></tr>
<tr><td>II</td><td>2</td><td>3</td><td>5</td></tr>
<tr><td>III</td><td>2</td><td>4</td><td>4</td></tr>
<tr><td>IV</td><td>3</td><td>3</td><td>4</td></tr>
</table>

Dobbiamo però considerare, in corrispondenza di ciascuna soluzione, le possibili permutazioni dei numeri di caramelle ricevute da ciascun bambino. La situazione II presenta 3! = 6 possibilità, in quanto i te numeri 2, 3 e 5 possono essere scritti in 3! ordini diversi (come nel caso degli anagrammi di UBI) ; le altre situazioni invece permettono solamente ciascuna 3!/2! possibilità (permutazioni con ripetizione, come nel caso degli anagrammi di IBI), oppure più semplicemente possiamo dire che esistono solo tre modi per sistemare, in tre spazi, ad esempio un simbolo 2 e due simboli 4. Quindi in tutto troviamo 6+3*3 = 15 modi per distribuire le caramelle.</td>
</tr>
<tr>
<td>14.2002.9.M</td>
<td>In una classe ci sono 20 alunni, di cui 16 giocano a calcio, 12 a pallacanestro e 11 a pallavolo.
Quanti sono al minimo, coloro che praticano tutti e tre gli sport?</td>
<td>Risultato: zero

Indichiamo con C, B (Basket) e V (Volley) gli insiemi dei praticanti i tre sport.
Indichiamo inoltre, per comodità, con or, and, e not le operazioni sugli insiemi di unione,</td>
</tr>
</table>

		intersezione, e complemento rispetto all'insieme degli alunni. Si tratta di completare il grafico seguente indicando le cardinalità.  Se anche l'insieme dei praticanti il calcio includesse anche gli altri due, il numero dei non praticanti alcuno dei tre sport, da rappresentare fuori da tutti e tre gli insiemi, sarebbe al massimo 20 – 16 = 4 . La minima intersezione fra i tre insiemi si ha quando tutti praticano almeno uno sport: difatti è intuitivo che quante più persone non praticano, tanto più le altre dovranno praticare più di uno sport in modo che gli insiemi C,B e V raggiungano i numeri di elementi indicati. Proviamo assegnando 0 sia all'insieme (not (A or B or C)) che ad (A and B and C):  A questo punto, siccome Num(B) = 12 e Num(V)=11, si deduce che Num(C and B) = Num(C and V) +1. Proviamo partendo da Num(B and V) = 4, si deduce Num(C and B) = 8 e Num(C and V) = 7. A questo punto abbiamo collocato 19 elementi, e manca solo un calciatore, da collocare quindi

		nell'insieme C − (B or V). Ecco il grafico finale:  Sono rispettati tutti i vincoli, e il numero dei praticanti tutti e tre gli sport è il minimo possibile, cioè zero.
15.2006.3.F .	Quanti modelli di macchine di Formula 1 ha Mario se sono tutte Ferrari meno tre, sono tutte McLaren meno due ed ha anche una Williams? Risposta: a) 3 b) 4 c) 5 d) 6	**Risposta:** B La soluzione si ottiene risolvendo un sistema di 3 equazioni lineari in 3 incognite. Posto che F=Ferrari, M= McLaren, W = Williams e N= Totale otteniamo che: F=N-3 M=N-2 N= 1+F+M Quindi: N=1+(N-3)+(N-2) . Risolvendo si ottiene N=4
16.2004.11.D	In quante parti al massimo si può suddividere una torta con quattro tagli? *Risposte:* *a) 8* *b) 11* *c) 14* *d) nessuna delle precedenti*	**Risposta:** D Il problema chiede in sostanza di contare in quante parti può essere partizionato al massimo lo spazio "tagliandolo" con n = 4 piani. L'idea, ad ogni nuovo taglio, è di intersecare tutti i piani precedenti. Si osservi che per n = 1 la risposta è 2, per n = 2 la risposta è 4 e per n = 3 la risposta è 8. Tuttavia, per n = 4, la risposta è 15. La risposta generale a questa domanda, per n piani, è infatti data dalla formula: $C(n,0) + C(n,1) + C(n,2) + C(n,3)$ dove con $C(n,k)$ si è indicato il numero delle combinazioni di n oggetti su k posti. Nel nostro caso abbiamo quindi: 1 + 4 + 6 + 4 = 15. La risposta esatta è quindi la d ("nessuna delle precedenti"). La formula non è facilmente intuibile. Si può trovare una spiegazione, ad esempio, in : http://www.macalester.edu/~bressoud/pub/Art_of_Counting/Art_of_Counting.pdf

17.2004.10.M	Sapendo che in un cubo di due metri di lato ci possono stare 8 cubi di un metro di lato, in una sfera di due metri di raggio quante sfere di un metro di raggio ci possono stare? *Risposte:* *a) 1* *b) 4* *c) 8* *d) nessuna delle precedenti*	**Risposta:** D Osservare la figura relativa alla proiezione delle sfere sul piano. 
18.2006.6.F	Fondendo una statua di bronzo alta 50 cm e piena internamente, realizzo con il bronzo fuso ottenuto tante statuette simili (cioè con le stesse proporzioni della statua originale), anch'esse internamente piene, ma dell'altezza di 10 cm. Quante statuette riesco a realizzare? Risposte: a) 5 b) 25 c) 50 d) 125	**Risposta:** D Il volume è proporzionale al cubo della scala con cui si riducono i lati. Nel nostro caso la scala è di 1:5 10/50=1/5 Di conseguenza, il volume delle statuette piccole è 1/125 del volume iniziale. Quindi col Bronzo fuso della statua otteniamo 125 statuette.
19.2005.1.D **(identico a: 2.2005.1.M)**	Mario ha 83 macchinine che deve riporre in sacchetti. La suddivisione in sacchetti deve essere fatta in modo tale che quando i compagni di gioco di Mario gli chiederanno un numero qualsiasi di macchinine (compreso fra 1 e 83), Mario sarà in grado di consegnare il numero giusto di macchinine porgendo un certo numero di sacchetti senza aprirli per modificarne il contenuto. Quale è il numero minimo di sacchetti che Mario deve usare per riporre le sue macchinine? *Risposte:* *a) 7* *b) 8* *c) 21* *d) nessuna delle precedenti*	**Risposta:** A Possiamo fare riferimento alla rappresentazione binaria dei numeri da 1 a 63: con 6 sacchetti possiamo coprire qualsiasi numero da 1 a 63 (1+2+4+8+16+32). Ad esempio, nel caso in cui volessimo consegnare 10 macchinine, useremmo un sacchetto da 8 e un sacchetto da 2. Per i numeri *n* compresi fra 64 e 83 possiamo procedere così: riempiamo un ulteriore sacchetto con le restanti 20. La differenza *n*-20 sarà compresa tra 1 e 63 e la copriamo con il metodo precedentemente indicato. In questo modo potremo avere una soluzione per tutti i casi possibili.
20.2004.12.M	Una missione terrestre su Marte scopre una iscrizione.	**Risposta:** A

	Sapendo che dopo lunghe analisi si interpreta che l'iscrizione riporti 5+4=11, quante dita per mano aveva il marziano (si assuma che il marziano abbia avuto due mani)? *Risposte:* *a) 4* *b) 5* *c) 6* *d) nessuna delle precedenti*	Occorre riconoscere che la base del sistema di numerazione a cui si riferiscono le rappresentazioni dei numeri è 8. Si ricordi che la base di numerazione 10 è dovuto al numero delle dita della mano dell'uomo.
21.2005.12.F	Se 15+12=30 allora quanto fa 6+6? *Risposte:* *a) 15* *b) 17* *c) 19* *d) nessuna delle precedenti*	**Risposta:** A Occorre riconoscere che la base del sistema di numerazione della prima somma non è 10 ma 7. Basta quindi eseguire la seconda somma in base 7.

Soluzioni commentate ai problemi di programmazione in versione C tratti dalle gare scolastiche delle Olimpiadi Italiane di Informatica dal 2002 al 2007

22.2002.8.D	Il seguente frammento di programma e' stato scritto per effettuare la ricerca di un elemento identificato dalla variabile chiave in un array ordinato (in ordine crescente) vett con n componenti intere. Se l'elemento chiave e' presente nell'array allora il programma deve fornire in uscita il valore di un indice dell'array il cui valore e' pari al valore di chiave; se l'elemento chiave non e' presente allora il programma deve ritornare il valore –1. Il frammento di programma seguente non e' sempre corretto; tuttavia e' possibile ottenere un programma che risponde alle specifiche modificando una sola istruzione. Indicate il numero di istruzione e come deve essere scritta. (Potete, per semplicità, supporre che n sia pari a 4). ….. int ricerca(int vett[], int n, int chiave) { int inf = 0, sup = n-1, x; /* 1 */ while (inf <= sup) /* 2 */ { x = (inf + sup) /2; /* 3 */ if (chiave < vett [x]) /* 4 */ sup = x –1; /* 5 */ else if (chiave > vett[x]) /* 6 */ inf = x ; /* 7 */ else /* 8 */ return x; /* 9*/ } return –1 /* 10 */ …..	**Risposta:** riga 7 diventa inf = x+1; Il problema risulta difficile per coloro che non conoscono l'algoritmo di ricerca binaria. Occorre infatti sempre escludere l'elemento dalla bisezione successiva per non incorrere in cicli infiniti. Ad esempio, se eseguiamo il frammento: int a[] = {1}; printf("%d %", ricerca(a,1,2)); andremo incontro ad un ciclo infinto con inf = sup = 0.
23.2006.2.M	Si consideri la seguente funzione. int funzione ( ){ int contatore = 0; int sum = 0; while (contatore <= 4){ contatore = contatore + 1; sum = sum + contatore;	**Risposta:** B Il ciclo while viene eseguito 5 volte (contatore che va da 0 a 4 compreso). La variabile contatore viene incrementata all'inizio, quindi nel ciclo stesso il contatore assumerà valori da 1 a 5 compresi. Nel ciclo si calcola chiaramente la sommatoria di tali valori, che vale 15. La funzione restituisce quindi il valore 15.-

	} return sum; } Quale valore restituisce la funzione? Risposte: a) 10 b) 15 c) 16 d) Nessuna delle risposte precedenti	
24.2005.6.F	Si consideri la seguente funzione: int trova(int bersaglio, int *valori) { int contatore = 0; while(valori[contatore++] != bersaglio); return contatore-1; } Essa serve a determinare l'indice in cui si trova un certo valore (rappresentato dal parametro bersaglio) in un vettore (rappresentato dal parametro valori). La funzione, però, funziona sempre solo se vale un vincolo specifico rispetto ai dati in ingresso, quale? *Risposta aperta*	**Risposta:** Che il valore cercato esista effettivamente nel vettore di input. Si consideri che nel caso il valore non esista nel vettore, non essendoci un controllo sulla fine del vettore, avrà luogo una condizione di errore al termine della scansione del vettore stesso.
25.2006.5.D	Cosa stampa il seguente programma? #include <stdio.h> int funzione1(int arr[]) { int i = 1; while(arr[i] != -1) i = i * 2; return i; }; int funzione2(int arr[], int f, int k) { int i = 0; int m; while(i <= f) {	**Risposta:** B Il main chiama la funzione1 passando come unico parametro l'array arr. All'ingresso della funzione avremo che la variabile i ad ogni ciclo while moltiplicata per 2, assumendo quindi i valori 2, 4, 8,… Dal while si esce quando arr[i] vale -1, quindi quando i vale 4, che è il valore restituito. Quindi ora f = 4. Viene ora effettuata la chiamata: a=funzione2 (arr, f, 4). funzione2 riceve i parametri in arr, f, k, quindi avremo, al suo ingresso arr={1,2,4,8,-1,-1,-1,-1,-1,-1} i = 0 f = 4 k = 4

	m = (i + f) / 2; if(arr[m] == k) return m; if((arr[m] == -1) || (arr[m] > k)) f = m - 1; else i = m + 1; }; return -1; }; main() { int arr[10]={1,2,4,8,-1,-1,-1,-1,-1,-1}; int f=funzione1(arr); int a=funzione2(arr,f,4); int b=funzione2(arr,f,7); printf ("a=%d,b=%d\n",a,b); } Risposte: a) a=2,b=4 b) a=2,b=-1 c) a=-1,b=4 d) a=-1,b=-1	la condizione di ingresso nel while (i<=f) è vera, quindi viene calcolato m = (i+f)/2 = (0+4)/2 = 2 ora arr[m] == k è vera, quindi la funzione esce restituendo m, cioè 2. Ora nel main avremo: a = 2 f = 4 Viene effettuata la chiamata: b=funzione2 (arr, f, 7) In funzione2: i = 0 f = 4 k = 7 la condizione di ingresso nel while (i<=f) è vera, quindi viene calcolato: m = (i+f)/2 = (0+4)/2 = 2 ora (arr[m] == k) e ((arr[m] == -1) || (arr[m] > k)) sono false, e viene calcolata i = m+1 = 3 quindi ora i = 3 f = 4 k = 7 m = 2 (i<=f) è ancora vera, quindi viene calcolato: m = (i+f)/2 = (3+4)/2 = 3 (essendo m un integer) i = 3 f = 4 k = 7 m = 3 ora ((arr[m] == -1) || (arr[m] > k)) essendo vera la seconda, quindi f = m-1 = 3-1=2 i = 3 f = 2 k = 7 m = 3 (i<=f) è diventata falsa, il ciclo while non viene più eseguito e si esce dalla funzione restituendo -1. a = 2

		b = -1 sono i valori delle variabili nel main che vengono visualizzati nell'ordine.
26.2006.2.M	Si consideri la seguente funzione. int funzione ( ){ int contatore = 0; int sum = 0; while (contatore <= 4){ contatore = contatore + 1; sum = sum + contatore; } return sum; } Quale valore restituisce la funzione? Risposte: a) 10 b) 15 c) 16 d) Nessuna delle risposte precedenti	**Risposta:** B Il ciclo while viene eseguito 5 volte (contatore che va da 0 a 4 compreso). La variabile contatore viene incrementata all'inizio, quindi nel ciclo stesso il contatore assumerà valori da 1 a 5 compresi. Nel ciclo si calcola chiaramente la sommatoria di tali valori, che vale 15. La funzione restituisce quindi il valore 15..
27.2003.1bisF	Cosa stampa il seguente programma ? #include <stdio.h> int prova(int*, int); int main(void) { int a,b, c; a=11; b=0; c=prova(&a,b); printf("a = %d; b = %d; c = %d", a, b, c); return 0; } int prova(int *x, int y) { *x=y+*x; y= *x+2; return *x; } Risposte: a) a = 11; b = 0; c = 0; b) a = 13; b = 2; c = 11;	**Risposta:** D La variabile a viene passata alla funzione prova per indirizzo, la b per valore. Ci troviamo quindi la seguente situazione: prima della chiamata di prova a=11, b=0 all'ingresso nella funzione prova *x (cioè a) = 11, b = 0 dopo l'esecuzione di prova (prima del return) *x (cioè a) = 11, b = 13 all'uscita da prova (che restituisce *x cioè a) a = 11, b = 0, c = 11

	c) a = 11; b = 2; c = 11; d) a = 11; b = 0; c = 11;	
28.2007.7.D	Cosa stampa il seguente programma? #include <stdio.h> #include <stdlib.h> int mistero(int m, int n) { if (m == 0) return n; else if (n==0) return mistero(m-1, 1); else return mistero(n-1,mistero(m-1, n-1)); } int main(){ printf("%d %d %d %d\n", mistero(0,3), mistero(1,3), mistero(2,3), mistero(3,3)); return 0; } Risposte: a) 3 1 2 0 b) 3 1 1 0 c) 3 1 0 1 d) nessuna delle precedenti	**Risposta:** C La difficoltà consiste nel fatto che per valori di ingresso di m ed n diversi da 0 viene richiamata la funzione per calcolare il secondo parametro. Nella stessa chiamata viene inoltre passato come primo paramento il valore di n-1 invece di m-1- Bisogna quindi tenerne accuratamente conto nell'eseguire il tracing del programma. Vediamo, a titolo di esempio, il calcolo di mistero(1,3). Osserviamo preliminarmente che: mistero (0,x) = x (calcolo diretto, come base della ricorsione). mistero(1,3) = mistero(2,mistero(0,2)) = mistero(2,2) = mistero(1,mistero(1,1)) = mistero(0,mistero(0,0)) = mistero(0,0) = 0
29.2006.8.M	Si consideri la seguente funzione A. int B(int n); int A(int n){ if (n>1) return n*B(n+1); else return 1; } int B(int n){ if (n>1) return (n-1)*A(n-2); else return 1;	**Risposta:** B Esempio di funzioni mutuamente ricorsive. Nell'eseguire la traccia delle funzioni si tenga conto che le funzioni sono ricorsive solo per valori maggiori di 1. Quindi A(1) darà sempre 1. Durante il secondo ciclo di traccia si noterà che la chiamata della funzione A(n-2) invocata dalla funzione B è identica alla chiamata precedente come da esemplificazione che segue: A(1)=1 A(2): richiama B(3) che richiama A(1) A(3): richiama B(4) che richiama A(2) A(4): richiama B(5) che richiama A(3) A(5): richiama B(6) che richiama A(4)

	} Indicare quali sono i valori restituiti dalle invocazioni A(1), A(2), A(3), A(4), A(5). Risposte: a) 1, 4, 24, 192, 1920 b) 1, 4, 36, 576, 14400 c) 1, 4, 16, 256, 65536 d) nessuna delle precedenti	Quindi il risultato della chiamata di A da parte di B è il risultato della chiamata precedente di A. Pertanto la semplificazione della funzione è: A(n)=n*n*risultato chiamata precedente Es: A(4)=4*4*A(3) = 4*4*36 = 576
30.2007.6.F	Cosa stampa il seguente programma? #include <stdio.h> #include <stdlib.h> int calcola(int n) { if(n == 1) { return 1; } else if(n == 2) { return 3; } else if(n==3){ return n + calcola(n-1); } else { return n + calcola(n-1) + calcola(n-2); } } int main(){ printf("%d\n",calcola(6)); return 0; } Risposte: a) 41 b) 45 c) 49 d) nessuna delle precedenti	**Risposta:** D Esempio di funzione ricorsiva. Si tenga presente che per i valori da 1 a 2 il risultato è restituito direttamente: calcola (1) = 1 calcola (2) = 3 La ricorsione entra in gioco da 3 in poi: calcola(3) = 3 + calcola (2) = 3 + 3 = 6 calcola (6) = 6 + calcola (5) + calcola (4) calcola (5) = 5+ calcola 4) + calcola (3) = 5+ calcola (4) + 6 calcola (4) = 4 + calcola (3) + calcola (2) = 4+6+3 = 13 quindi : calcola (5) = 5+13+6 = 24 e infine: calcola (6) = 6+24+13 = 43

31.2005.3.D	Si consideri il seguente frammento di programma. `int B(int n);` `int A(int n){` `  int m;` `  if (n==0)` `  return 0;` `  else` `  if (n%2 == 0)` `  return n+B(n);` `  else` `  return B(n);` `}` `int B(int n){` `  int m;` `  if (n==0)` `  return 0;` `  else` `  if (n%2 == 1)` `  return n+A(n-1);` `  else` `  return A(n-1);` `}` Dire che cosa calcola la funzione A assumendo che venga invocata passando un intero positivo. Risposta aperta	**Risposta:** La somma di tutti di tutti i naturali tra 0 ed il parametro. Si tratta di un esempio di funzioni mutualmente ricorsive. In modo volutamente contorto viene calcolata la somma distinguendo tra n pari ed n dispari. Ad es. A(3) = B(3) = 3+A(2) = 3+2+B(1) = 3+2+1+A(0) = 3+2+1+0 = 6 Ricordiamo che l'operatore % corrisponde all'operatore **mod** del Pascal (resto della divisione tra interi).
32.2003.7.M	Considerate le seguenti sei funzioni, con argomento N (un intero non negativo). Ciascuna di esse calcola un qualche valore scelto tra: N, 2N , N^2, N^3, 2^N , N!, N^N. Dovete stabilire qual e' il valore calcolato da ogni funzione (potrebbero esserci doppioni: più funzioni potrebbero calcolare lo stesso valore; notate inoltre che N! = 1x2 x 3 …. x N; quindi 3! = 6 e 5!= 120;	Risposte * A: ..... 2^N * B: ..... N^2 * C: ..... 2^N * D: ..... N!

inoltre si assume che 0! = 1).

```
int A(int N) {
  if (N > 0)
    return A(N - 1) + A(N - 1);
  else
    return 1;
}


int B(int N) {
  if (N > 0)
    return B(N - 1) + 2 * N - 1;
  else
    return 0;
}


int C(int N) {
  int i, R;
  R = 1;
  for (i = 0; i < N; i++)
    R = R + R;
  return R;
}

int D(int N) {
  int i, R;
  R = 1;
  for (i = 1; i <= N; i++)
    R = R * i;
  return R;
}

int E(int N) {
  int i, R;
  R = 0;
  for (i = 0; i < N; i++)
    R = R + N;
  return R;
}
```

* E: ..... N^2

* F: ..... N!

Alcune funzioni sono ricorsive, in altre bisogna prestare attenzione agli operatori ++ per definire quando viene incrementata la variabile e calcolare quindi il suo valore esatto.

Per il resto la difficoltà sta nel riconoscere soluzioni classiche o loro piccole varianti, magari aiutandosi con piccole tracce. Ad esempio, la funzione B calcola la somma dei primi N numeri dispari e si sa che vale N^2 (ad es. 1+3+5 = 9).

	int F(int N) { if (N > 1) return N * F(N - 1); else return 1; }	
33.2004.3.F	Si considerino le due seguenti funzioni: int A(int n){ if (n > 0) return n+A(n-1); else return 0; } int B(int n){ return (n*(n+1)/2); } Quale delle seguenti affermazioni è vera? *Risposte:* *a) la funzione A calcola il fattoriale di un numero mentre la funzione B calcola la sommatoria di tutti i numeri compresi fra 1 ed n.* *b) sia la funzione A che la funzione B calcolano la sommatoria di tutti i numeri compresi fra 1 ed n, assumendo n maggiore o uguale a 1.* *c) la funzione A e la funzione B calcolano esattamente la stessa funzione.* *d) nessuna delle precedenti affermazioni è vera.*	***Risposta:*** *B* Si tratta di riconoscere semplici algoritmi e formule di base. Nel caso specifico della funzione A l'algoritmo ricorsivo che calcola la somma dei naturali da1 ad n. Nel caso della funzione B occorre riconoscere la formula di Gauss per lo stesso calcolo.

Soluzioni commentate ai problemi di programmazione in versione Pascal
tratti dalle gare scolastiche delle Olimpiadi Italiane di Informatica dal 2002 al 2007

<table>
<tr>
<td>22.2002.8.D</td>
<td>Il seguente frammento di programma e' stato scritto per effettuare la ricerca di un elemento identificato dalla variabile chiave in un array ordinato (in ordine crescente) vett con n componenti intere. Se l'elemento chiave e' presente nell'array allora il programma deve fornire in uscita il valore di un indice dell'array il cui valore e' pari al valore di chiave; se l'elemento chiave non e' presente allora il programma deve ritornare il valore –1. Il frammento di programma seguente non e' sempre corretto; tuttavia e' possibile ottenere un programma che risponde alle specifiche modificando una sola istruzione. Indicate il numero di istruzione e come deve essere scritta.

<pre>
const n=4;

type vett=array [1..n] of integer;

function posizione(v: vett; chiave: integer): integer;
var inf, sup, i: integer;
   found: boolean;
begin
      inf := 1;   (* 1 *)
      sup := n;   (* 2 *)
      found := false;   (* 3 *)
      posizione:= -1;   (* 4 *)
      while (inf<=sup) and (not found) do   (* 5 *)
            begin   (* 6 *)
                  i := (inf+sup) div 2;   (* 7 *)
                  if (chiave<v[i]) then   (* 8 *)
                        sup:=i   (* 9 *)
                  else if (chiave>v[i]) then   (* 10 *)
                        inf:=i +1   (* 11 *)
                  else   (* 12 *)
                        begin   (* 13 *)
                           found:=true;   (* 14 *)
                           posizione:=i   (* 15 *)
                        end   (* 16 *)
            end   (* 17 *)
end;   (* 18 *)
</pre>
</td>
<td>Risposta: riga 9 diventa sup := i - 1;

Il problema risulta difficile per coloro che non conoscono l'algoritmo di ricerca binaria. Occorre infatti sempre escludere l'elemento centrale (indicato, nel nostro caso, dalla variabile i) dalla bisezione successiva per non incorrere in cicli infiniti.
Questo evento su può creare, ad esempio, eseguendo il seguente corpo di programma principale:

<pre>
var a : vett;

begin
  a[1]:=10; a[2]:= 20; a[3]:=30; a[4]:=40;
  writeln(posizione(a,12));
end.
</pre>
I valori di inf e sup assunti via via all'inizio di ogni iterazione del while della funzione sono:

1 4
1 2
2 2
2 2
…

La variabile sup resta dunque bloccata sul valore 2 (come inf) impedendo così l'uscita dal while
Con la modifica proposta i valori assunti sarebbero invece:

1 4
1 1
2 1

Determinando così la falsità della condizione (inf<=sup) e la conseguente uscita dal while.</td>
</tr>
<tr>
<td>23.2006.2.M</td>
<td>Si consideri la seguente funzione.</td>
<td>Risposta: B</td>
</tr>
</table>

	function funzione ( ): integer ; var contatore: integer; sum: integer; begin contatore := 0; sum := 0; while (contatore <= 4) do begin contatore := contatore + 1; sum := sum + contatore; end; funzione := sum; end; Quale valore restituisce la funzione? Risposte: a) 10 b) 15 c) 16 d) Nessuna delle risposte precedenti	Il ciclo while viene eseguito 5 volte (contatore che va da 0 a 4 compreso). La variabile contatore viene incrementata all'inizio: quindi nel ciclo stesso il contatore assumerà i valori da 1 a 5. Nel ciclo si calcola chiaramente la sommatoria di tali valori, che vale 15. La funzione restituisce quindi il valore 15.
24.2005.6.F	Si consideri la seguente funzione: function trova(bersaglio: integer; valori: sequenza): integer; var contatore: integer; begin contatore := 1; while valori[contatore] <> bersaglio do begin contatore := contatore+1 end; trova := contatore end; Essa serve a determinare l'indice in cui si trova un certo valore (rappresentato dal parametro bersaglio) in un vettore (rappresentato dal parametro valori). La funzione, però, funziona sempre solo se vale un vincolo specifico rispetto ai dati in ingresso, quale? *Risposta aperta*	**Risposta:** Che il valore cercato esista effettivamente nel vettore di input. Si consideri che nel caso il valore non esista nel vettore, non essendoci un controllo sulla fine del vettore, avrà luogo una condizione di errore al termine della scansione del vettore stesso.

25.2006.5.D

Cosa stampa il seguente programma?

```
program Test(input,output);

  type vett=array[0..9] of integer;
  var i,f,a,b :integer;
  arr : vett;

  function funzione1(arr: vett) :integer;
  var i: integer;
  begin
    i := 1 ;
     while(arr[i] <> -1) do i:= i * 2;
     funzione1:=i;
  end;

  function funzione2(arr :vett; f: integer; k: integer) : integer;
  var i,m,temp: integer;
  begin
  i:= 0; m:= 0;
  temp:=-l;
  while(i <= f) do
      begin
        m:= (i + f) div 2;
        if(arr[m] = k) then temp:=m;
        if((arr[m] = -1) or (arr[m] > k)) then
            f:= m - 1
        else
            i := m + 1;
      end;
  funzione2:= temp;
  end;

   begin
   for i:=0 to 9 do arr[i]:= -1;
   arr[0]:=1;
   arr[1]:=2;
   arr[2]:=4;
   arr[3]:=8;
   f:=funzione1(arr);
   a:=funzione2(arr,f,4);
   b:=funzione2(arr,f,7) ;
   writeln('a=',a,',b=',b) ;
   end.
```

Risposta: B

Il programma principale chiama la funzione1 passando come unico parametro l'array arr.
All'ingresso della funzione avremo che la variabile i ad ogni ciclo while viene moltiplicata per 2, assumendo quindi i valori 2, 4, 8,… Dal while si esce quando arr[i] vale -1, quindi quando i vale 4, che è il valore restituito.
Quindi ora f = 4.
Viene ora effettuata la chiamata: a:=funzione2 (arr, f, 4).
funzione2 riceve i parametri in arr, f, k, quindi avremo, al suo ingresso
arr={1,2,4,8,-1,-1,-1,-1,-1,-1}
i = 0
f = 4
k = 4
la condizione di ingresso nel while (i<=f) è vera, quindi viene calcolato
m = (i+f)/2 = (0+4)/2 = 2
ora arr[m] = k è vera, quindi la funzione termina restituendo m, cioè 2.

Ora nel programma principale avremo:

a = 2
f = 4

Viene effettuata la chiamata: b:=funzione2 (arr, f, 7)
In funzione2:
i = 0
f = 4
k = 7

la condizione di ingresso nel while (i<=f) è vera, quindi viene calcolato:
m = (i+f)/2 = (0+4)/2 = 2
ora (arr[m] = k) e ((arr[m] = -1) or (arr[m] > k)) sono false, e viene calcolata i = m+1 = 3
quindi ora
i = 3
f = 4
k = 7
m = 2

(i<=f) è ancora vera, quindi viene calcolato:
m = (i+f)/2 = (3+4)/2 = 3 (essendo m un integer)
i = 3
f = 4

	Risposte: a) a=2,b=4 b) a=2,b=-1 c) a=-1,b=4 d) a=-1,b=-1	k = 7 m = 3 ora ((arr[m] = -1) or (arr[m] > k)) è vera, essendo vera arr[m] > k , quindi f = m-1 = 3-1=2 i = 3 f = 2 k = 7 m = 3 (i<=f) è diventata falsa, il ciclo while non viene più eseguito e si esce dalla funzione restituendo -1. a = 2 b = -1 sono i valori delle variabili del programma principale che vengono visualizzati nell'ordine.
26.2006.2.M	Si consideri la seguente funzione. function funzione ( ): integer ; var contatore: integer; sum: integer; begin contatore := 0; sum := 0; while (contatore <= 4) do begin contatore := contatore + 1; sum := sum + contatore; end; funzione := sum; end; Quale valore restituisce la funzione? Risposte: a) 10 b) 15 c) 16 d) Nessuna delle risposte precedenti	**Risposta:** B Il ciclo while viene eseguito 5 volte (contatore che va da 0 a 4 compreso). La variabile contatore viene incrementata all'inizio: quindi nel ciclo stesso il contatore assumerà i valori da 1 a 5. Nel ciclo si calcola chiaramente la sommatoria di tali valori, che vale 15. La funzione restituisce quindi il valore 15.

27.2003.1bisF

Cosa stampa il seguente programma ?

```
program Test (input, output);
var a,b,c: integer;

function  prova(x: integer;  var y:integer): integer;
begin
        x := x +y;
        y := x +2;
        prova := x
end ;

begin
        a:=11;
        b:=1;
        c:=prova(a,b);
        writeln ('a =',a, ';b =', b, ';c = ',c, '; ');
end.
```

Risposte:

a) a = 11; b = 1; c = 12;
b) a = 11; b = 1; c = 14;
c) a = 12; b = 14; c = 14;
d) a = 11; b = 14; c = 12;

Risposta: D

La variabile a viene passata alla funzione prova per valore e la variabile b per indirizzo.
Ci troviamo quindi la seguente situazione:

prima della chiamata di prova
a=11, b=1

all'ingresso nella funzione prova
x vale 11 e y (cioè b) vale 1

dopo l'esecuzione del corpo di prova
x = 12 e y (cioè b) vale = x +2 =14

all'uscita da prova (che restituisce il valore di x, cioè 12) avremo:
a = 11 (non modificato), b = 14, c = 12

28.2007.7.D

Cosa stampa il seguente programma?

```
program cosa(input,output);

function mistero(m, n : integer) : integer ;
begin
   if ( m = 0 ) then
      mistero :=  n
   else if ( n=0 ) then
      mistero := mistero( m-1, 1 )
   else
      mistero := mistero( n-1,mistero( m-1, n-1 ) )
end;

begin
  writeln(mistero(0,3),' ', mistero(1,3),' ', mistero(2,3),
  ' ', mistero(3,3) );
end.
```

Risposta: C

La difficoltà consiste nel fatto che per valori di ingresso di m ed n diversi da 0 viene richiamata la funzione per calcolare il secondo parametro.
Nella stessa chiamata viene inoltre passato come primo paramento il valore di n-1 invece di m-1.
Bisogna quindi tenerne accuratamente conto nell'eseguire lo sviluppo del calcolo:

Vediamo, a titolo di esempio, il calcolo di mistero(1,3).

Osserviamo preliminarmente che:

mistero (0,x) = x (calcolo diretto, come base della ricorsione).

e che

mistero (x,0) = mistero(x-1,1)

	Risposte: a) 3 1 2 0 b) 3 1 1 0 c) 3 1 0 1 d) nessuna delle precedenti	Calcoliamo dunque mistero(1,3): mistero(1,3) = mistero(2,mistero(0,2)) = mistero(2,2) = mistero(1,mistero(1,1)) = mistero(1,mistero(0,0)) = mistero(1,0) = mistero(0,1) = 1
29.2006.8.M	Si consideri la seguente funzione A. function B(n: integer) : integer; forward; function A(n: integer) : integer; begin if (n>1) then A := n*B(n+1) else A := 1 end; function B(n: integer) : integer; begin if (n>1) then B := (n-1)*A(n-2) else B := 1 end; Indicare quali sono i valori restituiti dalle invocazioni A(1), A(2), A(3), A(4), A(5). Risposte: a) 1, 4, 24, 192, 1920 b) 1, 4, 36, 576, 14400 c) 1, 4, 16, 256, 65536 d) nessuna delle precedenti	**Risposta:** B Esempio di funzioni mutuamente ricorsive. Nell'eseguire la traccia delle funzioni si tenga conto che le funzioni sono ricorsive solo per valori maggiori di 1. Quindi A(1) darà sempre 1. Durante il secondo ciclo di traccia si noterà che la chiamata della funzione A(n-2) invocata dalla funzione B è identica alla chiamata precedente come da esemplificazione che segue: A(1)=1 A(2): richiama B(3) che richiama A(1) A(3): richiama B(4) che richiama A(2) A(4): richiama B(5) che richiama A(3) A(5): richiama B(6) che richiama A(4) Quindi il risultato della chiamata di A da parte di B è il risultato della chiamata precedente di A. Pertanto la semplificazione della funzione è: A(n)=n*n*risultato chiamata precedente Es: A(4)=4*4*A(3) = 4*4*36 = 576
30.2007.6.F	Cosa stampa il seguente programma?	**Risposta:** D Esempio di funzione ricorsiva.

	program cosa(input,output); function calcola(n: integer) :integer ; begin if n = 1 then calcola := 1 else if n = 2 then calcola := 3 else if n = 3 then calcola := n + calcola(n-1) else calcola := n + calcola(n-1) + calcola(n-2) end; begin writeln(calcola(6)) end. Risposte: a) 41 b) 45 c) 49 d) nessuna delle precedenti	Si tenga presente che per i valori da 1 a 2 il risultato è restituito direttamente: calcola (1) = 1 calcola (2) = 3 La ricorsione entra in gioco da 3 in poi: calcola(3) = 3 + calcola (2) = 3 + 3 = 6 calcola (6) = 6 + calcola (5) + calcola (4) calcola (5) = 5+ calcola 4) + calcola (3) = 5+ calcola (4) + 6 calcola (4) = 4 + calcola (3) + calcola (2) = 4+6+3 = 13 quindi : calcola (5) = 5+13+6 = 24 e infine: calcola (6) = 6+24+13 = 43
31.2005.3.D	Si consideri il seguente frammento di programma. function B(n:integer): integer; forward; function A(n:integer): integer; begin if (n=0) then A:=0 else if (n mod 2 = 0) then A:=n+B(n) else A:=B(n) end; function B(n:integer): integer; begin if (n=0) then	**Risposta:** La somma di tutti di tutti i naturali tra 0 ed il parametro. Si tratta di un esempio di funzioni mutualmente ricorsive. In modo volutamente contorto viene calcolata la somma distinguendo tra n pari ed n dispari. Ad es. A(3) = B(3) = 3+A(2) = 3+2+B(1) = 3+2+1+A(0) = 3+2+1+0 = 6

	B:=0 else if (n mod 2 = 1) then B:=n+A(n-1) else B:=A(n-1) end; Dire che cosa calcola la funzione A assumendo che venga invocata passando un intero positivo. Risposta aperta	
32.2003.7.M	Considerate le seguenti sei funzioni, con argomento N (un intero non negativo). Ciascuna di esse calcola un qualche valore scelto tra: N, 2N , N^2, N^3, 2^N , N!, N^N. Dovete stabilire qual e' il valore calcolato da ogni funzione (potrebbero esserci doppioni: più funzioni potrebbero calcolare lo stesso valore; notate inoltre che N! = 1x2 x 3 …. x N; quindi 3! = 6 e 5!= 120; inoltre si assume che 0! = 1). FUNCTION A(N: INTEGER): INTEGER; BEGIN IF N > 0 THEN A := A(N - 1) + A(N - 1) ELSE A := 1 END; FUNCTION B(N: INTEGER): INTEGER; BEGIN IF N > 0 THEN B := B(N - 1) + 2 * N - 1 ELSE B := 0 END; FUNCTION C(N: INTEGER): INTEGER; VAR I, R : INTEGER; BEGIN R := 1;	Risposte * A: ..... 2^N * B: ..... N^2 * C: ..... 2^N * D: ..... N! * E: ..... N^2 * F: ..... N! La difficoltà sta nel riconoscere soluzioni classiche (ricorsivo o iterative) o loro piccole varianti, magari aiutandosi con piccole tracce. Ad esempio, la funzione B calcola la somma dei primi N numeri dispari e si sa che vale N^2 (ad es. 1+3+5 = 9).

	FOR I := 1 TO N DO R := R + R; C := R END; FUNCTION D(N: INTEGER): INTEGER; VAR I, R : INTEGER; BEGIN R := 1; FOR I := 1 TO N DO R := R * I; D := R END; FUNCTION E(N: INTEGER): INTEGER; VAR I, R : INTEGER; BEGIN R := 0; FOR I := 1 TO N DO R := R + N; E := R END; FUNCTION F(N: INTEGER): INTEGER; BEGIN IF N > 1 THEN F := N * F(N - 1) ELSE F := 1 END;	
33.2004.3.F	Si considerino le due seguenti funzioni: function A(n:integer):integer; begin if n>0 then A := n+A(n-1) else A := 0 end; function B(n:integer):integer; begin B := (n*(n+1) div 2) end;	**Risposta:** B Si tratta di riconoscere semplici algoritmi e formule di base; nel caso specifico della funzione A l'algoritmo ricorsivo che calcola la somma dei naturali da 1 ad n. Nel caso della funzione B occorre riconoscere la formula di Gauss per lo stesso calcolo.

	Quale delle seguenti affermazioni è vera? *Risposte:* *a) la funzione A calcola il fattoriale di un numero mentre la funzione B calcola la sommatoria di tutti i numeri compresi fra 1 ed n.* *b) sia la funzione A che la funzione B calcolano la sommatoria di tutti i numeri compresi fra 1 ed n, assumendo n maggiore o uguale a 1.* *c) la funzione A e la funzione B calcolano esattamente la stessa funzione.* *d) nessuna delle precedenti affermazioni è vera.*	