Project for the 2024/2025 autumn session

[Read very carefully "General information about the exam" as well as "Simulation of an exam project" both before starting this project and before submitting it.]

Write an ANSI C program that gets from the keyboard two non-empty strings containing only lower-case letters, recursively computes the length of the first one without using strlen – then prints to the screen the resulting number – and recursively verifies whether the first one occurs in second one and the second one occurs in the first one (not necessarily at the beginning or at the end) – then prints to the screen two suitable messages. The results have to be printed with giant characters each formed by asterisks and being 5 position high and wide.

[*Leggere molto attentamente "Informazioni generali sull'esame" e anche "Simulazione di un progetto d'esame" sia prima di iniziare questo progetto che prima di consegnarlo.*]

Scrivere un programma ANSI C che acquisisce dalla tastiera due stringhe non vuote contenenti solo lettere minuscole, calcola ricorsivamente la lunghezza della prima senza usare strlen – stampando sullo schermo il numero risultante – e verifica se la prima compare nella seconda e se la seconda compare nella prima (non necessariamente all'inizio o alla fine) – stampando sullo schermo due appositi messaggi. I risultati devono essere stampati in caratteri giganti ognuno formato da asterischi e occupante 5 posizioni sia in altezza che in larghezza.

```
*****
*   *
*****
*   *
*****

  *     ***** ***** *   * *****   *   ***** *****   ***** *   *   *****
 **     *     *   * ** ** *   *  * *  *   * *       *   **  *       *
* *     *     *   * * * * ***** ***** ***** *****     *   * * *   *****
  *     *     *   * *   * *     *   * *  *  *         *   *  **   *
*****   ***** ***** *   * *     *   * *   * *****   ***** *   *   *****

*****   ***** ***** *   * *****   *   ***** *****   ***** *   *     *
    *   *     *   * ** ** *   *  * *  *   * *       *   **  *    **
*****   *     *   * * * * ***** ***** ***** *****     *   * * *   * *
*       *     *   * *   * *     *   * *  *  *         *   *  **     *
*****   ***** ***** *   * *     *   * *   * *****   ***** *   *   *****
```

Project for the 2024/2025 summer session

[Read very carefully "General information about the exam" as well as "Simulation of an exam project" both before starting this project and before submitting it.]

Write an ANSI C program that gets from the keyboard two non-empty strings containing only lower-case letters, creates a third string given by their concatenation without using strcat – then prints to the screen the resulting string – and recursively verifies whether the first one occurs at the end of the second one and the second one occurs at the end of the first one – then prints to the screen two suitable messages with giant characters each formed by asterisks and being 5 position high and wide.

[*Leggere molto attentamente "Informazioni generali sull'esame" e anche "Simulazione di un progetto d'esame" sia prima di iniziare questo progetto che prima di consegnarlo.*]

Scrivere un programma ANSI C che acquisisce dalla tastiera due stringhe non vuote contenenti solo lettere minuscole, ne crea una terza data dalla loro concatenazione senza usare strcat – stampando sullo schermo la stringa risultante – e verifica ricorsivamente se la prima compare alla fine della seconda e se la seconda compare alla fine della prima – stampando sullo schermo due appositi messaggi in caratteri giganti ognuno formato da asterischi e occupante 5 posizioni sia in altezza che in larghezza.

```
  *     ***** ***** *   * *****   *   ***** *****   ***** ***** *   * *****   *****
 **     *     *   * ** ** *   *  * *  *   * *       *       *   **  * *           *
* *     *     *   * * * * ***** ***** ***** *****   *****   *   * * * *****   *****
  *     *     *   * *   * *     *   * *  *  *       *       *   *  ** *       *
*****   ***** ***** *   * *     *   * *   * *****   *     ***** *   * *****   *****

*****   ***** ***** *   * *****   *   ***** *****   ***** ***** *   * *****     *
    *   *     *   * ** ** *   *  * *  *   * *       *       *   **  * *         **
*****   *     *   * * * * ***** ***** ***** *****   *****   *   * * * *****   * *
*       *     *   * *   * *     *   * *  *  *       *       *   *  ** *         *
*****   ***** ***** *   * *     *   * *   * *****   *     ***** *   * *****   *****
```

Project for the 2024/2025 winter session

[Read very carefully "General information about the exam" as well as "Simulation of an exam project" both before starting this project and before submitting it.]

Write an ANSI C program that gets from the keyboard two non-empty strings containing only lower-case letters, compares them without using strcmp – then prints to the screen a value in the set $\{-1, 0, 1\}$ – and recursively verifies whether the former occurs at the beginning of the latter and the latter occurs at the beginning of the former – then prints to the screen two suitable messages. The results have to be printed with giant characters each formed by asterisks and being 5 position high and wide.

[*Leggere molto attentamente "Informazioni generali sull'esame" e anche "Simulazione di un progetto d'esame" sia prima di iniziare questo progetto che prima di consegnarlo.*]

Scrivere un programma ANSI C che acquisisce dalla tastiera due stringhe non vuote contenenti solo lettere minuscole, le confronta senza usare strcmp – stampando sullo schermo un valore nell'insieme $\{-1, 0, 1\}$ *– e verifica ricorsivamente se la prima compare all'inizio della seconda e se la seconda compare all'inizio della prima – stampando sullo schermo due appositi messaggi. I risultati devono essere stampati in caratteri giganti ognuno formato da asterischi e occupante 5 posizioni sia in altezza che in larghezza.*

```
*****
*   *
*   *
*   *
*****
```

```
  *     ***** ***** *   * *****   *   ***** *****   ***** *   * ***** ***** ***** *****   *****
 **     *     *   * ** ** *   *  * *  *   * *       *   **  *   *     *    *   *   *       *
* *     *     *   * * * * ***** ***** ***** *****   *   * * *   *     *    *   *   *   *****
  *     *     *   * *   * *     *   * *  *  *       *   *  **   *    *    *   *   *   *
*****   ***** ***** *   * *     *   * *   * *****   ***** *   * ***** ***** ***** *****   *****

*****   ***** ***** *   * *****   *   ***** *****   ***** *   * ***** ***** ***** *****     *
    *   *     *   * ** ** *   *  * *  *   * *       *   **  *   *     *    *   *   *    **
*****   *     *   * * * * ***** ***** ***** *****   *   * * *   *     *    *   *   *   * *
*       *     *   * *   * *     *   * *  *  *       *   *  **   *    *    *   *   *     *
*****   ***** ***** *   * *     *   * *   * *****   ***** *   * ***** ***** ***** *****   *****
```

Project for the 2023/2024 autumn session

[Read very carefully "General information about the exam" as well as "Simulation of an exam project" both before starting this project and before submitting it.]

Write an ANSI C program that gets from the keyboard two years between 1900 and 2099, computes the day and the month of Mardi Gras for the first year and Shrove Thursday for the second year according to the Gregorian calendar (fetching the dates from prepopulated tables is not allowed), and then for every date prints to the screen the digits of the day and the first three upper-case letters of the month (ignore all the subsequent letters) with giant characters each formed by asterisks and being 5 position high and wide.

[*Leggere molto attentamente "Informazioni generali sull'esame" e anche "Simulazione di un progetto d'esame" sia prima di iniziare questo progetto che prima di consegnarlo.*]

Scrivere un programma ANSI C che acquisisce dalla tastiera due anni compresi tra 1900 e 2099, calcola il giorno e il mese in cui cadono il Martedì Grasso per il primo anno e il Giovedì Grasso per il secondo anno in base al calendario gregoriano (non è consentito prelevare le date da tabelle precompilate) e poi per ciascuna delle due date stampa sullo schermo le cifre del giorno e le prime tre lettere maiuscole del mese (ignorare tutte le lettere successive) con caratteri giganti ognuno formato da asterischi e occupante 5 posizioni sia in altezza che in larghezza.

```
*****   *         *   *   *   *****
    *  **         ** **  * *  *   *
***** * *         * * * ***** *****
    *   *         *   * *   * *  *
***** *****       *   * *   * *   *
```

Project for the 2023/2024 summer session

[Read very carefully "General information about the exam" as well as "Simulation of an exam project" both before starting this project and before submitting it.]

Write an ANSI C program that gets from the keyboard a year between 1900 and 2099, computes the day and the month of Palm Sunday and Ash Wednesday in that year according to the Gregorian calendar (fetching the dates from prepopulated tables is not allowed), and then for every date prints to the screen the digits of the day and the first three upper-case letters of the month (ignore all the subsequent letters) with giant characters each formed by asterisks and being 5 position high and wide.

[*Leggere molto attentamente "Informazioni generali sull'esame" e anche "Simulazione di un progetto d'esame" sia prima di iniziare questo progetto che prima di consegnarlo.*]

Scrivere un programma ANSI C che acquisisce dalla tastiera un anno compreso tra 1900 e 2099, calcola il giorno e il mese in cui cadono la Domenica delle Palme e il Mercoledì delle Ceneri in quell'anno secondo il calendario gregoriano (non è consentito prelevare le date da tabelle precompilate) e poi per ciascuna delle due date stampa sullo schermo le cifre del giorno e le prime tre lettere maiuscole del mese (ignorare tutte le lettere successive) con caratteri giganti ognuno formato da asterischi e occupante 5 posizioni sia in altezza che in larghezza.

```
*****   *         *   *   *   *****
    *  **         ** **  * *  *   *
***** * *         * * * ***** *****
    *   *         *   * *   * *  *
***** *****       *   * *   * *   *
```

Project for the 2023/2024 winter session

[Read very carefully "General information about the exam" as well as "Simulation of an exam project" both before starting this project and before submitting it.]

Write an ANSI C program that gets from the keyboard a year between 1900 and 2099, computes the day and the month of Easter Monday in that year according to both the Gregorian calendar and the Julian calendar (fetching the dates from prepopulated tables is not allowed), and then for every date prints to the screen the digits of the day and the first three upper-case letters of the month (ignore all the subsequent letters) with giant characters each formed by asterisks and being 5 position high and wide.

[*Leggere molto attentamente "Informazioni generali sull'esame" e anche "Simulazione di un progetto d'esame" sia prima di iniziare questo progetto che prima di consegnarlo.*]

Scrivere un programma ANSI C che acquisisce dalla tastiera un anno compreso tra 1900 e 2099, calcola il giorno e il mese in cui cade il Lunedì dell'Angelo in quell'anno sia secondo il calendario gregoriano che secondo il calendario giuliano (non è consentito prelevare le date da tabelle precompilate) e poi per ciascuna delle due date stampa sullo schermo le cifre del giorno e le prime tre lettere maiuscole del mese (ignorare tutte le lettere successive) con caratteri giganti ognuno formato da asterischi e occupante 5 posizioni sia in altezza che in larghezza.

```
*****   *         *   *   *   *****
    *  **         ** **  * *  *   *
***** * *         * * * ***** *****
    *   *         *   * *   * *  *
***** *****       *   * *   * *   *
```

Project for the 2022/2023 autumn session

[Read very carefully "General information about the exam" as well as "Report of an exam project" both before starting the project and before submitting the project.]

Levy's conjecture claims that every odd natural number greater than 5 can be written as the sum of an odd prime number and the double of another prime number.
Marshall Hall's conjecture claims that, given two integer numbers, if the cube of the former is different from the square of the latter, then the absolute value of the difference between the cube and the square is greater than the square root of the absolute value of the former number.
Mersenne's conjecture claims that all numbers of the form $2^n - 1$ are prime when n is prime.
Write an ANSI C program that asks the user which conjecture to consider and then verifies it after getting from the keyboard the odd natural number greater than 5 in the first case (the program prints to the screen the two prime numbers), the two integer numbers in the second case (the program prints to the screen the cube of the former, the square of the latter, the absolute value of the difference between the cube and the square, the square root of the absolute value of the former number), the prime number n in the third case (the program prints to the screen the value of $2^n - 1$ and indicates whether it is prime or not).

[*Leggere molto attentamente "Informazioni generali sull'esame" come pure "Relazione di un progetto d'esame" sia prima di iniziare il progetto che prima di consegnare il progetto.*]

La congettura di Levy asserisce che ogni numero naturale dispari maggiore di 5 *può essere scritto come somma di un numero primo dispari e del doppio di un altro numero primo.*
La congettura di Marshall Hall asserisce che, dati due numeri interi, se il cubo del primo è diverso dal quadrato del secondo, allora il valore assoluto della differenza tra il cubo e il quadrato è maggiore della radice quadrata del valore assoluto del primo numero.
La congettura di Mersenne asserisce che tutti i numeri della forma $2^n - 1$ *sono primi quando n è primo.*
Scrivere un programma ANSI C che chiede all'utente quale congettura intende considerare e poi la verifica dopo aver acquisito dalla tastiera il numero naturale dispari maggiore di 5 *nel primo caso (il programma stampa sullo schermo i due numeri primi), i due numeri interi nel secondo caso (il programma stampa sullo schermo il cubo del primo numero, il quadrato del secondo numero, il valore assoluto della differenza tra il cubo e il quadrato, la radice quadrata del valore assoluto del primo numero), il numero primo n nel terzo caso (il programma stampa sullo schermo il valore di* $2^n - 1$ *e indica se è primo oppure no).*

Project for the 2022/2023 summer session

[Read very carefully "General information about the exam" as well as "Report of an exam project" both before starting the project and before submitting the project.]

Gilbreath's conjecture claims that if one examines the sequence of prime numbers, the sequence of natural numbers obtained by computing the absolute value of the difference between each pair of consecutive prime numbers in the previous sequence, the further sequence of natural numbers obtained by computing the absolute value of the difference between each pair of consecutive numbers in the previous sequence, and so on, the first number of each of the obtained sequences is always 1.
Goldbach's weak conjecture claims that all odd natural numbers greater than 5 can be written as the sum of three prime numbers.
Legendre's conjecture claims that for all $n \in \mathbb{N}$ such that $n \geq 1$ there exists a prime number between n^2 and $(n+1)^2$.
Write an ANSI C program that asks the user which conjecture to consider and then verifies it after getting from the keyboard the number of sequences to generate in the first case (for each sequence the program prints to the screen only the first 10 numbers), the odd natural number greater than 5 in the second case (the program prints to the screen the three prime numbers), $n \geq 1$ in the third case (the program prints to the screen the prime number inbetween).

[*Leggere molto attentamente "Informazioni generali sull'esame" come pure "Relazione di un progetto d'esame" sia prima di iniziare il progetto che prima di consegnare il progetto.*]

La congettura di Gilbreath asserisce che se si esaminano la sequenza dei numeri primi, la sequenza dei numeri naturali ottenuta calcolando il valore assoluto della differenza tra ciascuna coppia di numeri primi consecutivi nella precedente sequenza, l'ulteriore sequenza dei numeri naturali ottenuta calcolando il valore assoluto della differenza tra ciascuna coppia di numeri consecutivi nella precedente sequenza, e così via, il primo numero di ciascuna delle sequenze ottenute è sempre 1.
La congettura debole di Goldbach asserisce che tutti i numeri naturali dispari maggiori di 5 *possono essere scritti come somma di tre numeri primi.*
La congettura di Legendre asserisce che per ogni $n \in \mathbb{N}$ *tale che* $n \geq 1$ *esiste un numero primo compreso tra* n^2 *ed* $(n+1)^2$.
Scrivere un programma ANSI C che chiede all'utente quale congettura intende considerare e poi la verifica dopo aver acquisito dalla tastiera il numero di sequenze da generare nel primo caso (per ogni sequenza il programma stampa sullo schermo solo i primi 10 *numeri), il numero naturale dispari maggiore di* 5 *nel secondo caso (il programma stampa sullo schermo i tre numeri primi),* $n \geq 1$ *nel terzo caso (il programma stampa sullo schermo il numero primo compreso).*

Project for the 2022/2023 winter session

[Read very carefully "General information about the exam" as well as "Report of an exam project" both before starting the project and before submitting the project.]

Beal's conjecture claims that if $a^x + b^y = c^z$ where $a, b, c, x, y, z \in \mathbb{N}$ with $a, b, c \geq 1$ and $x, y, z \geq 3$, then a, b, c share a prime factor.
Collatz's conjecture claims that the function $f : \mathbb{N}_{>0} \to \mathbb{N}_{>0}$ defined by letting $f(n) = n/2$ if n is even and $f(n) = 3 \cdot n + 1$ if n is odd generates 1 after finitely many applications to the obtained numbers.
Cramer's conjecture claims that the absolute value of the difference between two consecutive prime numbers ≥ 11 is less than the square of the natural logarithm of the smallest number.
Write an ANSI C program that asks the user which conjecture to consider and then verifies it after getting from the keyboard a, b, c, x, y, z in the first case (if $a^x + b^y = c^z$ does not hold, the program prints it to the screen and then verifies anyhow whether a, b, c share a prime factor and prints its outcome to the screen), $n > 0$ in the second case (the program prints to the screen all the generated numbers), two consecutive prime numbers ≥ 11 in the third case (the program prints to the screen both the absolute value of the difference between the two numbers and the square of the natural logarithm of the smallest number).

[*Leggere molto attentamente "Informazioni generali sull'esame" come pure "Relazione di un progetto d'esame" sia prima di iniziare il progetto che prima di consegnare il progetto.*]

La congettura di Beal asserisce che se $a^x + b^y = c^z$ *dove* $a, b, c, x, y, z \in \mathbb{N}$ *con* $a, b, c \geq 1$ *e* $x, y, z \geq 3$*, allora* a, b, c *hanno un fattore primo in comune.*
La congettura di Collatz asserisce che la funzione $f : \mathbb{N}_{>0} \to \mathbb{N}_{>0}$ *definita ponendo* $f(n) = n/2$ *se* n *è pari ed* $f(n) = 3 \cdot n + 1$ *se* n *è dispari genera* 1 *dopo un numero finito di applicazioni ai numeri man mano ottenuti.*
La congettura di Cramer asserisce che il valore assoluto della differenza tra due numeri primi consecutivi ≥ 11 *è minore del quadrato del logaritmo naturale del minimo dei due.*
Scrivere un programma ANSI C che chiede all'utente quale congettura intende considerare e poi la verifica dopo aver acquisito dalla tastiera a, b, c, x, y, z *nel primo caso (se non vale* $a^x + b^y = c^z$*, il programma lo stampa sullo schermo e poi verifica comunque se* a, b, c *hanno un fattore primo in comune e ne stampa l'esito sullo schermo),* $n > 0$ *nel secondo caso (il programma stampa sullo schermo tutti i numeri generati), due numeri primi consecutivi* ≥ 11 *nel terzo caso (il programma stampa sullo schermo sia il valore assoluto della differenza tra i due numeri che il quadrato del logaritmo naturale del più piccolo dei due numeri).*

Project for the 2021/2022 autumn session

[Read very carefully `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` before starting the project as well as before submitting the project.]

An alphabet is a finite set of symbols. A finite language over an alphabet is a finite set of finite-length sequences of symbols of the alphabet. Given the alphabet $\{a, e, i, o, u\}$, write an ANSI C program that gets from the keyboard two finite languages over that alphabet and then prints to the screen the difference and the composition of the two languages. At least one of the two operations must be computed recursively.

[*Leggere molto attentamente* `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Un alfabeto è un insieme finito di simboli. Un linguaggio finito su un alfabeto è un insieme finito di sequenze di lunghezza finita di simboli dell'alfabeto. Dato l'alfabeto $\{a, e, i, o, u\}$*, scrivere un programma ANSI C che acquisisce dalla tastiera due linguaggi finiti su quell'alfabeto e poi stampa sullo schermo la differenza e la composizione dei due linguaggi. Almeno una delle due operazioni deve essere calcolata ricorsivamente.*

Project for the 2021/2022 summer session

[Read very carefully `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` before starting the project as well as before submitting the project.]

An alphabet is a finite set of symbols. A finite language over an alphabet is a finite set of finite-length sequences of symbols of the alphabet. Given the alphabet $\{a, e, i, o, u\}$, write an ANSI C program that gets from the keyboard two finite languages over that alphabet and then prints to the screen the intersection and the symmetric difference of the two languages. At least one of the two operations must be computed recursively.

[*Leggere molto attentamente* `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Un alfabeto è un insieme finito di simboli. Un linguaggio finito su un alfabeto è un insieme finito di sequenze di lunghezza finita di simboli dell'alfabeto. Dato l'alfabeto $\{a, e, i, o, u\}$*, scrivere un programma ANSI C che acquisisce dalla tastiera due linguaggi finiti su quell'alfabeto e poi stampa sullo schermo l'intersezione e la differenza simmetrica dei due linguaggi. Almeno una delle due operazioni deve essere calcolata ricorsivamente.*

Project for the 2021/2022 winter session

[Read very carefully `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` before starting the project as well as before submitting the project.]

An alphabet is a finite set of symbols. A finite language over an alphabet is a finite set of finite-length sequences of symbols of the alphabet. Given the alphabet $\{a, e, i, o, u\}$, write an ANSI C program that gets from the keyboard two finite languages over that alphabet and then prints to the screen the union of the two languages and the difference between the first language and the second one. At least one of the two operations must be computed recursively.

[*Leggere molto attentamente* `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Un alfabeto è un insieme finito di simboli. Un linguaggio finito su un alfabeto è un insieme finito di sequenze di lunghezza finita di simboli dell'alfabeto. Dato l'alfabeto $\{a, e, i, o, u\}$*, scrivere un programma ANSI C che acquisisce dalla tastiera due linguaggi finiti su quell'alfabeto e poi stampa sullo schermo l'unione dei due linguaggi e la differenza tra il primo linguaggio e il secondo. Almeno una delle due operazioni deve essere calcolata ricorsivamente.*

Project for the 2020/2021 autumn session

[Read very carefully `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` before starting the project as well as before submitting the project.]

Write an ANSI C program that gets from the keyboard a finite set of natural numbers and two binary relations over that set and then prints to the screen the difference between the first relation and the second one and the difference between the second relation and the first one, both computed recursively, followed by the reflexive closure of the first relation and the symmetric closure of the second relation.

[*Leggere molto attentamente* `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Scrivere un programma ANSI C che acquisisce dalla tastiera un insieme finito di numeri naturali e due relazioni binarie su quell'insieme e poi stampa sullo schermo la differenza tra la prima relazione e la seconda e la differenza tra la seconda relazione e la prima, entrambe calcolate ricorsivamente, seguite dalla chiusura riflessiva della prima relazione e dalla chiusura simmetrica della seconda relazione.

Project for the 2020/2021 summer session

[Read very carefully `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` before starting the project as well as before submitting the project.]

Write an ANSI C program that gets from the keyboard a finite set of natural numbers and two binary relations over that set and then prints to the screen composition and symmetric difference of the two relations. At least one of the two operations must be computed recursively.

[*Leggere molto attentamente* `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Scrivere un programma ANSI C che acquisisce dalla tastiera un insieme finito di numeri naturali e due relazioni binarie su quell'insieme e poi stampa sullo schermo composizione e differenza simmetrica delle due relazioni. Almeno una delle due operazioni deve essere calcolata ricorsivamente.

Project for the 2020/2021 winter session

[Read very carefully `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` before starting the project as well as before submitting the project.]

Write an ANSI C program that gets from the keyboard a finite set of natural numbers and two binary relations over that set and then prints to the screen union and intersection of the two relations. At least one of the two operations must be computed recursively.

[*Leggere molto attentamente* `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Scrivere un programma ANSI C che acquisisce dalla tastiera un insieme finito di numeri naturali e due relazioni binarie su quell'insieme e poi stampa sullo schermo unione e intersezione delle due relazioni. Almeno una delle due operazioni deve essere calcolata ricorsivamente.

Project for the 2019/2020 autumn session

[Read very carefully `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` before starting the project as well as before submitting the project.]

Write an ANSI C program that gets from the keyboard two square matrices of reals by allocating them dynamically and then prints to the screen determinant and inverse of the former and transpose of the latter.

[*Leggere molto attentamente* `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Scrivere un programma ANSI C che acquisisce dalla tastiera due matrici quadrate di reali allocandole dinamicamente e poi stampa sullo schermo determinante e inversa della prima e trasposta della seconda.

Project for the 2019/2020 summer session

[Read very carefully `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` before starting the project as well as before submitting the project.]

Write an ANSI C program that gets from the keyboard two matrices of real numbers by allocating them dynamically and then prints to the screen their direct sum and their direct product.

[*Leggere molto attentamente* `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Scrivere un programma ANSI C che acquisisce dalla tastiera due matrici di numeri reali allocandole dinamicamente e poi stampa sullo schermo la loro somma diretta e il loro prodotto diretto.

Project for the 2019/2020 winter session

[Read very carefully `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` before starting the project as well as before submitting the project.]

Write an ANSI C program that gets from the keyboard two matrices of real numbers by allocating them dynamically and then prints to the screen their sum and their product.

[*Leggere molto attentamente* `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Scrivere un programma ANSI C che acquisisce dalla tastiera due matrici di numeri reali allocandole dinamicamente e poi stampa sullo schermo la loro somma e il loro prodotto.

Project for the 2018/2019 autumn session

[Read very carefully `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` before starting the project as well as before submitting the project.]

Write an ANSI C library that manages sets of natural numbers by exporting the following four functions. The first function has one set as input parameter and returns the number of its elements computed recursively. The second function has one set as input parameter and returns its complement, with respect to the set of natural numbers between 0 and 100, computed recursively. The third function has two sets as input parameters and recursively establishes whether they are different. The fourth function has two sets as input parameters and returns the union of their even elements computed recursively.

[*Leggere molto attentamente* `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Scrivere una libreria ANSI C che gestisce insiemi di numeri naturali esportando le seguenti quattro funzioni. La prima funzione ha come parametro di ingresso un insieme e restituisce il numero dei suoi elementi calcolato ricorsivamente. La seconda funzione ha come parametro di ingresso un insieme e restituisce il suo complementare, rispetto all'insieme dei numeri naturali compresi tra 0 e 100, calcolato ricorsivamente. La terza funzione ha come parametri di ingresso due insiemi e stabilisce ricorsivamente se sono diversi. La quarta funzione ha come parametri di ingresso due insiemi e restituisce l'unione dei loro elementi pari calcolata ricorsivamente.

Project for the 2018/2019 summer session

[Read very carefully `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` before starting the project as well as before submitting the project.]

Write an ANSI C library that manages sets of real numbers by exporting the following four functions. The first function has two sets as input parameters and recursively establishes whether they are equal. The second function has two sets as input parameters and returns their intersection computed recursively. The third function has two sets as input parameters and returns their difference computed recursively. The fourth function has two sets as input parameters and returns their symmetric difference computed recursively.

[*Leggere molto attentamente* `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Scrivere una libreria ANSI C che gestisce insiemi di numeri reali esportando le seguenti quattro funzioni. La prima funzione ha come parametri di ingresso due insiemi e stabilisce ricorsivamente se sono uguali. La seconda funzione ha come parametri di ingresso due insiemi e restituisce la loro intersezione calcolata ricorsivamente. La terza funzione ha come parametri di ingresso due insiemi e restituisce la loro differenza calcolata ricorsivamente. La quarta funzione ha come parametri di ingresso due insiemi e restituisce la loro differenza simmetrica calcolata ricorsivamente.

Project for the 2018/2019 winter session

[Read very carefully `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` before starting the project as well as before submitting the project.]

Write an ANSI C library that manages sets of real numbers by exporting the following five functions. The first function returns a set introduced through the keyboard. The second function has a set as input parameter and prints it to the screen. The third function has a real number and a set as input parameters and recursively establishes whether that number belongs to that set. The fourth function has two sets as input parameters and recursively establishes whether the former is contained in the latter. The fifth function has two sets as input parameters and returns their union computed recursively.

[*Leggere molto attentamente* `info_esame_PP-PPL-PE_PPro-PLPr-CPro.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Scrivere una libreria ANSI C che gestisce insiemi di numeri reali esportando le seguenti cinque funzioni. La prima funzione restituisce un insieme acquisito dalla tastiera. La seconda funzione ha come parametro di ingresso un insieme e lo stampa sullo schermo. La terza funzione ha come parametri di ingresso un numero reale e un insieme e stabilisce ricorsivamente se quel numero appartiene a quell'insieme. La quarta funzione ha come parametri di ingresso due insiemi e stabilisce ricorsivamente se il primo è contenuto nel secondo. La quinta funzione ha come parametri di ingresso due insiemi e restituisce la loro unione calcolata ricorsivamente.

Project for the 2017/2018 autumn session
(PPL/PE – PLPr/CPro)

[Read very carefully `info_esame_prog_proc_logi.pdf` before starting the project as well as before submitting the project.]

Consider a game in which there are five I's on the first row, four I's on the second row, three I's on the third row, two I's on the fourth row, and one I on the fifth row. Two players alternate to delete I's, where each deleted I becomes −. In each turn, the respective player can delete arbitrarily many I's, provided that they are on the same row and consecutive (i.e., without − in between); the losing player is the one who deletes the last I. Write an ANSI C program that gets from the keyboard a move at a time by showing on the screen the updated situation (in terms of I's and − on the various rows) and at the end communicates the win of one of the two players by printing to the screen a suitable message.

[*Leggere molto attentamente* `info_esame_prog_proc_logi.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Considerare un gioco in cui ci sono cinque I sulla prima fila, quattro I sulla seconda fila, tre I sulla terza fila, due I sulla quarta fila e una I sulla quinta fila. Due giocatori si alternano a cancellare le I, dove ogni I cancellata diventa −. In ogni turno il rispettivo giocatore può cancellare un numero arbitrario di I, purché queste siano sulla stessa fila e consecutive (cioè senza − in mezzo); perde chi cancella l'ultima I. Scrivere un programma ANSI C che acquisisce dalla tastiera una mossa alla volta mostrando sullo schermo la situazione aggiornata (in termini di I e − sulle varie file) e alla fine comunica la vittoria di uno dei due giocatori stampando sullo schermo un apposito messaggio.

Project for the 2017/2018 summer session
(PPL/PE – PLPr/CPro)

[Read very carefully `info_esame_prog_proc_logi.pdf` before starting the project as well as before submitting the project.]

The tic-tac-toe game takes place on a 3×3 table on which the moves of two players respectively denoted by X and O alternate. The game ends with the victory of X (resp. O) if three X (resp. O) symbols have been placed on the same row, column, or diagonal. Write an ANSI C program that gets from the keyboard a move at a time by showing on the screen the updated table contents and determines the win of one of the two players or draw by printing to the screen a suitable message.

[*Leggere molto attentamente* `info_esame_prog_proc_logi.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Il gioco del tris si svolge su una tabella 3×3 *in cui si alternano le mosse di due giocatori indicati rispettivamente con* X *e* O*. Il gioco si conclude con la vittoria di* X *(risp.* O*) se sono stati messi tre simboli* X *(risp.* O*) sulla stessa riga, colonna o diagonale. Scrivere un programma ANSI C che acquisisce dalla tastiera una mossa alla volta mostrando sullo schermo il contenuto aggiornato della tabella e determina la vittoria di uno dei due giocatori o il pareggio stampando sullo schermo un apposito messaggio.*

Project for the 2017/2018 winter session
(PPL/PE – PLPr/CPro)

[Read very carefully `info_esame_prog_proc_logi.pdf` before starting the project as well as before submitting the project.]

Numeric addresses for devices on the Internet are of the form $m.n.p.q$ where m, n, p, q are positive integers. Two devices are on the same local network when they have matching numbers in the first two components of their addresses. Write an ANSI C program that gets from the keyboard a list of numeric addresses, each equipped with the symbolic name of the corresponding device (maximum length 10 characters), and prints to the screen all sets of names of devices that are on the same local network.

[This project can be submitted also by first-year PLPr students, although they will not be allowed to take the exam in the winter session.]

[*Leggere molto attentamente* `info_esame_prog_proc_logi.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Gli indirizzi numerici per i dispositivi collegati a Internet sono della forma m.n.p.q dove m, n, p, q sono interi positivi. Due dispositivi sono nella stessa rete locale quando hanno numeri coincidenti nelle prime due componenti dei loro indirizzi. Scrivere un programma ANSI C che acquisisce dalla tastiera un elenco di indirizzi numerici, ciascuno corredato dal nome simbolico del relativo dispositivo (lunghezza massima 10 *caratteri), e stampa sullo schermo tutti gli insiemi di nomi di dispositivi che sono nella stessa rete locale.*

[*Questo progetto può essere consegnato anche da studenti di PPL del primo anno, i quali però non potranno sostenere l'esame nella sessione invernale.*]

Project for the 2016/2017 autumn session
(PPL/PE – PLPr/CPro)

[Read very carefully `info_esame_prog_proc_logi.pdf` before starting the project as well as before submitting the project.]

A circular list is a linear dynamic data structure in which every element, including the last one, has a pointer to the next one. Assuming that the information content of each element is an integer number, write an ANSI C library for circular lists that exports functions for calculating the length of a list, traversing a list starting from a certain element specified through its position in the list where the position has to be considered modulo the length of the list, inserting a given value at the beginning or at the end of a list or just before or just after an element of the list that contains a certain value, removing the element at the beginning or at the end of a list or that contains a certain value.

[*Leggere molto attentamente* `info_esame_prog_proc_logi.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Una lista circolare è una struttura dati dinamica lineare in cui ogni elemento, incluso l'ultimo, ha un puntatore a quello successivo. Assumendo che il contenuto informativo di ciascun elemento sia un numero intero, scrivere una libreria ANSI C per le liste circolari che esporta le funzioni per calcolare la lunghezza di una lista, visitare una lista a partire da un certo elemento specificato attraverso la sua posizione nella lista dove la posizione va considerata modulo la lunghezza della lista, inserire un dato valore all'inizio o alla fine di una lista oppure subito prima o subito dopo un elemento della lista che contiene un certo valore, rimuovere l'elemento che si trova all'inizio o alla fine di una lista oppure che contiene un certo valore.

Project for the 2016/2017 summer session
(PPL/PE – PLPr/CPro)

[Read very carefully `info_esame_prog_proc_logi.pdf` before starting the project as well as before submitting the project.]

A doubly linked list is a linear dynamic data structure in which every element has a pointer to the next one and a pointer to the previous one. Assuming that the information content of each element is an integer number, write an ANSI C library that, for a doubly linked list, exports functions for inserting a given value at the beginning or at the end of the list or just before or just after an element of the list that contains a certain value, removing the element at the beginning or at the end of the list or that contains a certain value, traversing the list forward or backward starting from a certain element specified through its position in the list.

[*Leggere molto attentamente* `info_esame_prog_proc_logi.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Una lista con doppio collegamento è una struttura dati dinamica lineare in cui ogni elemento ha un puntatore a quello successivo e un puntatore a quello precedente. Assumendo che il contenuto informativo di ciascun elemento sia un numero intero, scrivere una libreria ANSI C che, per una lista con doppio collegamento, esporta le funzioni per inserire un dato valore all'inizio o alla fine della lista oppure subito prima o subito dopo un elemento della lista che contiene un certo valore, rimuovere l'elemento che si trova all'inizio o alla fine della lista oppure che contiene un certo valore, visitare la lista all'avanti o all'indietro a partire da un certo elemento specificato attraverso la sua posizione nella lista.

Project for the 2016/2017 winter session
(PPL/PE – PLPr/CPro)

[Read very carefully `info_esame_prog_proc_logi.pdf` before starting the project as well as before submitting the project.]

Write an ANSI C program that gets from the keyboard a sequence of parentheses, square brackets, and curly braces and prints to the screen the longest prefix of the sequence in which they correctly match. For instance, in ([{()}])] the considered prefix is ([{()} because the second right curly brace does not match any left curly brace, while in ([{()}] the considered prefix is ([{(because between the two curly braces there is no right parenthesis.

[This project can be submitted also by first-year PLPr students, although they will not be allowed to take the exam in the winter session.]

[*Leggere molto attentamente* `info_esame_prog_proc_logi.pdf` *sia prima di iniziare il progetto che prima di consegnare il progetto.*]

Scrivere un programma ANSI C che acquisisce dalla tastiera una sequenza di parentesi tonde, quadre e graffe e stampa sullo schermo il prefisso più lungo della sequenza in cui esse sono chiuse correttamente. Ad esempio, in ([{()}])] *il prefisso in questione è* ([{()} *perché la seconda graffa chiusa non corrisponde a nessuna graffa aperta, mentre in* ([{()}] *il prefisso in questione è* ([{(*perché tra le due graffe manca la tonda chiusa.*

[*Questo progetto può essere consegnato anche da studenti di PPL del primo anno, i quali però non potranno sostenere l'esame nella sessione invernale.*]

Project for the 2015/2016 autumn session
(PPL/PE – PLPr/CPro)

Write an ANSI C program that gets from the keyboard an arithmetical expression in infix notation and prints to the screen the corresponding arithmetical expressions in prefix and postfix notation, respectively. For the sake of simplicity, assume that the original expression contains only numbers (no variables) and occurrences of the four binary arithmetical operators (no unary operators), together with possible parentheses.

Scrivere un programma ANSI C che acquisisce dalla tastiera un'espressione aritmetica in notazione infissa e stampa sullo schermo le corrispondenti espressioni aritmetiche in notazione prefissa e postfissa, rispettivamente. Per semplicità, assumere che l'espressione di partenza contenga soltanto numeri (no variabili) e occorrenze dei quattro operatori aritmetici binari (no operatori unari), più eventuali parentesi tonde.

Project for the 2015/2016 summer session
(PPL/PE – PLPr/CPro)

Write an ANSI C program that reads from a file an alphabet together with the description of an automata based on that alphabet and then verifies whether, for each sequence of symbols of the alphabet gotten from the keyboard, the automata accepts that sequence or not. The automata is a directed graph in which every edge is labeled with a symbol of the alphabet, exactly one state is designated as initial, and several states can be designated as accepting. A sequence of symbols of the alphabet is accepted by the automata if there exists in the graph a path from the initial state to one of the accepting states, such that the sequence of symbols labeling the edges of the path coincides with the given sequence.

Scrivere un programma ANSI C che legge da file un alfabeto assieme alla descrizione di un automa basato su quell'alfabeto e poi verifica se, per ogni sequenza di simboli dell'alfabeto acquisita dalla tastiera, l'automa accetta o meno quella sequenza. L'automa è un grafo diretto in cui ogni arco è etichettato con un simbolo dell'alfabeto, esattamente uno stato è designato come iniziale e più stati possono essere designati come riconoscitori. Una sequenza di simboli dell'alfabeto è accettata dall'automa se esiste nel grafo un percorso dallo stato iniziale a uno degli stati riconoscitori, tale che la sequenza di simboli che etichettano gli archi del percorso coincide con la sequenza data.

Project for the 2015/2016 winter session
(PPL/PE – PLPr/CPro)

Write an ANSI C library that handles complex numbers by exporting the following functions. The first C function gets from the keyboard a complex number in algebraic (respectively, polar) form and prints to the screen the equivalent complex number in polar (respectively, algebraic) form. The second C function gets from the keyboard two complex numbers in algebraic (respectively, polar) form and prints to the screen their sum in algebraic (respectively, polar) form. The third C function gets from the keyboard two complex numbers in algebraic (respectively, polar) form and prints to the screen their difference in algebraic (respectively, polar) form. The fourth C function gets from the keyboard two complex numbers in algebraic (respectively, polar) form and prints to the screen their product in algebraic (respectively, polar) form. The fifth C function gets from the keyboard two complex numbers in algebraic (respectively, polar) form and prints to the screen their quotient in algebraic (respectively, polar) form.

[This project can be submitted also by first-year PLPr students, although they will not be allowed to take the exam in the winter session.]

Scrivere una libreria ANSI C che gestisce i numeri complessi esportando le seguenti funzioni. La prima funzione C acquisisce dalla tastiera un numero complesso in forma algebrica (rispettivamente, trigonometrica) e stampa sullo schermo il numero complesso equivalente in forma trigonometrica (rispettivamente, algebrica). La seconda funzione C acquisisce dalla tastiera due numeri complessi in forma algebrica (rispettivamente, trigonometrica) e stampa sullo schermo la loro somma in forma algebrica (rispettivamente, trigonometrica). La terza funzione C acquisisce dalla tastiera due numeri complessi in forma algebrica (rispettivamente, trigonometrica) e stampa sullo schermo la loro differenza in forma algebrica (rispettivamente, trigonometrica). La quarta funzione C acquisisce dalla tastiera due numeri complessi in forma algebrica (rispettivamente, trigonometrica) e stampa sullo schermo il loro prodotto in forma algebrica (rispettivamente, trigonometrica). La quinta funzione C acquisisce dalla tastiera due numeri complessi in forma algebrica (rispettivamente, trigonometrica) e stampa sullo schermo il loro quoziente in forma algebrica (rispettivamente, trigonometrica).

[*Questo progetto può essere consegnato anche da studenti di PPL del primo anno, i quali però non potranno sostenere l'esame nella sessione invernale.*]

Project for the 2014/2015 autumn session
(PPL/PE – PLPr/CPro)

Write an ANSI C program that gets from the keyboard a set and a binary operation with respect to which that set is closed and then verifies whether the operation satisfies the following properties: commutativity, associativity, idempotence, existence of neutral element, existence of absorbing element. For each property that is not satisfied, the program must show a counterexample.

Scrivere un programma ANSI C che acquisisce dalla tastiera un insieme e un'operazione binaria rispetto alla quale quell'insieme è chiuso e poi verifica se l'operazione soddisfa le seguenti proprietà: commutatività, associatività, idempotenza, esistenza elemento neutro, esistenza elemento assorbente. Per ciascuna proprietà non soddisfatta, il programma deve riportare un controesempio.

Project for the 2014/2015 summer session
(PPL/PE – PLPr/CPro)

Write an ANSI C program that gets from the keyboard a set, a binary relation over that set, and a binary operation over that set and then verifies whether the set is closed with respect to the operation and whether the relation is a congruence with respect to the operation.

Scrivere un programma ANSI C che acquisisce dalla tastiera un insieme, una relazione binaria su quell'insieme e un'operazione binaria su quell'insieme e poi verifica se l'insieme è chiuso rispetto all'operazione e se la relazione è una congruenza rispetto all'operazione.

Project for the 2014/2015 winter session
(PPL/PE – PLPr/CPro)

Write an ANSI C library that manages binary relations by exporting the following functions. The first C function returns a binary relation introduced through the keyboard. The second C function has a binary relation as input parameter and prints it to the screen. The third C function has a binary relation as input parameter and establishes whether it is a partial order relation, printing to the screen which property does not hold in the case that the relation is not such. The fourth C function has a binary relation as input parameter and establishes whether it is a total order relation, printing to the screen which property does not hold in the case that the relation is not such. The fifth C function has a binary relation as input parameter and establishes whether it is an equivalence relation, printing to the screen which property does not hold in the case that the relation is not such. The sixth C function has a binary relation as input parameter and establishes whether it is a mathematical function; if it is not, then the element violating the property will be printed to the screen, otherwise a message will be printed to the screen indicating whether the function is injective, surjective, or bijective.

[This project can be submitted also by first-year PLPr students, although they will not be allowed to take the exam in the winter session.]

Scrivere una libreria ANSI C che gestisce le relazioni binarie esportando le seguenti funzioni. La prima funzione C restituisce una relazione binaria acquisita dalla tastiera. La seconda funzione C ha come parametro di ingresso una relazione binaria e la stampa sullo schermo. La terza funzione C ha come parametro di ingresso una relazione binaria e stabilisce se essa è una relazione d'ordine parziale, stampando sullo schermo quale proprietà non vale nel caso la relazione non sia tale. La quarta funzione C ha come parametro di ingresso una relazione binaria e stabilisce se essa è una relazione d'ordine totale, stampando sullo schermo quale proprietà non vale nel caso la relazione non sia tale. La quinta funzione C ha come parametro di ingresso una relazione binaria e stabilisce se essa è una relazione d'equivalenza, stampando sullo schermo quale proprietà non vale nel caso la relazione non sia tale. La sesta funzione C ha come parametro di ingresso una relazione binaria e stabilisce se essa è una funzione matematica; se non lo è, allora si stamperà sullo schermo quale elemento violi la proprietà, altrimenti si stamperà sullo schermo un messaggio che indica se la funzione è iniettiva, suriettiva o biiettiva.

[*Questo progetto può essere consegnato anche da studenti di PPL del primo anno, i quali però non potranno sostenere l'esame nella sessione invernale.*]

Project for the 2013/2014 autumn session
(PPL/PE – PLPr/CPro)

Write an ANSI C program that gets a Prolog program stored in a file and a goal from the keyboard and prints to the screen a path of the resulting and-or tree at a time. After generating each path, the user should be offered the choice between generating a further path or terminating. For the sake of simplicity, every atomic formula within the Horn clauses of the Prolog program and the goal will only consist of a predicate symbol, which thus will not be followed by a sequence of terms enclosed in parentheses.

Scrivere un programma ANSI C che acquisisce un programma Prolog memorizzato su file e un obiettivo dalla tastiera e stampa sullo schermo un cammino del risultante albero e-o alla volta. Dopo la generazione di ogni cammino, deve essere data all'utente la possibilità di generare un ulteriore cammino oppure di terminare. Per semplicità, ogni formula atomica all'interno delle clausole Horn del programma Prolog e dell'obiettivo consisterà unicamente in un simbolo di predicato, che quindi non sarà seguito da una sequenza di termini racchiusi tra parentesi tonde.

Project for the 2013/2014 summer session
(PPL/PE – PLPr/CPro)

Write an ANSI C program that gets from the keyboard a predicate logic term (not a formula) together with two substitutions and prints to the screen the term resulting from the application of the composition of the two substitutions.

Scrivere un programma ANSI C che acquisisce dalla tastiera un termine (non una formula) di logica dei predicati assieme a due sostituzioni e stampa sullo schermo il termine risultante dall'applicazione della composizione delle due sostituzioni.

Project for the 2013/2014 winter session
(PPL/PE – PLPr/CPro)

Write an ANSI C program that gets from the keyboard two propositional logic formulas and establishes whether they are equivalent. For the sake of simplicity, each formula can contain at most three occurrences of logical connectives (not, or, and, if-then, iff), which can be represented as single decimal digits. As a consequence, the formula can contain at most four occurrences of propositions, which can be represented as single letters. Each formula must not contain parentheses, thus it is necessary to apply precedence and associativity rules.

[This project can be submitted also by first-year PLPr students, although they will not be allowed to take the exam in the winter session.]

Scrivere un programma ANSI C che acquisisce dalla tastiera due formule di logica proposizionale e stabilisce se esse sono equivalenti. Per semplicità, ciascuna formula può contenere al più tre occorrenze di connettivi logici (negazione, disgiunzione, congiunzione, implicazione, doppia implicazione), i quali possono essere rappresentati come singole cifre decimali. Di conseguenza, la formula può contenere al più quattro occorrenze di proposizioni, le quali possono essere rappresentate come singole lettere. Ciascuna formula non deve contenere parentesi, quindi bisogna applicare le regole di precedenza e associatività.

[*Questo progetto può essere consegnato anche da studenti di PPL del primo anno, i quali però non potranno sostenere l'esame nella sessione invernale.*]

Project for the 2012/2013 autumn session
(PPL/PE – PLPr/CPro)

Write an ANSI C program that gets from the keyboard a propositional logic formula and a truth assignment and establishes whether the assignment satisfies the formula by applying the recursive definition of relation $\models$.

Scrivere un programma ANSI C che acquisisce dalla tastiera una formula di logica proposizionale e un assegnamento di verità e stabilisce se l'assegnamento soddisfa la formula applicando la definizione ricorsiva della relazione $\models$.

Project for the 2012/2013 summer session
(PPL/PE – PLPr/CPro)

Write an ANSI C program that gets from the keyboard two predicate logic terms and establishes whether they are unifiable, in which case it prints to the screen a unifier for them.

Scrivere un programma ANSI C che acquisisce dalla tastiera due termini di logica dei predicati e stabilisce se essi sono unificabili, nel qual caso stampa sullo schermo un unificatore per essi.

Project for the 2012/2013 winter session

(PPL/PE – PLPr/CPro)

Write an ANSI C program that gets from the keyboard a predicate logic formula and establishes whether it is closed or open after printing to the screen, for each variable occurrence, whether that occurrence is bound or free. For the sake of simplicity, assume that all predicates are unary and that their only argument is always a variable.

[This project can be submitted also by first-year PLPr students, although they will not be allowed to take the exam in the winter session.]

Scrivere un programma ANSI C che acquisisce dalla tastiera una formula di logica dei predicati e stabilisce se essa è chiusa o aperta dopo aver stampato sullo schermo, per ogni occorrenza di variabile, se quella occorrenza è legata o libera. Per semplicità, assumere che tutti i predicati siano unari e che il loro unico argomento sia sempre una variabile.

[*Questo progetto può essere consegnato anche da studenti di PPL del primo anno, i quali però non potranno sostenere l'esame nella sessione invernale.*]

Project for the 2011/2012 autumn session
(PPL/PE – PLPr/CPro)

Write an ANSI C program that gets from the keyboard a predicate logic formula in prenex normal form, transforms it into Skolem normal form, and prints to the screen the latter normal form. For the sake of simplicity, assume that in the matrix of the formula there is a single occurrence of predicate symbol and that it is applied only to constants and variables.

Scrivere un programma ANSI C che acquisisce dalla tastiera una formula di logica dei predicati in forma normale prenessa, la trasforma in forma normale di Skolem e stampa sullo schermo quest'ultima forma normale. Per semplicità, assumere che nella matrice della formula compaia una sola occorrenza di simbolo di predicato e che essa sia applicata solo a costanti e variabili.

Project for the 2011/2012 summer session
(PPL/PE – PLPr/CPro)

Write an ANSI C program that gets from the keyboard a set of propositional logic clauses and establishes whether it is unsatisfiable by using the Robinson resolution method. At each resolution step, the program must print to the screen the two considered clauses and their resolvent clause.

Scrivere un programma ANSI C che acquisisce dalla tastiera un insieme di clausole di logica proposizionale e stabilisce se esso è insoddisfacibile utilizzando il metodo di risoluzione di Robinson. A ogni passo di risoluzione, il programma deve stampare sullo schermo le due clausole considerate e la loro clausola risolvente.

Project for the 2011/2012 winter session
(PPL/PE – PLPr/CPro)

Write an ANSI C program that gets from the keyboard a propositional logic formula and, if any, prints to the screen a truth assignment that satisfies it. The formula must contain at most three occurrences of logical connectives (not, or, and, if-then, iff, xor, nor, nand), which can be represented for simplicity as decimal digits. As a consequence, the formula must contain at most four occurrences of propositions, which can be represented for simplicity as single letters. The formula must not contain parentheses, thus in order to establish its truth value it is necessary to apply precedence and associativity rules.

[This project can be submitted also by first-year PLPr students, although they will not be allowed to take the exam in the winter session.]

Scrivere un programma ANSI C che acquisisce dalla tastiera una formula di logica proposizionale e, se esiste, stampa sullo schermo un assegnamento di verità che la soddisfa. La formula deve contenere al più tre occorrenze di connettivi logici (negazione, disgiunzione, congiunzione, implicazione, doppia implicazione, disgiunzione esclusiva, disgiunzione negata, congiunzione negata), i quali possono essere rappresentati per semplicità come cifre decimali. Di conseguenza, la formula deve contenere al più quattro occorrenze di proposizioni, le quali possono essere rappresentate per semplicità come singole lettere. La formula non deve contenere parentesi, quindi per stabilirne il valore di verità bisogna applicare le regole di precedenza e associatività.

[*Questo progetto può essere consegnato anche da studenti di PPL del primo anno, i quali però non potranno sostenere l'esame nella sessione invernale.*]

Project for the 2010/2011 autumn session
(PPL/PE – PLPr/CPro)

Write an ANSI C program that gets from the keyboard a propositional logic formula in conjunctive normal form, establishes whether it is a Horn formula and, if this is the case, establishes whether it is satisfiable by printing to the screen the minimum truth assignment that satisfies it.

Scrivere un programma ANSI C che acquisisce dalla tastiera una formula di logica proposizionale in forma normale congiuntiva, stabilisce se essa è una formula di Horn e, in caso affermativo, stabilisce se essa è soddisfacibile stampando sullo schermo il minimo assegnamento di verità che la soddisfa.

Project for the 2010/2011 summer session
(PPL/PE – PLPr/CPro)

Write an ANSI C program that gets from the keyboard a propositional logic formula in conjunctive normal form, transforms it into an equivalent propositional logic formula in disjunctive normal form, prints to the screen the latter formula, and establishes whether it is satisfiable.

Scrivere un programma ANSI C che acquisisce dalla tastiera una formula di logica proposizionale in forma normale congiuntiva, la trasforma nell'equivalente formula di logica proposizionale in forma normale disgiuntiva, stampa sullo schermo quest'ultima formula e stabilisce se essa è soddisfacibile.

Project for the 2010/2011 winter session
(PPL/PE – PLPr/CPro)

Write an ANSI C program that gets from the keyboard a propositional logic formula in disjunctive normal form, transforms it into an equivalent propositional logic formula in conjunctive normal form, prints to the screen the latter formula, and establishes whether it is a tautology.

[This project can be submitted also by first-year PLPr students, although they will not be allowed to take the exam in the winter session.]

Scrivere un programma ANSI C che acquisisce dalla tastiera una formula di logica proposizionale in forma normale disgiuntiva, la trasforma nell'equivalente formula di logica proposizionale in forma normale congiuntiva, stampa sullo schermo quest'ultima formula e stabilisce se essa è una tautologia.

[*Questo progetto può essere consegnato anche da studenti di PPL del primo anno, i quali però non potranno sostenere l'esame nella sessione invernale.*]

Project for the 2009/2010 autumn session

(PE – CPro)

Write an ANSI C library that manages sets of natural numbers by exporting the following five functions. The first function returns a set introduced through the keyboard. The second function has a set as input parameter and prints it to the screen. The third function has a nonempty set as input parameter and returns its standard deviation. The fourth function has a nonempty set as input parameter and returns the sum of its even values, which has to be computed recursively. The fifth function has a nonempty set as input parameter and returns the sum of its odd values, which has to be computed recursively.

Scrivere una libreria ANSI C che gestisce insiemi di numeri naturali esportando le seguenti cinque funzioni. La prima funzione restituisce un insieme acquisito dalla tastiera. La seconda funzione ha come parametro di ingresso un insieme e lo stampa sullo schermo. La terza funzione ha come parametro di ingresso un insieme non vuoto e restituisce il suo scarto quadratico medio. La quarta funzione ha come parametro di ingresso un insieme non vuoto e restituisce la somma dei suoi valori pari, la quale deve essere calcolata ricorsivamente. La quinta funzione ha come parametro di ingresso un insieme non vuoto e restituisce la somma dei suoi valori dispari, la quale deve essere calcolata ricorsivamente.

Project for the 2009/2010 summer session

(PE – CPro)

Write an ANSI C library that manages sets of integer numbers by exporting the following five functions. The first function returns a set introduced through the keyboard. The second function has a set as input parameter and prints it to the screen. The third function has a nonempty set as input parameter and returns its median value. The fourth function has a nonempty set as input parameter and returns its maximum value, which has to be computed recursively. The fifth function has a nonempty set as input parameter and returns its minimum value, which has to be computed recursively.

Scrivere una libreria ANSI C che gestisce insiemi di numeri interi esportando le seguenti cinque funzioni. La prima funzione restituisce un insieme acquisito dalla tastiera. La seconda funzione ha come parametro di ingresso un insieme e lo stampa sullo schermo. La terza funzione ha come parametro di ingresso un insieme non vuoto e restituisce il suo valore mediano. La quarta funzione ha come parametro di ingresso un insieme non vuoto e restituisce il suo valore massimo, il quale deve essere calcolato ricorsivamente. La quinta funzione ha come parametro di ingresso un insieme non vuoto e restituisce il suo valore minimo, il quale deve essere calcolato ricorsivamente.

Project for the 2009/2010 winter session

(PE – CPro)

Write an ANSI C library that manages sets of integer numbers by exporting the following five functions. The first function returns a set introduced through the keyboard. The second function has a set as input parameter and prints it to the screen. The third function has two sets as input parameters and returns their union. The fourth function has two sets as input parameters and returns their intersection. The fifth function has two sets as input parameters and returns their difference.

Scrivere una libreria ANSI C che gestisce insiemi di numeri interi esportando le seguenti cinque funzioni. La prima funzione restituisce un insieme acquisito dalla tastiera. La seconda funzione ha come parametro di ingresso un insieme e lo stampa sullo schermo. La terza funzione ha come parametri di ingresso due insiemi e restituisce la loro unione. La quarta funzione ha come parametri di ingresso due insiemi e restituisce la loro intersezione. La quinta funzione ha come parametri di ingresso due insiemi e restituisce la loro differenza.

Project for the 2008/2009 autumn session

(PE – CPro)

Write an ANSI C program that gets from the keyboard two sequences of upper-case letters, stores them into two lists of characters after putting them into reverse alphabetical order, carries out their ordered fusion into a single array of characters, and finally prints to the scren the contents of the array (e.g., if the sequences are "EXIT" and "ZERO", then "ZXTROIEE" shall be printed).

Scrivere un programma ANSI C che acquisisce dalla tastiera due sequenze di lettere maiuscole, le memorizza in due liste di caratteri dopo averle messe in ordine alfabetico inverso, le fonde ordinatamente in un unico array di caratteri, e infine stampa sullo schermo il contenuto dell'array (p.e., se le sequenze sono "USCITE" e "ZERO", allora dovrà essere stampato "ZUTSROIEEC").

Project for the 2008/2009 summer session

(PE – CPro)

Write an ANSI C program that gets from the keyboard two sequences of lower-case letters, stores them into two arrays of characters after putting them into alphabetical order, carries out their ordered fusion into a single list, and finally prints to the screen the contents of the list (e.g., if the sequences are "exit" and "zero", then "eeiortxz" shall be printed).

Scrivere un programma ANSI C che acquisisce dalla tastiera due sequenze di lettere minuscole, le memorizza in due array di caratteri dopo averle messe in ordine alfabetico, le fonde ordinatamente in un'unica lista, e infine stampa sullo schermo il contenuto della lista (p.e., se le sequenze sono "uscite" e "zero", allora dovrà essere stampato "ceeiorstuz").

Project for the 2008/2009 winter session

(PE – CPro)

Write an ANSI C program that gets from the keyboard a word, recursively builds in a string variable the palindrome corresponding to the word (e.g., the palindrome of "hallo" is "halloollah"), and prints it to the screen by overriding its vowels.

Scrivere un programma ANSI C che acquisisce dalla tastiera una parola, costruisce ricorsivamente in una variabile stringa il palindromo corrispondente alla parola (p.e., il palindromo di "ciao" è "ciaooaic") e lo stampa sullo schermo omettendone le vocali.

Project for the 2007/2008 autumn session

(PE – CPro)

Write an ANSI C library that exports the following five functions, each of which has as input parameters two integer numbers interpreted as truth values and returns 1 or 0. The first function computes the logical conjunction of its two parameters. The second function computes the logical disjunction of its two parameters. The third function computes the exclusive logical disjunction of its two parameters. The fourth function computes the logical implication of its two parameters. The fifth function computes the logical double implication of its two parameters.

Scrivere una libreria ANSI C che esporta le seguenti cinque funzioni, ciascuna delle quali ha come parametri di ingresso due numeri interi interpretati come valori di verità e restituisce 1 *oppure* 0. *La prima funzione calcola la congiunzione logica dei suoi due parametri. La seconda funzione calcola la disgiunzione logica dei suoi due parametri. La terza funzione calcola la disgiunzione logica esclusiva dei suoi due parametri. La quarta funzione calcola l'implicazione logica dei suoi due parametri. La quinta funzione calcola la doppia implicazione logica dei suoi due parametri.*

Project for the 2007/2008 summer session
(PE – CPro)

Write an ANSI C program that gets from the keyboard a logical formula in conjunctive normal form and establishes whether the formula is satisfiable, i.e., whether there exists an assignment of truth values to the propositions occurring in the formula that makes the formula hold true.

Scrivere un programma ANSI C che acquisisce dalla tastiera una formula logica in forma normale congiuntiva e stabilisce se la formula è soddisfacibile, cioè se esiste un assegnamento di valori di verità alle proposizioni presenti nella formula che rende la formula vera.

Project for the 2007/2008 winter session

(PE – CPro)

Write an ANSI C program that gets from the keyboard a polynomial of degree at most 2 and computes its roots.

Scrivere un programma ANSI C che acquisisce dalla tastiera un polinomio di grado al più 2 *e ne calcola le radici.*

Project for the 2006/2007 autumn session

(PE – CPro)

Write an ANSI C program that gets from the keyboard a positive integer number and rewrites it as a sum of powers of 2.

Scrivere un programma ANSI C che acquisisce dalla tastiera un numero intero positivo e lo riscrive come somma di potenze di 2.

Project for the 2006/2007 summer session

(PE – CPro)

Write an ANSI C program that gets from the keyboard a sequence of positive integer numbers and computes their arithmetic average, their greatest common divisor, and their least common multiplier.

Scrivere un programma ANSI C che acquisisce dalla tastiera una sequenza di numeri interi positivi e ne calcola la media aritmetica, il massimo comun divisore e il minimo comune multiplo.

Project for the 2006/2007 winter session

(PE – CPro)

Goldbach's conjecture claims that every integer number that is even and greater than 2 can be written as the sum of two prime numbers. Write an ANSI C program that gets from the keyboard an even integer number greater than two and checks whether it satisfies Goldbach's conjecture.

La congettura di Goldbach asserisce che ogni numero intero pari maggiore di 2 *possa essere scritto come la somma di due numeri primi. Scrivere un programma ANSI C che acquisisce dalla tastiera un numero intero pari maggiore di due e verifica se esso soddisfa la congettura di Goldbach.*

Project for the 2005/2006 autumn session

(PE – CPro)

Write an ANSI C library that exports the following two functions. The first one has as input parameter a positive integer number and establishes whether it is prime or not. The second one has as input parameter a positive integer number and rewrites it as product of prime numbers.

Scrivere una libreria ANSI C che esporta le due seguenti funzioni. La prima ha come parametro di ingresso un numero intero positivo e stabilisce se è primo oppure no. La seconda ha come parametro di ingresso un numero intero positivo e lo fattorizza in numeri primi.

Project for the 2005/2006 summer session
(PE – CPro)

Write an ANSI C program that gets from the keyboard a binary, octal, or hexadecimal number and converts it into the corresponding decimal number, or gets from the keyboard a decimal number and converts it into the corresponding binary, octal, and hexadecimal numbers. Consider only non-negative integer numbers.

Scrivere un programma ANSI C che acquisisce dalla tastiera un numero binario, ottale o esadecimale e lo converte nel corrispondente numero decimale, oppure acquisisce dalla tastiera un numero decimale e lo converte nei corrispondenti numeri binario, ottale ed esadecimale. Si considerino solo numeri interi non negativi.

Project for the 2005/2006 winter session

(PE – CPro)

A simple cryptographic scheme consists of exchanging the letters of the alphabet through a translation table for the 52 upper-case and lower-case letters. Write an ANSI C library that exports the following two functions. The first one gets from the keyboard the name of a file to be read and then writes onto the file called `encoded.txt` the contents of the first file encoded on the basis of the previously explained scheme. The second one gets from the keyboard the name of a file to be read and then writes onto the file called `decoded.txt` the contents of the first file decoded on the basis of the previously explained scheme.

Un semplice schema di crittografia consiste nello scambiare le lettere dell'alfabeto l'una con l'altra, mediante una tabella di traduzione per le 52 *lettere maiuscole e minuscole. Scrivere una libreria ANSI C che esporta le due seguenti funzioni. La prima acquisisce dalla tastiera il nome di un file da leggere e poi scrive sul file chiamato* `codificato.txt` *il contenuto del primo file codificato in base allo schema precedentemente spiegato. La seconda acquisisce dalla tastiera il nome di un file da leggere e poi scrive sul file chiamato* `decodificato.txt` *il contenuto del primo file decodificato in base allo schema precedentemente spiegato.*

Project for the 2004/2005 autumn session

(PE – CPro)

A sparse matrix is a matrix in which most elements hold zero. A sparse matrix can conveniently be represented by means of lists (related to the rows or the columns) that contain the non-zero values, instead of a bidimensional array. Write an ANSI C library that exports four functions for reading a sparse matrix from the keyboard, printing a sparse matrix to the screen, computing the sum of two sparse matrices, and computing the product of two sparse matrices.

Una matrice sparsa è una matrice nella quale la maggior parte degli elementi valgono zero. Una matrice sparsa può essere convenientemente rappresentata utilizzando delle liste (relative alle righe o alle colonne) che contengono i valori non nulli, anziché un array bidimensionale. Scrivere una libreria ANSI C che esporta quattro funzioni per la lettura dalla tastiera di una matrice sparsa, la stampa sullo schermo di una matrice sparsa, il calcolo della somma di due matrici sparse e il calcolo del prodotto di due matrici sparse.

Project for the 2004/2005 summer session

(PE – CPro)

Write an ANSI C program that gets from the keyboard a sequence composed only of the symbols x, 0, and 1, computes recursively all the binary numbers that can be obtained from the sequence by replacing each occurrence of x with 0 or 1, and prints to the screen such numbers (e.g., if the sequence is $1x0x$, the numbers that can be obtained are 1000, 1001, 1100, and 1101).

Scrivere un programma ANSI C che acquisisce dalla tastiera una sequenza costituita dai soli simboli x, 0 *e* 1, *calcola ricorsivamente tutti i numeri in base* 2 *ottenibili dalla sequenza sostituendo ogni occorrenza di* x *con* 0 *oppure* 1 *e stampa sullo schermo tali numeri (p.e., se la sequenza è* $1x0x$, *i numeri ottenibili sono* 1000, 1001, 1100 *e* 1101*).*

Project for the 2004/2005 winter session

(PE – CPro)

Write an ANSI C program that takes from the command line (i.e., through the parameters of function `main`) the name of a text file and produces a new text file such that:

- The first line of the new file contains the name of the old file together with a newly added `.lis` suffix, with the extended name being preceded and followed by 10 asterisks.
- Any other line of the new file is equal to the corresponding line of the old file, preceded by the line number enclosed in parentheses plus a blank.

Scrivere un programma ANSI C che prende dalla linea di comando (cioè tramite i parametri della funzione `main`*) il nome di un file di testo e produce un nuovo file di testo tale che:*

- *La prima linea del nuovo file contiene il nome del vecchio file con l'aggiunta del suffisso* `.lis`*, col nome così esteso preceduto e seguito da* 10 *asterischi.*
- *Ogni altra linea del nuovo file è uguale alla corrispondente linea del vecchio file, preceduta dal numero di linea racchiuso tra parentesi tonde più uno spazio.*

Project for the 2003/2004 autumn session
(PE – CPro)

Write an ANSI C library that exports the following five functions for:

- Getting from the keyboard a polynomial and storing it into a suitable data structure.
- Computing the sum of two polynomials and storing the resulting polynomial into a suitable data structure.
- Computing the difference of two polynomials and storing the resulting polynomial into a suitable data structure.
- Computing the product of two polynomials and storing the resulting polynomial into a suitable data structure.
- Printing to the screen a polynomial previously stored into a suitable data structure.

Scrivere una libreria ANSI C che esporta le seguenti cinque funzioni per:

- *Acquisire dalla tastiera un polinomio e memorizzarlo in un'apposita struttura dati.*
- *Calcolare la somma di due polinomi e memorizzare il polinomio risultante in un'apposita struttura dati.*
- *Calcolare la differenza di due polinomi e memorizzare il polinomio risultante in un'apposita struttura dati.*
- *Calcolare il prodotto di due polinomi e memorizzare il polinomio risultante in un'apposita struttura dati.*
- *Stampare sullo schermo un polinomio precedentemente memorizzato in un'apposita struttura dati.*

Project for the 2003/2004 summer session

(PE – CPro)

Write an ANSI C program that gets from the keyboard a set of integer numbers different from each other, computes recursively its power set (i.e., the set of all the possible subsets of the considered set), and prints to the screen the power set.

Scrivere un programma ANSI C che acquisisce dalla tastiera un insieme di numeri interi diversi tra loro, ne calcola ricorsivamente l'insieme delle parti (cioè l'insieme di tutti i possibili sottoinsiemi dell'insieme considerato) e stampa sullo schermo l'insieme delle parti.

Project for the 2003/2004 winter session

(PE – CPro)

Write an ANSI C program that implements the following game between two players:

- In the first phase, the program asks the first player to introduce a certain number of words (in English). More precisely, the program starts by asking the first player the number of words to be introduced (at least 10), then gets them one at a time by verifying that each is formed only by the 26 lower-case letters of the alphabet (every wrong word has to be introduced again as long as it is not correct).
- In the second phase, the program randomly chooses a word among those introduced by the first player, then asks the second player to guess it by providing the player with a number of attempts equal to the number of letters that constitute the word. The word must initially be visualized as a sequence of asterisks, which are then transformed into the corresponding letters as far as the second player guesses them. Once guessed the word or run out of attempts, the program asks the second player whether he/she wishes to guess another word.

Scrivere un programma ANSI C che realizza il seguente gioco tra due giocatori:

- *Nella prima fase, il programma chiede al primo giocatore di inserire un certo numero di parole (in italiano). Per la precisione, il programma inizia chiedendo al primo giocatore il numero di parole che vuole inserire (almeno* 10*), poi le acquisisce una alla volta verificando che ciascuna sia formata solo dalle* 26 *lettere minuscole dell'alfabeto (ogni parola errata va introdotta di nuovo finché non è corretta).*
- *Nella seconda fase, il programma sceglie a caso una parola tra quelle inserite dal primo giocatore, poi chiede al secondo giocatore di indovinarla mettendogli a disposizione un numero di tentativi pari al numero di lettere che costituiscono la parola. La parola deve inizialmente essere visualizzata come una sequenza di asterischi, che vengono poi tramutati nelle corrispondenti lettere man mano che il secondo giocatore le indovina. Una volta indovinata la parola oppure esauriti i tentativi a disposizione, il programma chiede al secondo giocatore se vuole indovinare un'altra parola.*

Project for the 2002/2003 autumn session

(PE – CPro)

A picture can be represented through a bidimensional grid of pixels (picture elements), whose values can be 0 (empty pixel) and 1 (full pixel). In this context, a figure is constituted by a set of full pixels that are adjacent to each other (horizontally, vertically, or diagonally). Write an ANSI C program that gets from the keyboard the size of the grid, generates its contents at random, and then, given the coordinates of a pixel, prints to the screen the area of the figure in which the specified pixel is located, i.e., the number of pixels that constitute the figure in which the specified pixel is located.

Una fotografia può essere rappresentata tramite una griglia bidimensionale di pixel (picture element), i cui valori possono essere 0 *(pixel vuoto) e* 1 *(pixel pieno). In questo contesto, una figura è costituita da un insieme di pixel pieni adiacenti (in orizzontale, verticale o diagonale). Scrivere un programma ANSI C che acquisisce dalla tastiera la dimensione della griglia, ne genera il contenuto in maniera casuale e poi, date le coordinate di un pixel, stampa sullo schermo l'area della figura in cui si trova il pixel specificato, cioè il numero di pixel che costituiscono la figura in cui si trova il pixel specificato.*

Project for the 2002/2003 summer session

(PE – CPro)

Write an ANSI C library that exports five functions that respectively implement the operations of addition, subtraction, multiplication, division, and raising for two natural numbers, under the following constraints:

- The five functions must preliminarily carry out suitable checks to guarantee that the two numbers on which they work are natural numbers and that the result is well defined.
- The core of the computation of each of the five functions (i.e., the part following the checks) must be recursive.
- The only ANSI C arithmetical operators that can be used in the five functions are the unary operators of increment/decrement by one unit.

Scrivere una libreria ANSI C che esporta cinque funzioni che realizzano rispettivamente le operazioni di addizione, sottrazione, moltiplicazione, divisione ed elevamento a potenza per due numeri naturali, sotto i seguenti vincoli:

- *Le cinque funzioni debbono preventivamente effettuare gli opportuni controlli per garantire che i due numeri su cui esse operano siano numeri naturali e che il risultato sia ben definito.*
- *La parte di calcolo vero e proprio di ciascuna delle cinque funzioni (cioè la parte successiva ai controlli) deve essere ricorsiva.*
- *Gli unici operatori aritmetici ANSI C utilizzabili nelle cinque funzioni sono gli operatori unari di incremento e decremento di una unità.*

Project for the 2002/2003 winter session

(PE – CPro)

A simple arithmetical expression in ANSI C is constituted by identifiers of variables of type int or double, numerical constants of type int or double, the two unary arithmetical operators, the five binary arithmetical operators (two additive ones and three multiplicative ones), and the parentheses. Write an ANSI C program that, after getting from the keyboard a simple arithmetical expression, prints to the screen a simple arithmetical expression obtained from the given one by inserting parentheses that reflect the precedence and the associativity of the operators. Moreover, the program must print to the screen the type of the given expression (assume that the variables that begins with a letter between 'a' and 'l' are of type int while the others are of type double).

Un'espressione aritmetica semplice in ANSI C è costituita da identificatori di variabili di tipo int o double, costanti numeriche di tipo int o double, i due operatori aritmetici unari, i cinque operatori aritmetici binari (due additivi e tre moltiplicativi) e le parentesi tonde. Scrivere un programma ANSI C che, acquisita dalla tastiera un'espressione aritmetica semplice, stampa sullo schermo un'espressione aritmetica semplice ottenuta da quella data inserendo delle parentesi che riflettono la precedenza e l'associatività degli operatori. Il programma deve inoltre stampare sullo schermo il tipo dell'espressione data (assumere che le variabili che iniziano con una lettera compresa tra 'a' ed 'l' siano di tipo int mentre le altre siano di tipo double).

Project for the 2001/2002 autumn session

(PE – CPro)

Given a board with 10 rows and 10 columns, every square can be unoccupied or can contain one organism. Every square – excluding those at the boundary – is considered to have 8 adjacents squares, which are those that surround it. Given a generation of organisms, i.e., a configuration of the board, the next generation is determined according to the following rules:

- An organism can be born in every empty square that has exactly three adjacent squares containing an organism.
- An organism with more than three adjacent organisms dies due to overcrowding.
- An organism with less than two adjacent organisms dies due to loneliness.
- An organism with two or three adjacent organisms survive.

Write an ANSI C program that, after getting from the keyboard the initial generation and the number of subsequent generations, prints to the screen all the desired subsequent generations.

Data una scacchiera con 10 *righe e* 10 *colonne, ogni casella può essere libera oppure può contenere un organismo. Ogni casella – escluse quelle ai bordi – è considerata avere* 8 *caselle adiacenti, che sono quelle che la circondano. Data una generazione di organismi, ovvero una configurazione della scacchiera, la generazione successiva viene determinata in base alle seguenti regole:*

- *Un organismo può nascere in ogni casella vuota che ha esattamente tre caselle adiacenti contenenti un organismo.*
- *Un organismo con più di tre organismi adiacenti muore per sovraffollamento.*
- *Un organismo con meno di due organismi adiacenti muore di solitudine.*
- *Un organismo con due o tre organismi adiacenti sopravvive.*

Scrivere un programma ANSI C che, acquisiti dalla tastiera la generazione iniziale e il numero di generazioni successive, stampa sullo schermo tutte le generazioni successive desiderate.

Project for the 2001/2002 summer session
(PE – CPro)

Write an ANSI C library that exports:

- A function for getting from the keyboard a fixed-point real number and transforming it into the equivalent floating-point real number. The latter number must be stored into an array of 32 integers where:
  - $a[0]$ holds 0 is the number is not negative, 1 otherwise.
  - $a[1]$ holds 0 is the exponent is not negative, 1 otherwise.
  - $a[2], \ldots, a[7]$ respectively hold the decimal digits that constitute the absolute value of the exponent, aligned to $a[7]$.
  - $a[8], \ldots, a[31]$ respectively hold the decimal digits that constitute the fraction, aligned to $a[8]$ (note that $a[8]$ cannot be the digit 0 except for the case that the number is 0).
- Four functions that implement the four arithmetical operations over floating-point real numbers. These functions must signal possible overflow problems (absolute value of the exponent not representable with only 6 decimal digits) and truncation problems (value of the fraction not representable with only 24 decimal digits).
- A function for printing to the screen the fixed-point real number equivalent to a floating-point real number.

Scrivere una libreria ANSI C che esporta:

- *Una funzione per acquisire dalla tastiera un numero reale in virgola fissa e trasformarlo nell'equivalente numero reale in virgola mobile. Quest'ultimo numero deve essere memorizzato in un array di* 32 *interi dove:*
  - $a[0]$ *vale* 0 *se il numero non è negativo,* 1 *altrimenti.*
  - $a[1]$ *vale* 0 *se l'esponente non è negativo,* 1 *altrimenti.*
  - $a[2], \ldots, a[7]$ *contengono rispettivamente le cifre decimali che costituiscono il valore assoluto dell'esponente, con allineamento ad* $a[7]$.
  - $a[8], \ldots, a[31]$ *contengono rispettivamente le cifre decimali che costituiscono la mantissa, con allineamento ad* $a[8]$ *(notare che* $a[8]$ *non può essere la cifra* 0 *ad eccezione del caso in cui il numero reale in virgola mobile sia* 0*).*
- *Quattro funzioni che implementano le quattro operazioni aritmetiche sui numeri reali in virgola mobile. Queste funzioni debbono segnalare eventuali problemi di overflow (valore assoluto dell'esponente non rappresentabile con sole* 6 *cifre decimali) e troncamento (valore della mantissa non rappresentabile con sole* 24 *cifre decimali).*
- *Una funzione per stampare sullo schermo il numero reale in virgola fissa equivalente a un numero reale in virgola mobile.*

Project for the 2001/2002 winter session

(PE – CPro)

A file contains a sequence of non-negative integer numbers. The first pair of numbers refers to the births in the month of January of the current year and the previous year, the second pair of numbers refers to the month of February, and so on. The strings that are not non-negative numbers have to be left out. If the file contains less than two valid numbers, no processing can take place. If the file contains more than 24 valid numbers, only the first 24 have to be considered. If the file contains an odd number of valid numbers between 3 and 23, the last valid number has to be left out. Write an ANSI C program that:

- Prints to the screen the numbers of births in tabular form. The table must contain two rows (current year and previous year) and as many columns as there are months for which data is available. A generic element of coordinates (i, j) of the table holds the number of births in month j of year i.
- Prints to the screen the total number of births in each of the two years and the average number of births per month in each of the two years.
- Prints to the screen for each month of the current year for which data is available its name followed by the related number of births, in decreasing order with respect to the number of births. If several months have the same number of births, the related information have to appear according to the temporal order of the months.

Un file contiene una sequenza di numeri interi non negativi. La prima coppia di numeri è relativa alle nascite avvenute nel mese di gennaio dell'anno corrente e dell'anno precedente, la seconda coppia di numeri è relativa al mese di febbraio, e così via. Le stringhe non riconducibili a numeri interi non negativi vanno ignorate. Se il file contiene meno di due numeri validi, nessuna elaborazione può avere luogo. Se il file contiene più di 24 *numeri validi, solo i primi* 24 *debbono essere considerati. Se il file contiene un numero dispari di numeri validi compreso tra* 3 *e* 23*, l'ultimo numero valido va ignorato. Scrivere un programma ANSI C che:*

- *Stampa sullo schermo i numeri delle nascite in forma tabellare. La tabella deve contenere due righe (anno corrente e anno precedente) e tante colonne quanti sono i mesi per i quali si hanno dati a disposizione. Un generico elemento di coordinate (i, j) della tabella riporta il numero di nascite avvenute durante il mese j dell'anno i.*
- *Stampa sullo schermo il numero totale di nascite avvenute in ciascuno dei due anni e il numero medio di nascite per mese avvenute in ciascuno dei due anni.*
- *Stampa sullo schermo per ciascun mese dell'anno corrente per il quale si hanno dati a disposizione il suo nome seguito dal relativo numero di nascite, in ordine decrescente rispetto al numero di nascite. Se più mesi hanno lo stesso numero di nascite, le relative informazioni debbono comparire secondo l'ordine temporale dei mesi.*