# Informatica A

**Cognome Matricola**

**Nome Firma**

### Istruzioni

* Non separate questi fogli. Scrivete la soluzione **solo** **sui fogli distribuiti**, utilizzando il retro delle pagine in caso di necessità. **Cancellate le parti di brutta** (o ripudiate) con un tratto di **penna**.
* Ogni parte non cancellata a penna sarà considerata parte integrante della soluzione.
* **È possibile scrivere a matita** (e non ricalcare al momento della consegna!).
* È **vietato** utilizzare **calcolatrici** o **telefoni**. Chi tenti di farlo vedrà **annullata** la sua prova.
* È ammessa la consultazione di **libri** e **appunti**, purché con pacata discrezione e senza disturbare.
* Qualsiasi **tentativo** di comunicare con altri studenti comporta **l’espulsione** dall’aula.
* È possibile **ritirarsi senza penalità**.
* Non è possibile lasciare l’aula conservando il tema della prova in corso.
* Tempo a disposizione: **2 h 30 m**

### Valoredegli esercizi, voti parziali e voto finale:

**Esercizio 1 ( 2 punti )**

**Esercizio 2 ( 4 punti )**

**Esercizio 3 ( 4 punti )**

**Esercizio 4 ( 7 punti )**

**Esercizio 5 ( 8 punti )**

**Esercizio 6 ( 4 punti )**

**Totale: ( 29 punti )**

## Esercizio 1 - Algebra di Boole, Aritmetica Binaria, Codifica delle Informazioni (2 punti)

1. Si costruisca la tabella di verità della seguente espressione booleana, badando alla precedenza tra gli operatori logici. (1 punto).

**( A or B or A ) and ( ( B and ( not C ) ) and C )**

1. Si stabilisca il minimo numero di bit sufficiente a rappresentare in complemento a due i numeri A = 26dec e B = –81dec, li si converta, se ne calcoli la differenza (A–B) in complemento a due e si indichi se si genera riporto sulla colonna dei bit più significativi e se si verifica overflow (1 punto).

## Esercizio 2 ( 4 punti )

Dato il seguente schema di base di dati relazionale relativo a menù serviti in vari ristoranti:

RISTORANTE(PartitaIva, Nome, Indirizzo, Città, GiornoChiusura, CodiceFiscaleProprietario)

MENU(Codice, PIvaRistorante, Nome, Prezzo)

COMPOSIZIONE(CodiceMenu, Portata)

PORTATA(Nome, Descrizione, Tipo, Calorie, Prezzo)

INGREDIENTE(NomePortata, Ingrediente, Quantita)

1. Esprimere in SQL e nei tre linguaggi formali la query che trova le diverse città che non offrono nessun menù a meno di 10 euro.

RISTORANTE(PartitaIva, Nome, Indirizzo, Città, GiornoChiusura, CodiceFiscaleProprietario)

MENU(Codice, PIvaRistorante, Nome, Prezzo)

COMPOSIZIONE(CodiceMenu, Portata)

PORTATA(Nome, Descrizione, Tipo, Calorie, Prezzo)

INGREDIENTE(NomePortata, Ingrediente, Quantita)

2. Esprimere in SQL la query che trova i ristoranti che hanno solo menù in cui complessivamente le portate hanno in totale meno di 1000 calorie.

**Esercizio 3** ( 4 punti )

Si dica cosa stampa il seguente codice

#define N 2

#include<stdio.h>

#include<conio.h>

#include<string.h>

void mistero(char \* s, int k) {

char v[N\*3];

strcpy(v,s);

printf("%c",v[0]);

if( k > 0 && strlen(v)>N )

mistero(v+N,k);

return;

}

int main() {

char s[] = "AOZNI";

int i;

for(i=0;i<=N;i++)

mistero(&s[i%N],N-i);

getch();

return 0;

}

## Esercizio 4 ( 7 punti )

Si scriva una funzione

Lista f(char M[][N], int R)

che riceve in ingresso una matrice M quadrata NxN (con N costante predefinita con l’istruzione #define N …) di caratteri e un intero R. La funzione verifica che esistano almeno R caratteri distinti nella matrice. In caso negativo restituisce NULL, in caso positivo invece restituisce una lista contenente i caratteri distinti e il numero di volte in cui ognuno di essi è presente nella matrice (si ponga attenzione alla struttura del nodo della lista che dovrà essere restituita).

## Esercizio 5 ( 8 punti )

#define N 5

**typedef** **char** \* nome; //*una stringa*

**typedef** nome\*elencocandidati**;** //*il tipo di un vettore di puntatori a carattere*

**typedef** nome candidati[N]; //*un vettore di N puntatori a carattere*

**typedef** **struct** lis { **int** numLista;

elencocandidati elenconomi; //”*punta a” un vettore di tipo* candidati

**struct** lis \* next; } listacandidati;

**typedef** listacandidati \* lista; //*una lista di listacandidati (una per lista)*

La struttura definita sopra supporta un metodo di voto elettronico per i rappresentanti degli studenti in Senato Accademico. Le liste sono identificate da numeri interi (per comodità, da 0 a quante sono le liste meno uno) e ogni lista presenta esattamente N candidati (tanti quanti i seggi in Senato).

Si consideri la seguente definizione per (una lista di) schede elettorali scrutinate. Ogni scheda contiene l’indicazione della lista votata e l’eventuale preferenza (che ha valore NULL se non espressa).

**typedef** **struct** sc { **int** numLista;

**char** \* preferenza;

**struct** sc \* next; } scheda;

**typedef** scheda \* listascrutinio; // *una lista di schede scrutinate*

Si consideri ora il conteggio delle preferenze dopo lo scrutinio dei voti, con le schede già inserite in una listascrutinio. Si codifichi in C la funzione

**int** piuVotato(nome n, **int** numLista, listascrutinio ls, lista L)

che restituisce il numero di preferenze ottenute dal candidato più votato della lista numLista e inserisce nella stringa n (inizialmente passata alla funzione con valore ‘\0’) il suo nome.

## Esercizio 6 ( 4 punti )

Si consideri la seguente definizione di albero

**typedef struct EL { int dato;**

**struct EL \* left, \* right; } node;**

**typedef node \* tree;**

Si scriva una funzione che riceve in ingresso un albero T e restituisce 1 se esiste almeno un cammino a partire dalla radice fino a una qualsiasi delle foglie lungo il quale il valore del campo **dato** sia strettamente decrescente, 0 altrimenti.