# Informatica A

**Cognome (IN STAMPATELLO)**

**Nome**

**Matricola o Codice Persona**

Istruzioni

1. Non separate questi fogli. Scrivete la soluzione **solo** **sui fogli distribuiti**, utilizzando il retro delle pagine in caso di necessità.
2. **È possibile scrivere a matita** (e non ricalcare al momento della consegna).
3. Ogni parte non cancellata sarà considerata parte integrante della soluzione.
4. È **vietato** utilizzare **calcolatrici** o **telefoni**. Chi tenti di farlo vedrà **annullata** la sua prova.
5. Non è ammesso consultare **libri** o **appunti** o altro.
6. Qualsiasi **tentativo** di comunicare con altri studenti comporta **l’espulsione** dall’aula.
7. È possibile **ritirarsi senza penalità** lasciando il tema d’esame con nome e cognome.
8. Non è possibile lasciare l’aula conservando il tema della prova in corso.
9. Tempo a disposizione: 2 ore e 30 minuti

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

**Esercizio 1 ( 2 punti )**

**Esercizio 2 ( 4 punti )**

**Esercizio 3 ( 4 punti )**

**Esercizio 4 ( 8 punti )**

**Esercizio 5 (10 punti )**

**Esercizio 6 ( 4 punti )**

**Totale: ( 32 punti )**

## Esercizio 1 (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 and ( not C ) ) ) and ( not A or ( (not B) and C )**

1. Si stabilisca il minimo numero di bit sufficiente a rappresentare in complemento a due entrambi i numeri A = 119dec e B = –95dec

Si converta A e B in CP2 usando bit

Si calcoli quindi (A+B) e (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 )

Il seguente schema rappresenta le informazioni riguardo al noleggio di automobili:

Auto ( Targa, Modello, Marca )

Cliente ( CF, Nome, Cognome, Indirizzo, Telefono, Convenzione )

Noleggio ( CF, Targa, DataInizio, DataFine, CostoAlGiorno, Giorni, Km )

Scrivere una query che estrae il clienti che ha noleggiato più modelli di auto diversi nel 2020.

Auto ( Targa, Modello, Marca )

Cliente ( CF, Nome, Cognome, Indirizzo, Telefono, Convenzione )

Noleggio ( CF, Targa, DataInizio, DataFine, CostoAlGiorno, Giorni, Km )

Scrivere una query che estrae i clienti che non hanno mai noleggiato auto che costano più di 100 euro al giorno.

**Esercizio 3** ( 4 punti )

Si dica cosa stampa il seguente codice

**#include <stdio.h>**

**#include <stdlib.h>**

**typedef struct el {**

**char c;**

**struct el \* next; } nodo;**

**void f(char \* str, nodo \*\* n, int i) {**

**int d = 0;**

**if(i > 12)**

**return;**

**if(i==0)**

**(\*(\*n+i)).c = 'A';**

**else**

**switch(i % 4)**

**{**

**case 0:**

**(\*(\*n+i)).c = (\*(\*n + i - 1)).c;**

**break;**

**case 1:**

**(\*(\*n+i)).c = '-';**

**break;**

**case 2:**

**(\*(\*n+i)).c = '>';**

**break;**

**case 3:**

**(\*(\*n+i)).c = (\*(\*n + i - 3)).c + str[0] - '0';**

**d = 1;**

**break;**

**}**

**f( str+d, n, i+1 );**

**}**

**int main() {**

**char str[3];**

**int dim,i;**

**printf("Inserire le ultime tre cifre del tuo codice persona");**

**scanf("%s", str);**

**printf("%s", str);**

**dim = sizeof(nodo);**

**nodo \* a = (nodo \*)malloc(3 \* 4 \* dim);**

**f(str, &a, 0 );**

**for(i = 0; i <12; i++)**

**printf("%c ", (a+i)-> c);**

**return 0;}**

## Esercizio 4 ( 8 punti )

Scrivere una funzione ricorsiva disposizioni che riceve in ingresso un array di char v, la dimensione n dell'array, un intero k e stampa a schermo tutte le disposizioni con ripetizioni degli n elementi di v in sequenze composte da k elementi.

Es. se v = {'a', 'b', 'c'} (quindi n = 3), k = 2;

ci sono 3^2 disposizioni con ripetizione

aa

ab

ac

ba

bb

bc

ca

cb

cc

Si scriva poi un frammento di codice che invoca la funzione disposizioni per stampare tutti i numeri binari distinti di k bit, dove k viene definito dall’utente.

## Esercizio 5 ( 10 punti )

Si consideri la seguente definizione di lista:

**typedef struct EL {**

**int dato;**

**struct EL \* next;**

**} nodo;**

**typedef nodo \* lista;**

Scrivere una funzione

**lista intreccia(lista lis1, lista lis2)**

che, prese in input due liste, le *intreccia* in modo tale che un elemento della prima lista sia sempre seguito da un elemento della seconda lista e viceversa. Nel caso in cui una delle due liste sia più lunga dell’altra, gli elementi rimanenti della lista devono comparire in coda alla lista intrecciata.

**Ad esempio:**

Se si forniscono in input:

lis1: 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> NULL

lis2:  7 -> 8 -> 9 -> 10 -> NULL

la funzione restituisce:

1 -> 7 -> 2 -> 8 -> 3 -> 9 -> 4 -> 10 -> 5 -> 6 -> NULL

Nota: Assumere per semplicità che le due liste siano non vuote.

Si codifichi una funzione che **cancellaElementiGrandi** che rimuove da una lista tutti gli elementi maggiori della somma degli elementi seguenti

Ad esempio

Input: 1 -> 40 -> 3 -> 15 -> 5 -> 6 -> NULL

La funzione restituisce

1 -> 3 -> 5 -> 6 -> NULL

Esercizio 6 **( 4 punti )**

Si consideri la seguente di albero binario:

typedef struct nodeS { int v;

struct nodeS \* left, right; } node;

typedef node \* tree;

Si scriva una funzione “simmetrici” che dati due alberi binari restituisce 1 se i due alberi sono simmetrici, 0 altrimenti.

int simmetrici(Tree t1, Tree t2)

Esempio:

1

|

|-------------|

3 2

|----------|

5 4

È simmetrico con

1

|---------------|

2 3

|-------|

4 5