



Politecnico di Milano

## Dipartimento di Elettronica, Informazione e Bioingegneria

**Informatica A - a.a 2020/2021 - 16 luglio 2021**

**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** con un tratto di **penna**.
- Ogni parte non cancellata a penna sarà considerata parte integrante della soluzione.
- **NON è possibile scrivere a matita.**
- Scrivere nome e cognome su tutti i fogli. Non verranno corretti compiti non firmati o con nome illeggibile
- È **vietato** utilizzare **calcolatrici, telefoni o pc**. Chi tenti di farlo vedrà **annullata** la sua prova.
- **Non è ammessa la consultazione di libri e appunti.**
- 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: **2h30m**

### Valore indicativo degli esercizi, voti parziali e voto finale:

<b>Esercizio 1</b>	<b>6 punti</b>	_____
<b>Esercizio 2</b>	<b>7 punti</b>	_____
<b>Esercizio 3</b>	<b>6 punti</b>	_____
<b>Esercizio 4</b>	<b>13 punti</b>	_____
<b>Totale(32)</b>		_____

**Esercizio 1 - Algebra di Boole, Aritmetica Binaria, Codifica delle Informazioni (6 punti)**

**Punteggio totale quesiti a risposta multipla: 3 punti. Ogni domanda può avere da 0 a 4 risposte corrette. Risposte errate comporteranno rimozione di 0,5 punti.**

- (a) Qual è il risultato dell'espressione logica  
 $\text{NOT} ((\text{NOT } A \text{ OR } \text{NOT } B) \text{ AND } (\text{NOT } B \text{ AND } C)) \text{ OR } B \text{ AND } (A \text{ OR } B)$
- ☐ Un valore logico VERO
  - ☐ Non dipende da A
  - ☐  $A \text{ OR } B \text{ OR } \text{NOT } C$
  - ☐ Nessuna delle precedenti

**Risposta:**

Non dipende da A

- (b) Qual è il risultato dell'espressione logica  
 $\text{NOT} (\text{NOT } B \text{ AND } C \text{ AND } (A \text{ AND } B \text{ AND } (\text{NOT} (C \text{ OR } D))))$
- ☐ Non dipende da C
  - ☐ Un valore logico FALSO
  - ☐ Un valore logico VERO
  - ☐  $\text{NOT } A \text{ OR } \text{NOT } B \text{ OR } \text{NOT } C \text{ OR } D$

**Risposta:**

Non dipende da C Un valore logico VERO

- (c) Ho un insieme contenente  $n = 33$  numeri interi casuali tra -128 e 127. Quanti bit mi servono per codificarne gli elementi?
- ☐ 6 bit
  - ☐  $\log(128)+1$
  - ☐  $\log(N)-1$
  - ☐ 8 bit

**Risposta:**

Essendo 33 numeri, bastano  $\log_2(n) = 6$  bit

- (d) Si converta il numero decimale 123 in base 8 ed in base 16. Formattare la risposta in modo da descrivere il procedimento. (2 punti)

**Risposta:**

123 — 1

61 — 1

30 — 0

15 — 1  
7 — 1  
3 — 1  
1 — 1  
0

binario = 01111011

in base 8, i numeri vanno da 0 a 7. Partendo dal numero binario si considerano 3 bit alla volta

01 111 011  
1 7 3

base 8 = 173

In base sedici i numeri vanno da 0 a 15. Dal numero binario si considerano 4 bit alla volta per la conversione.

0111 1011  
7 11 = B

base 16 = 7B

(e) Si stabilisca il minimo numero di bit sufficiente a rappresentare in complemento a due i numeri

A = 37dec

B = 27dec

Li si converta, se ne calcolino la somma (A+B) e 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.

Formattare le risposte con spazi in modo che siano leggibili.

Non verranno accettate risposte senza procedimento. (1 punto)

**Risposta:**

37dec = 0100101b 27dec = 0011011b

37+27 =

0100101b 0011011b = 1000000b

No riporto perduto ma overflow. 1000000b = -64dec

37-27 =

0100101b 1100101b = 1 0001010b

Riporto perduto, no overflow.

**Esercizio 2 - Programmazione C (7 punti)**

Una macchina per il taglio laser planare (con piano di lavoro 50x50mm) ha la testina in grado di muoversi nel piano nelle quattro direzioni nord, sud, est, ovest (denotate rispettivamente dai caratteri 'N', 'S', 'E', 'W').

I comandi per il taglio vengono passati attraverso due array di lunghezza massimo 100 (la lunghezza reale è contenuta in una variabile LEN). I due array contengono rispettivamente la direzione di spostamento e i mm (numeri interi) di cui la testina si deve spostare.

Esempio:

```
array_direzione = ['N','E','S','W','S']  
array_spostamento = [105431]
```

1. Definire tutte le strutture dati necessarie alla gestione della macchina (1 punto)

**Risposta:**

```
int x,y;  
int mmArray[100] = {10,5,5,6};  
char directionArray[100] = {'N','E','S','W'};  
int LEN = 4;
```

2. Scrivere una funzione che dati in ingresso il numero di istruzioni ed i due array, restituisca se il taglio è realmente possibile in base alla dimensione del piano (si consideri come punto di partenza l'angolo basso a sinistra come posizione 0,0) (3 punti)

**Risposta:**

```
int checkCut(int mm[], char dir[], int len){  
    int x=0,y=0;  
  
    for (int i=0; i<len; i++){  
        switch (dir[i]) {  
            case 'N':  
                y = y+mm[i];  
                break;  
  
            case 'S':  
                y = y-mm[i];  
                break;  
  
            case 'W':  
                x = x-mm[i];  
                break;  
  
            case 'E':  
                x = x+mm[i];  
                break;  
        }  
        if ((x<0) || (x>50) || (y<0) || (y>50))  
            return 0;  
    }  
    return 1;  
}
```

```

    }
    return 1;
}

```

3. Scrivere una funzione che riceve in ingresso il numero di istruzioni ed i due array per gli spostamenti e stampa a video le coordinate del punto di arrivo. Ad esempio, nell'esempio presentato sopra il punto di arrivo sarebbe (N5, E2). La funzione restituisce anche una matrice di interi. In ciascuna casella della matrice viene posto il numero 0 se la testina non è passata di lì, e 1 se invece è passata di lì. (3 punti)

#### Risposta:

```

void taglio(int mm[], char dir[], int len, int*x, int *y, int cutMatrix[][DIM]){
    for (int j=0; j<DIM; j++)
        for (int k=0; k<DIM; k++)
            cutMatrix[j][k]=0;

    for (int i=0; i<len; i++){
        switch (dir[i]) {
            case 'N':
                for (;mm[i]>0 && *y<=DIM; (*y)++, mm[i]--){
                    cutMatrix[*x][*y] = 1;
                    break;
                }

            case 'S':
                for (;mm[i]>0 && *y>=0; (*y)--, mm[i]--){
                    cutMatrix[*x][*y] = 1;
                    break;
                }

            case 'W':
                for (;mm[i]>0 && *x>=0; (*x)--, mm[i]--){
                    cutMatrix[*x][*y] = 1;
                    break;
                }

            case 'E':
                for (;mm[i]>0 && *x<=DIM; (*x)++, mm[i]--){
                    cutMatrix[*x][*y] = 1;
                    break;
                }
        }
    }
}

```

**Esercizio 3 - Matlab (6 punti)**

Scrivere il codice Matlab che restituisca i valori richiesti. Attenersi al numero massimo di righe di codice indicato.

1. Creare una matrice quadrata A di dimensione casuale tra 3 e 10 contenente solo numeri pari a 7 (1 riga - 1 punto)

**Risposta:**

```
A = 7*ones(randi([3 10]))
```

2. Moltiplicare tutte le celle della matrice per un numero casuale tra 0 e 10. Ogni cella va moltiplicata per un numero differente (1 riga - 1 punto)

**Risposta:**

```
A = A.*randi([0 10],size(A))
```

3. Calcolare la media di tutte le righe della matrice (1 riga - 1 punto)

**Risposta:**

```
mean(A,2)
```

4. Sostituire ogni elemento della matrice pari a 0 con il valore -1 (1 riga - 2 punti)

**Risposta:**

```
A(A==0)=-1
```

5. Scrivere una funzione Matlab che presi in ingresso i valori a,b e c di una equazione di secondo grado, restituisca le radici dell'equazione tramite un vettore. La funzione controlla inoltre quante soluzioni reali esistono e restituisce:

- 0 se non esistono soluzioni reali  
1 se esistono due soluzioni coincidenti  
2 se esistono due soluzioni reali e distinte. (2 punti)

**Risposta:**

```
function [sol val] = radici(a,b,c,)  
    delta = b*b-4*a*c;  
    if (delta>0)  
        sol = 2;  
        val = [(-b-sqrt(delta))/2*a (-b+sqrt(delta))/2*a];  
    elseif (delta == 0)  
        val = -b/2*a;  
    else  
        sol = 0;  
        val = [];  
    end
```



**Esercizio 4 - Programmazione C - Liste (13 punti)**

Si presuma di dovere implementare un applicativo per la gestione del menu di un ristorante. Il menu è da considerarsi come la lista dei piatti che lo chef è in grado di offrire ai suoi clienti. Ogni piatto è caratterizzato da un nome, da una descrizione, e da un costo. Inoltre, un piatto può essere vegetariano oppure no.

A. Si definiscano innanzitutto le strutture dati necessarie alla gestione del menu. (1 punto).

**Risposta:**

```
typedef struct node{
    char nome[20];
    char descrizione[100];
    float costo;
    char vegetariano;
    struct node *next;
} node;
typedef node* ptrNode;
```

B. Si implementi una funzione che crea e restituisce una lista contenente solo i piatti del menu che sono vegetariani. (3 punti).

**Risposta:**

```
ptrNode listaVeg(ptrNode lista, ptrNode veggy){
    ptrNode temp;
    if (lista==NULL)
        return NULL;

    for (; lista!=NULL; lista = lista->next){
        if(lista->vegetariano == 'S'){
            temp = (ptrNode)malloc(sizeof(node));
            strcpy(temp->nome, lista->nome);
            strcpy(temp->descrizione, lista->descrizione);
            temp->costo = lista->costo;
            temp->vegetariano = lista->vegetariano;
            temp->next = veggy;
            veggy = temp;
        }
    }
    return veggy;
}
```

C. Si implementi una funzione **ricorsiva** che riceve una lista dei piatti scelti dai clienti e calcola il costo totale della consumazione. (4 punti).

**Risposta:**

```
float totale(ptrNode lista){
    if (lista == NULL)
```

```
    return 0;
    return lista->costo+totale(lista->next);
}
```

D. Scrivere una funzione **ricorsiva** che, presa in ingresso la lista dei piatti e i dati relativi ad un nuovo piatto, aggiunga il nuovo nodo alla lista in ordine crescente di prezzo (5 punti).

**Risposta:**

```
ptrNode aggiungiPiatto(ptrNode lista, char nome[], char desc[], float costo, char veg){
    ptrNode temp;
    if (lista==NULL){
        temp = (ptrNode)malloc(sizeof(node));
        strcpy(temp->nome, nome);
        strcpy(temp->descrizione, desc);
        temp->costo = costo;
        temp->vegetariano = veg;
        temp->next = lista;
        return temp;
    }
    if (costo>lista->costo){
        lista->next = aggiungiPiatto(lista->next, nome, desc, costo, veg);
        return lista;
    }
    else{
        temp = (ptrNode)malloc(sizeof(node));
        strcpy(temp->nome, nome);
        strcpy(temp->descrizione, desc);
        temp->costo = costo;
        temp->vegetariano = veg;
        temp->next = lista;
        return temp;
    }
}
```

**NOME e COGNOME:** \_\_\_\_\_

**NOME e COGNOME:** \_\_\_\_\_