Politecnico di Milano

Dipartimento di Elettronica, Informazione e Bioingegneria

Informatica A - Prof. A. Fuggetta - a.a 2023/2024 - 21 giugno 2024

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: **2h**

Valore indicativo degli esercizi, voti parziali e voto finale:

Esercizio 1	**5 punti**	_____
Esercizio 2	**5 punti**	_____
Esercizio 3	**7 punti**	_____
Esercizio 4	**15 punti**	_____
Totale(32)		_____

NOME e COGNOME: _____

Esercizio 1 - Algebra di Boole, Aritmetica Binaria, Codifica delle Informazioni (5 punti)

(a) Si costruisca la tabella di verità della seguente espressione booleana in tre variabili, badando alla precedenza tra gli operatori logici. Eventualmente si aggiungano parentesi.
Scrivere l'espressione semplificata. Non si accetteranno soluzioni senza il procedimento. (2 punti)

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

$AB + C\overline{(A + B\bar{C})}$ ↓ DM $\bar{A}\,\overline{B\bar{C}}$

$AB + C[\bar{A}(\bar{B} + C)]$

$AB + C[\bar{A}\bar{B} + \bar{A}C]$

$AB + C\bar{A}\bar{B} + C\bar{A} \rightarrow C(\bar{A}\bar{B} + \bar{A})$

Regola di assorbimento

$AB + C\bar{A}$

Truth Table

A	B	C	Output
0	0	0	F
0	0	1	T
0	1	0	F
0	1	1	T
1	0	0	F
1	0	1	F
1	1	0	T
1	1	1	T

(b) Si stabilisca il minimo numero di bit sufficiente a rappresentare in complemento a due i numeri A = -57dec e B = -71dec 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. Non si accetteranno soluzioni senza il procedimento. (2 punti)

A = -57 → 8 bit

→ 00111001 → 11000110 + 1 = 11000111

A = -71 → 8 bit

→ 01000111 → 10111000 + 1 = 10111001

A+B =

```
  11000111
+ 10111001
----------
 (1)10000000
```

No overflow
carriage return

A − B = A + (−B)

```
  11000111
+ 01000111
----------
 (1)00001110
```

Riporto No overflow

NOME e COGNOME: _____

(c) Si converta il numero 22,125 in virgola fissa e in virgola mobile con codifica IEEE 754 con precisione singola. Non si accetteranno soluzioni senza il procedimento. (1 punti)

32 bit

10110.001 $e = 4 + 127 = 131$ $s = 0$

$m = 0110\ 0010\ 0\ldots0$

s	e	
0	1000 0011	0110 0010 0...0
1	8	23

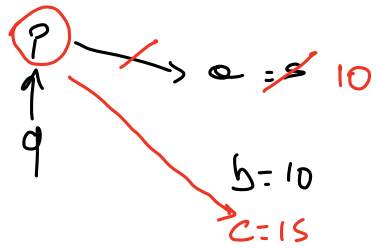
NOME e COGNOME: _____

Esercizio 2 - Teoria (5 punti)

Segnare con una crocetta le risposte che si ritengono corrette. Per ogni domanda, possono essere presenti da 1 a 4 soluzioni corrette.

(a) Si consideri il seguente segmento di codice in C:

```
int a = 5;
int *p = &a;
int **q = &p;
int b = 10;
*p = b;
int c = 15;
*q = &c;
```



Quale/i delle seguenti affermazioni sono corrette dopo l'esecuzione del codice sopra riportato?

- ☑ q punta a p
- ☑ il valore puntato da p è 15
- ☑ il valore di a è 10
- ☐ p punta ad a

(b) Dato il programma

```
#include <stdio.h>
#include <stdlib.h>

void modifyValue(int *ptr) {
    int *q;
    q = (int*)malloc(sizeof(int));
    *q = 75;
    ptr = q;
}

int main() {
    int number = 80;
    modifyValue(&number);
    printf("%d\n", number);
    return 0;
}
```

Cosa viene visualizzato a schermo?

- ☐ Il programma si blocca
- ☐ 75
- ☒ 80
- ☐ Il valore non è determinabile

(c) Date tre variabili booleane A, B e C, quale delle seguenti espressioni booleane è equivalente all'espressione (A AND B) OR (NOT C)?

- ☐ (A OR NOT C) AND (B OR NOT C)
- ☐ (NOT A AND NOT C) OR (NOT B AND NOT C)
- ☐ Nessuna delle risposte
- ☐ (A AND C) OR (B AND C)
- ☐ (NOT A OR C) AND (NOT B OR C)

$\bar{A}(\bar{B}+\bar{A})+$

NOME e COGNOME: _____

(d) Date le istruzioni

```
int *p, n=4;
p = (int*)malloc(n*sizeof(int));
```

Quando si manifesta un errore legato all'heap?

- ☐ Quando p assume un valore negativo
- ☐ Quando si passa a un sottoprogramma l'array p con modalità valore
- ☐ Quando l'allocazione porta ad uno stack overflow
- ☒ Quando p assume il valore NULL
- ☐ Quando il puntatore p viene allocato come variabile locale

(e) Una memoria indirizzabile su 5byte quanti valori numerici differenti può descrivere?

- ☐ 2^{45}
- ☐ 2^{43}
- ☒ 2^{40}
- ☐ Nessuna delle risposte

Start
$(A+\overline{C})(B+\overline{C})$

Apply: Distribution
$(B+\overline{C})A+(B+\overline{C})\overline{C}$

Apply: Distribution
$AB+A\overline{C}+(B+\overline{C})\overline{C}$

Apply: Distribution
$AB+A\overline{C}+\overline{C}B+\overline{C}\overline{C}$

Apply the Idempotent Law: $AA = A$
$AB+A\overline{C}+\overline{C}B+\overline{C}$

Apply the Absorption Law: $A+AB = A$
$AB+\overline{C}B+\overline{C}$

Apply the Absorption Law: $A+AB = A$
$AB+\overline{C}$

$A(B+\bar{C})+\bar{C}(B+\bar{C})$

$AB+A\bar{C}+\bar{C}B+\bar{C}$

$AB+\bar{C}B+\bar{C}$

$AB+\bar{C}$

NOME e COGNOME: _____

Esercizio 3 - Matlab (7 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 4 x 7 contenente lo stesso numero casuale tra 1 e 10 (1 riga - 1 punto)

2. Moltiplicare a tutte le celle della matrice, un numero casuale tra -5 e 5. Ogni cella va moltiplicata per un numero differente (1 riga - 1 punto)

3. Calcolare la somma totale della sottomatrice matrice 4x4 (1 riga - 1 punto)

4. Eliminare le colonne della matrice A la cui media delle componenti è minore di 0. (1 riga - 2 punti)

5. Scrivere una funzione Matlab che presa in ingresso la matrice, restituisca due vettori. Il primo vettore contiene la prima e l'ultima riga della matrice, il secondo vettore contiene la seconda e la terza riga. (2 punti)

1. Creare una matrice quadrata A di dimensione 4x7 contenente lo stesso numero casuale tra 1 e 10:

```
A = ones(4, 7) * randi([1, 10]);
```

2. Moltiplicare ogni cella della matrice per un numero casuale differente tra -5 e 5:

```
A = A .* randi([-5, 5], 4, 7);
```

3. Calcolare la somma totale della sottomatrice 4x4:

```
sum_submatrix = sum(sum(A(1:4, 1:4)));
```

4. Eliminare le colonne della matrice A con media delle componenti minore di 0:

```
A(:, mean(A) < 0) = [];
```

mean(A) < 0

TRUE
FALSE

5. Scrivere una funzione che restituisca due vettori:

```
function [vec1, vec2] = extract_rows(matrix)
    vec1 = [matrix(1, :), matrix(end, :)]; % Prima e ultima riga
    vec2 = [matrix(2, :), matrix(3, :)];  % Seconda e terza riga
end
```

1° 2° 3°

NOME e COGNOME: _____

Esercizio 4 - Programmazione C Liste (15 punti)

Si presuma di dover implementare un porogramma per la gestione della fase a gironi degli europei di calcio. Ogni squadra è caratterizzata da nome, gruppo, punti
Una seconda lista contiene invece tutte le partite da giocare, la lista contiene il nome delle due squadre, la data ed il risultato della partita.
NON E' RICHIESTO SCRIVERE IL MAIN.

1. Si definiscano le strutture dati necessarie allo sviluppo di questo programma. (1 punto)

2. Si implemente la funzione ricorsiva che prende in ingresso la lista delle partite, i nomi di due squadre ed il punteggio. La funzione aggiorna il nodo corrispondente della lista. La funzione restituisce inoltre il valore 0 se la partita tra le due squadre non è presente nella lista, 1 altrimenti.(3 punti)

3. Si implementi una funzione RICORSIVA che riceve in ingresso la lista totale di partite, ed il nome di una squadra, e crea una nuova lista con solo le partite di quella squadra, SENZA MODIFICARE LA LISTA DI PARTENZA (4 punti)

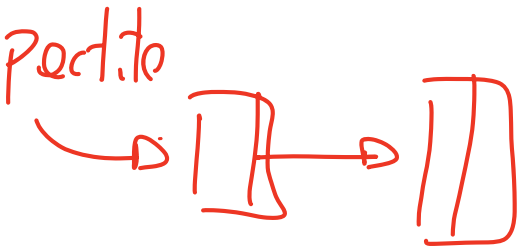
4. Scrivere una funzione che prese in ingresso le due liste, scorre tutta la lista di partite e aggiorna i punti della lista delle squadre. Vengono assegnati 3 punti per la vittoria, 1 punto per il pareggio e 0 per la sconfitta. (3 punti)

5. Scrivere una funzione che prendere in ingresso la lista delle squadre ed il gruppo. La funzione restituisce il nome delle due squadre con maggior punteggio. Si dia per scontato che non vi siano squadere a parimerito (4 punti)

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

// Struttura per rappresentare una squadra
typedef struct Squadra {
    char nome[50];
    char gruppo[5];
    int punti;
    struct Squadra *next;
} Squadra;

// Struttura per rappresentare una partita
typedef struct Partita {
    char squadra1[50];
    char squadra2[50];
    char data[11]; // Data in formato "YYYY-MM-DD"
    int risultato[2]; // [0] = gol squadra1, [1] = gol squadra2
    struct Partita *next;
} Partita;
```

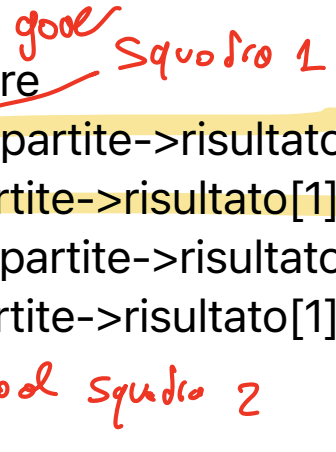
NOME e COGNOME: _____

```
int aggiornaRisultato(Partita *partite, const char *squadra1, const char
*squadra2, int gol1, int gol2) {
    if (partite == NULL) {
        return 0; // Partita non trovata
    }

    // Confronto dei nomi delle squadre
    if ((strcmp(partite->squadra1, squadra1) == 0 && strcmp(partite-
>squadra2, squadra2) == 0) ||
        (strcmp(partite->squadra1, squadra2) == 0 && strcmp(partite-
>squadra2, squadra1) == 0)) {
        partite->risultato[0] = gol1;
        partite->risultato[1] = gol2;
        return 1; // Partita trovata e aggiornata
    }

    // Passa al nodo successivo
    return aggiornaRisultato(partite->next, squadra1, squadra2, gol1, gol2);
}
```



```
Partita* filtraPartite(const Partita *partite, const char *squadra) {
    if (partite == NULL) {
        return NULL; // Lista vuota
    }

    // Verifica se la partita include la squadra
    if (strcmp(partite->squadra1, squadra) == 0 || strcmp(partite-
>squadra2, squadra) == 0) {
        // Crea un nuovo nodo e copia i dati
        Partita *nuovaPartita = (Partita *)malloc(sizeof(Partita));
        strcpy(nuovaPartita->squadra1, partite->squadra1);
        strcpy(nuovaPartita->squadra2, partite->squadra2);
        strcpy(nuovaPartita->data, partite->data);
        nuovaPartita->risultato[0] = partite->risultato[0];
        nuovaPartita->risultato[1] = partite->risultato[1];

        // Chiamata ricorsiva per il resto della lista
        nuovaPartita->next = filtraPartite(partite->next, squadra);
        return nuovaPartita;
    }
```

```
void aggiornaPunti(Squadra *squadre, const Partita *partite) {
    if (partite == NULL) {
        return; // Nessuna partita da processare
    }

    // Determina i punteggi delle squadre
    int punti1 = (partite->risultato[0] > partite->risultato[1]) ? 3 :
                 (partite->risultato[0] == partite->risultato[1]) ? 1 : 0;
    int punti2 = (partite->risultato[1] > partite->risultato[0]) ? 3 :
                 (partite->risultato[0] == partite->risultato[1]) ? 1 : 0;

    // Aggiorna i punti per squadra1
    Squadra *tmp = squadre;
    while (tmp != NULL) {
        if (strcmp(tmp->nome, partite->squadra1) == 0) {
            tmp->punti += punti1;
        } else if (strcmp(tmp->nome, partite->squadra2) == 0) {
            tmp->punti += punti2;
        }
        tmp = tmp->next;
    }

    // Passa alla prossima partita
    aggiornaPunti(squadre, partite->next);
}
```



```
void miglioriSquadre(const Squadra *squadre, const char *gruppo, char
*prima, char *seconda) {
    const Squadra *primaSquadra = NULL;
    const Squadra *secondaSquadra = NULL;

    while (squadre != NULL) {
        if (strcmp(squadre->gruppo, gruppo) == 0) {
            if (primaSquadra == NULL || squadre->punti > primaSquadra->punti)
{
                secondaSquadra = primaSquadra;
                primaSquadra = squadre;
            } else if (secondaSquadra == NULL || squadre->punti >
secondaSquadra->punti) {
                secondaSquadra = squadre;
            }
        }
        squadre = squadre->next;
    }

    if (primaSquadra != NULL) {
        strcpy(prima, primaSquadra->nome);
    }
    if (secondaSquadra != NULL) {
        strcpy(seconda, secondaSquadra->nome);
    }
}
```