Politecnico di Milano

Dipartimento di Elettronica, Informazione e Bioingegneria

Informatica A - a.a 2018/2019 - 30 Agosto 2019

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.**
- 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:

Esercizio 1	**3 punti**	_____
Esercizio 2	**3 punti**	_____
Esercizio 3	**4 punti**	_____
Esercizio 4	**6 punti**	_____
Esercizio 5	**12 punti**	_____

Totale(28) _____

Esercizio 1 - Algebra di Boole, Aritmetica Binaria, Codifica delle Informazioni (3 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. Non si accetteranno soluzioni senza il procedimento. (1 punto)

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

A	**B**	**C**	
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Risposta:
Semplificata: A + B

A	B	C	OUT
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

(b) Si stabilisca il minimo numero di bit sufficiente a rappresentare in complemento a due i numeri A = 23dec e B = 72, 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. (1 punto)

Risposta:
Servono 8 bit: da -128 a 127 23d = 10111b
cp2 = 010111b
72d = 01001000b
-72d cp2 = 10111000b

A + B

riporto	
A +	00010111b
B	01001000b
A+B	01011111b

né riporto perduto né overflow

A - B

riporto	11
A -	00010111b
B	10111000b
A-B	11001111b

Nè riporto perduto, né overflow

con 8 bit si possono rappresentare numeri da -128 a 127

(c) Si converta il numero 29.625 in virgola fissa e in virgola mobile con codifica IEEE 754, sapendo che 1/2 = 0.5, 1/4 = 0.25, 1/8 = 0.125, 1/16 = 0.0625, 1/32 = 0.03125, 1/64 = 0.015625, e 1/128 = 0.0078125. Non si accetteranno soluzioni senza il procedimento. (1 punto)

Risposta:
5bit + segno
29d = 11101b
0.5 + 0.125 = 0.101
37.625 = 011101.101 Virgola fissa

Virgola mobile
segno = 0
mantissa = 1.1101101 non normalizzata
exp = 4 + 127 = 131d = 10000011

IEEE754
segno (1bit) = 0
esponente (8bit) = 10000011
mantissa (23bit) = 110 1101 0000 0000 0000 0000 - nomralizzato eliminando il primo 1

Esercizio 2 - Domanda di teoria (3 punti)

1. Illustrare cosa sono le funzioni in C e qual'è la loro utilità.

2. Illustare la differena tra passaggio per valore e passaggio per puntatore. A cosa serve? Quando deve essere usato?

Esercizio 3 - Esercizio C (4 punti)

Scrivere la funzione **ricorsiva** *LetterChanges(char str[])* che preso in ingresso un'array di caratteri (lettere minuscole, numeri e segni), lo modifica secondo il seguente algoritmo:

- sostituisce ogni lettera nell'array con la lettera successiva; la z diventa a.
- trasforma in maiuscolo le vocali minuscole.

Esempi:
Input:"hello*3"
Output:"Ifmmp*3"

Input:"fun times!"
Output:"gvO Ujnft!"

suggerimento: ascii 'a' = 97; ascii 'A' = 65
È possibile aggiungere parametri alla funzione se necessari.

Si scriva inoltre il prototipo della funzione e la porzione di programma con l'invocazione della funzione (non è necessario scrivere tutto il main, scanf ecc.).

Risposta:

```c
#include <stdio.h>

void LetterChanges(char str[]) {
    if (*str == '\0')
        return;
    else{
        if ((*str>='a') && (*str<='z')){
            *str = *str+1;
            if (*str>'z')
                *str = *str-'z'+'a'-1;
            switch(*str){
                case 'a':
                case 'e':
                case 'i':
                case 'o':
                case 'u':
                    *str = *str -'a'+'A';
                    break;
            }
        }
        LetterChanges(str+1);
    }
}

int main(void) {

    // keep this function call here
    char stringa[20] = "fun times!z";

    LetterChanges(stringa);
    printf("%s\n",stringa);
    return 0;

}
```

Esercizio 4 - Matlab (6 punti)

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

(a) Scrivere una funzione che presi in ingresso tre numeri, restituisca
- 0 se i tre numeri sono sia divisibili per 2 che per 3
- 1 altrimenti (max 10 riga) (2 punto)

Risposta:

```
function out = agosto4_19(a,b,c)
   out = rem(a,2) + rem(b,2) + rem(c,2) + rem(a,3) + rem(b,3) + rem(c,3);
   if (out~=0)
       out = 1;
end
```

(b) Generare una matrice M con 6 righe e 4 colonne, contenente numeri casuali tra 0 e 100 (max 1 riga) (1 punto).

Risposta:

```
randi([0 100],[6 4])
```

(c) Eliminare la colonne con media inferiore a 10 (max 2 riga) (1 punto).

Risposta:

```
M(:,mean(M,1)<10)=[]
```

(d) Sostituire l'ultima riga con la radice quadrata dei valori della prima riga(max 1 riga) (1 punto).

Risposta:

```
M(end,:) = sqrt(M(1,:))
```

(e) Inserire una nuova colonna contenente la media delle righe (max 2 righe) (1 punto).

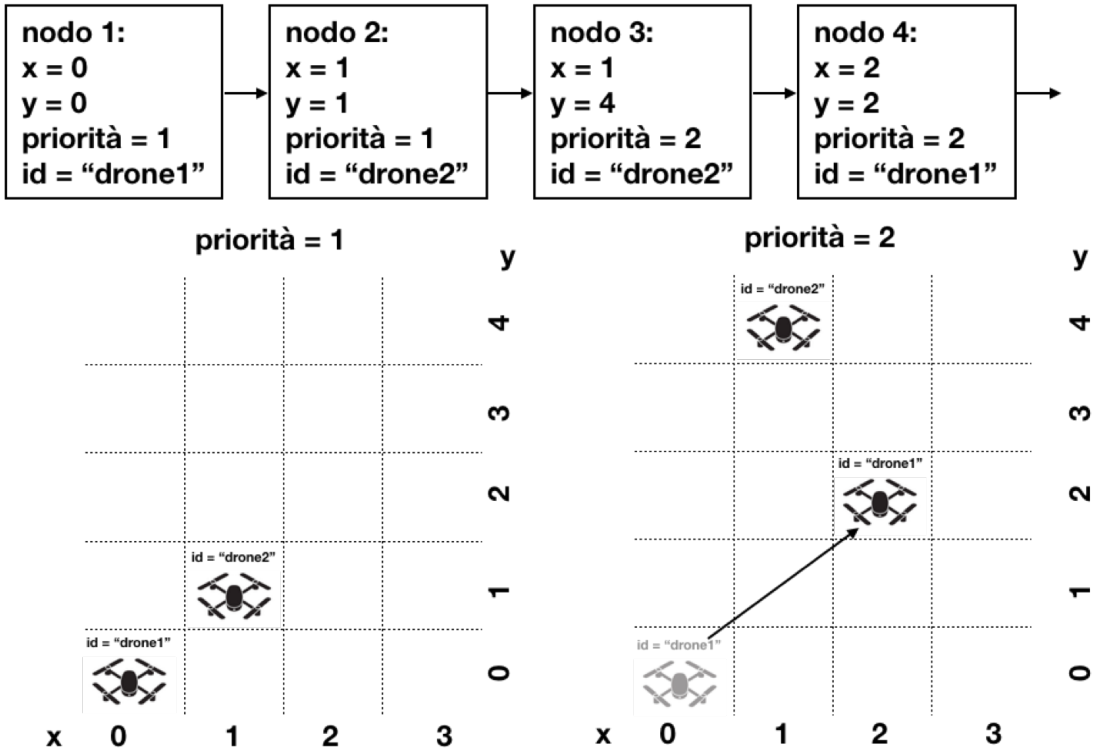
Risposta:

```
M = [M mean(M,2)]
```

Esercizio 5 - Programmazione C Liste (12 punti)

Si vuole creare un programma per la gestione di coreogragie con droni. Il programma è basato su *UNA SOLA LISTA* che contiene:

- le coordinate cartesiane X e Y che identificano i punti nel piano nei quali si trovano i droni nei differenti passaggi;
- il nome/identificativo del drone;
- la priorità di esecuzione (un numero incrementale che identifica la posizione temporale).



La distanza percorsa dal “drone1” è: $\sqrt{(x2-x1)^2+(y2-y1)^2} = \sqrt{(2-0)^2+(2-0)^2} = \sqrt{8} = 2{,}83$

Svolgere l’esercizio attenendosi a quanto richiesto.
NON È RICHIESTO SCRIVERE IL MAIN.

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

2. Scrivere la funzione RICORSIVA *”ptrNodo inserisciOrdinato(ptrNodo lista, int priorita, int x, int y, char id[])”* per inserire in modo ordinato (in ordine crescente rispetto alla priorità di esecuzione) un nuovo nodo. I valori da inserire sono letti nel main e passati come parametro. Non occorre effetture controlli su eventuali nodi duplicati. (3 punti)

3. Scrivere una funzione *”int checkCollisione(ptrNodo lista, int priority)”* che preso in ingresso la lista e la priorità di esecuzione, restituisca 1 se esiste più di un drone nella stessa posizione X,Y con la medesima priorità (c’è collisione), 0 altrimenti (4 punti)

4. Scrivere la funzione *”int distanza(ptrNodo lista, char id[])”* che preso in ingresso la lista e l’identificativo di un drone, restituisca lo spostamento totale del drone (la somma totale della distanza tra i punti appartenenti al drone). La lista è già ordinata. (4 punti)

Risposta:

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

typedef struct nodo{
    int x;
    int y;
    int priorita;
    char id[10];
    struct nodo *next;
} nodo;

typedef nodo *ptrNodo;

void stampa(ptrNodo lista);

ptrNodo sortInsert(ptrNodo lista, int priorita, int x, int y, char id[]);
int checkCollisione(ptrNodo lista, int priority);
int distanza(ptrNodo lista, char id[]);

int main(int argc, const char * argv[]) {
    ptrNodo pianoVolo = NULL;
    int r,x,y,priorita;
    char id[10];

    do{
        printf("MENU\n");
        printf("1) Inserisci nodo\n2) Stampa lista\n3) Check collisione\n4) Calcola spostamento\n
        5) ESCI\n\n>> ");
        scanf("%d",&r);
        switch (r) {
            case 1:
                printf("Inserisci x: ");
                scanf("%d",&x);
                printf("Inserisci y: ");
                scanf("%d",&y);
                printf("Inserisci priorita: ");
                scanf("%d",&priorita);
                printf("Inserisci id: ");
                scanf("%s",id);
                pianoVolo = sortInsert(pianoVolo, priorita, x, y, id);
                stampa(pianoVolo);
                break;
            case 2:
                stampa(pianoVolo);
                break;

            case 3:
                if (checkCollisione(pianoVolo,1)==1)
                    printf("C' collisione\n");
                else
                    printf("Nessuna collisione\n");
                break;

            case 4:
                printf("La distanza totale del drone 1 : %d\n\n",distanza(pianoVolo, "1"));
                break;
```

```
            default:
                break;
        }
    }while(r!=6);
    return 0;
}

void stampa(ptrNodo lista)
{
    if (lista == NULL)
        return;
    printf("X: %d\nY: %d\nPriority: %d\nID: %s\n\n",lista->x,lista->y,lista->priorita,lista->id);
    stampa(lista->next);
    return;
}

int checkCollisione(ptrNodo lista, int priority){
    ptrNodo savedLista = lista;
    ptrNodo l;

    int x,y;
    int first = 1;
    if (lista==NULL)
        return 0;
    for (l = savedLista; l!=NULL; l=l->next){
        first = 1;
        for (lista = l; lista!=NULL; lista=lista->next)
        {
            if (lista->priorita == priority){
                if (first==1){
                    x = lista->x;
                    y = lista->y;
                    first = 0;
                }
                else{
                    if ((x==lista->x) && (y==lista->y))
                        return 1;
                }
            }
        }
    }
    return 0;
}

ptrNodo sortInsert(ptrNodo lista, int priorita, int x, int y, char id[]) {
    ptrNodo n;
    if (lista != NULL && lista->priorita < priorita) {
        lista->next = sortInsert(lista->next, priorita, x, y, id);
        return lista;
    } else {
        n = (ptrNodo) malloc(sizeof(nodo));
        n->priorita = priorita;
        n->x = x;
        n->y = y;
        strcpy(n->id,id);
        if (lista != NULL) {
            n->next = lista;
        } else {
            n->next = NULL;
```

```
        }
        return n;
    }
}

int distanza(ptrNodo lista, char id[]){
    ptrNodo n=NULL;
    int somma = 0;

    if ((lista == NULL) || (lista->next==NULL))
        return 0;

    for (lista; lista!=NULL; lista = lista->next)
    {
        if (strcmp(lista->id,id)==0){
            if (n==NULL)
                n = lista;
            else
            {
                somma = somma + sqrt(pow(lista->x-n->x,2)+pow(lista->y-n->y,2));
                n = lista;
            }
        }


    }
    return somma;
}
```