

**PROVA SCRITTA DEL MODULO DI
ELEMENTI DI INFORMATICA
CORSO DI LAUREA IN INGEGNERIA BIOMEDICA
11 luglio 2019**

MOTIVARE IN MANIERA CHIARA LE SOLUZIONI PROPOSTE A CIASCUNO DEGLI ESERCIZI SVOLTI

ESERCIZIO 1 (4 punti)

- (1) (2 punti) Scrivere la tabella di verità degli operatori XOR e AND.
- (2) (2 punti) Siete stati incaricati di progettare un sensore per l'acquisizione di impronte digitali. Sapete di disporre di una risoluzione pari a 500 pixel per pollice e di 256 possibili livelli di grigio per pixel. Il pollice è un'unità di misura che indica approssimativamente la larghezza di un pollice. Quanto può essere la minima grandezza, in byte, dell'immagine lungo i due assi del piano per la cattura della totalità di una qualsiasi impronta, come nell'immagine a lato?
Nota: non è importante la ricerca di un valore "giusto", quanto il saper indicare il ragionamento che ha portato alla risposta.



ESERCIZIO 2 (4 punti)

Descrivere in modo chiaro e sintetico l'architettura di un moderno calcolatore elettronico.

ESERCIZIO 3 (25 punti)

(4 punti) In un'azienda dedicata alla differenziazione dei rifiuti sono stati installati dei separatori ottici per analizzare le tipologie di rifiuti ed inoltrarli ai trattamenti più opportuni. Attraverso i sensori ed un opportuno algoritmo, il sistema è in grado di categorizzare il rifiuto rilevato in:

- Classe 0 - CPL: bottiglie e flaconi in plastica;
- Classe 1 - Imballaggio: contenitori per liquidi non inclusi nei CPL;
- Classe 2 - Tracciante: shoppers e materiali d'imballaggio;
- Classe 3 - Frazione neutra: cassette per prodotti ortofrutticoli e alimentari;
- Classe 4 - Frazione estranea: qualsiasi manufatto non in plastica.

Il sistema scrive per ogni riga di un file "rifiuti.txt" la classe associata al rifiuto osservato in un dato istante, fra le cinque viste sopra, come numero intero compreso tra 0 e 4.

A voi si richiede di scrivere un programma C che stampi a video il numero di oggetti per ciascuna classe, espressa in chiaro come CPL, tracciante, etc., in ordine decrescente di quantità.

Per fare ciò, decidete di inserire tutti i valori letti da file in una lista concatenata di interi, che poi elaborerete per ottenere i dati richiesti. Inoltre implementate le seguenti funzioni:

- (4 punti) `leggiRifiuti()` – apre il file "rifiuti.txt" e restituisce una lista concatenata di valori interi corrispondenti ciascuno alle classi individuate dal separatore ottico.
- (6 punti) `elaboraRifiuti(lista)` – riceve una lista concatenata di valori interi e restituisce un vettore di interi chiamato `frequenze` indicizzato dai valori delle classi. Il valore `frequenze[i]` conterrà il numero di oggetti di classe `i`.
- (8 punti) `ordinaRifiuti(vettore)` – riceve un vettore di frequenze e restituisce un altro vettore contenente gli indici delle classi più frequenti, in ordine decrescente di quantità individuata dal vettore in ingresso.
- (3 punti) `stampaRifiuti(vettore, classi)` – stampa a video la coppia classe-quantità, in ordine decrescente, a partire dal vettore `classi` che contiene la sequenza degli indici delle classi per numero di rifiuti, ordinata, in ingresso. Il valore "classe" dev'esser la stringa corrispondente a ciascuna classe. Per esempio, se `vettore={12, 0, 3, 45, 1}`, `classi={3, 0, 2, 4, 1}`, la funzione dovrà stampare prima la coppia "Frazione neutra: 45" e a seguire le altre secondo l'ordine fornito dal vettore `classi`.

Nota: Le funzioni `CONS`, `HEAD`, `TAIL` e `ISEMPTY` per la gestione della lista di interi sono già implementate. Bonus di 3 punti per chi riuscirà ad implementare ulteriori funzioni per modularizzare il codice.

Soluzione dell'esercizio 1.

- 1) V. dispense del corso.
- 2) Poiché la larghezza di un pollice viene catturata da 500 pixel, ed il pollice presenta larghezza maggiore di quella delle altre dita, una possibile misura-base per la grandezza dell'immagine può essere 500x500 pixel, pari effettivamente alla grandezza delle immagini prodotte da alcuni sensori presenti nel mercato. Volendo utilizzare una potenza di 2, 512x512 pixel è una dimensione più che accettabile. Con 256 livelli di grigio, per rappresentare i quali occorre 1 byte, abbiamo un'immagine di 2^{18} byte, ovvero 256 KB. Più in generale, qualunque dimensione 500xd, con $d \geq 500$ pixel, va bene, in quanto l'altezza del pollice risulterà maggiore della sua larghezza, con la dimensione in B calcolata di conseguenza.

Soluzione dell'esercizio 2.

V. dispense del corso.

Soluzione dell'esercizio 3.

```
#define N 5 //numero classi di rifiuti
#include <stdio.h>
#include <stdlib.h>

typedef struct atomo
{
    int v;
    struct atomo* successivo;
} lista;

//Funzioni per la gestione della lista concatenata
lista* CONS(lista* l, int x)
{
    lista* t=(lista*)malloc(sizeof(lista));

    t->v=x;
    t->successivo=l;

    return t;
}

int HEAD(lista *l)
{
    return l->v;
}

lista* TAIL(lista *l)
{
    t=l->successivo;
    free(l);
    return t;
}

int ISEMPY(lista *l)
{
    return l==NULL;
}
```

```
//Funzioni richieste e funzioni bonus
```

```
lista* leggiRifiuti()
{
    lista* l=NULL;
    FILE *f;
    int x;

    f=fopen("rifiuti.txt","r");
    while(!feof(f))
    {
        fscanf(f,"%d",&x);
        l=CONS(l,x)
    }
    fclose(f);

    return l;
}
```

```
int* inizializza()
{
    int i;
    int *v=(int*)malloc(sizeof(int)*N);
    for(i=0; i<N; i++)
        v[i]=0;
    return v;
}
```

```
int* elaboraRifiuti(lista *l)
{
    int *frequenze;
    int i;

    frequenze=inizializza();

    while (!ISEMPTY(l))
    {
        i=HEAD(l);
        frequenze[i]++;
        l=TAIL(l);
    }

    return frequenze;
}
```

```
int *copia(int *v)
{
    int i;
    int *c=(int*)malloc(sizeof(int)*N);
    for(i=0; i<N; i++)
        c[i]=v[i];
    return c;
}
```

```

int *generaIndici()
{
    int i;
    int *c=(int*)malloc(sizeof(int)*N);
    for(i=0; i<N; i++)
        c[i]=i;
    return c;
}

int trovaMassimo(int *v, int start, int stop)
{
    int imax=start;
    int vmax=v[start];
    int i;
    for (i=start+1; i<stop; i++)
        if (v[i]>vmax)
        {
            vmax=v[i];
            imax=i;
        }
    return imax;
}

void scambia(int* v, int i, int j)
{
    int t=v[i];
    v[i]=v[j];
    v[j]=t;
}

int* ordinaRifiuti(int* frequenze) //v. dispense su selection-sort
{
    int* w=generaIndici();
    int* c=copia(frequenze); //necessario lavorare su copia di frequenze
    int i, imax;                //perché?

    for (i=0; i<N-1; i++)
    {
        imax=trovaMassimo(c,i,N);
        scambia(c,i,imax);
        scambia(w,i,imax);
    }

    return w;
}

char* estraiClasse(int i)
{
    switch(i)
    {
        case 0: return "CPL";
        case 1: return "Imballaggi";
        case 2: return "Traccianti";
        case 3: return "Frazione neutra";
    }
    return "Frazione estranea";
}

```

```

void stampaRifiuti(int* frequenze, int* ordinata)
{
    int i, indice;
    char *classe;

    for(i=0; i<N; i++)
    {
        indice=ordinata[i];
        classe=estraiClasse(indice);
        printf("%s: %d\n",classe,frequenze[indice]);
    }
}

```

```

int main()
{
    lista* l;
    int *f, *o;

    l=leggiRifiuti();
    f=elaboraRifiuti(l);
    o=ordinaRifiuti(f);
    stampaRifiuti(f,o);

    return 0;
}

```