

Fondamenti di Informatica e Laboratorio T-AB e Fondamenti di Informatica T1
Ingegneria Elettronica e Telecomunicazioni e
Ingegneria dell'Automazione
a.a. 2010/2011

Lab 08

Array

Esercizio 1

- Creare un programma che legga da input due sequenze di interi, di lunghezza non nota a priori (al più 10), e terminate da 0. A tal fine, si realizzi una funzione apposita che riceva come parametri un vettore vuoto (da riempire) e la sua dimensione fisica, e restituisca la dimensione logica.
- Per semplicità, si ipotizzi che ogni sequenza non contenga elementi ripetuti
- Si memorizzino in un terzo vettore tutti gli elementi che compaiono in entrambi gli array iniziali (intersezione), e lo si stampi a video

Esercizio 1 – Soluzione

```
#include <stdio.h>
#include <stdlib.h>
#define DIM 10

int leggi(int vet[], int dim) {
    int size = 0, num;
    do {
        printf("Inserisci un numero: ");
        scanf("%d", &num);
        if (num!=0 && size<dim) {
            vet[size] = num;
            size++;
        }
    } while (num!=0 && size<dim);
    return size;
}

int main(void) {
    int num, size1, size2, size3, i, j, trovato;
    int values1[DIM], values2[DIM], intersez[DIM];
    size1 = 0; size2 = 0; size3 = 0;
    size1 = leggi(values1, DIM);
    size2 = leggi(values2, DIM);
    ...
```

Esercizio 1 – Soluzione

```
...
for (i=0; i<size1; i++) {
    trovato = 0;
    for (j=0; j<size2 && !trovato; j++)
        if (values1[i] == values2[j])
            trovato = 1;
    if (trovato) {
        intersez[size3] = values1[i];
        size3++;
    }
}
for (i=0; i<size3; i++)
    printf("Valore comune: %d\n", intersez[i]);

system("PAUSE");
return (0);
}
```

Esercizio 2

- Creare un programma che legga da input una sequenza di interi, di lunghezza non nota a priori (al più 10), e terminata da 0. A tal scopo, si realizzi una funzione che riceva come parametri di ingresso un vettore e la sua dimensione fisica, e restituisca la dimensione logica del vettore. Tale funzione si deve fare carico della fase di lettura e riempimento dell'array.
- La sequenza può contenere elementi ripetuti (anche più volte).
- Si realizzi una funzione che, ricevuti in ingresso il primo vettore con la sua dimensione logica, ed un secondo vettore con la sua dimensione fisica, memorizzi nel secondo vettore tutti gli elementi del primo, ma senza ripetizioni. La funzione restituisca la dimensione logica del secondo vettore.
- Si realizzi un main che invoca le funzioni, e che stampi a video l'elenco degli elementi non ripetuti

Esercizio 2 – Soluzione

```
#include <stdio.h>
#include <stdlib.h>
#define DIM 10

int leggi(int vet[], int dim) {
    int size = 0, num;
    do {
        printf("Inserisci un numero: ");
        scanf("%d", &num);
        if (num!=0 && size<dim) {
            vet[size] = num;
            size++;
        }
    } while (num!=0 && size<dim);
    return size;
}

int main(void) {
    int num, size_v, size_s, i, j, trovato;
    int values[DIM], single[DIM];

    size_s = 0;
    size_v = leggi(values, DIM);
    ...
```

Esercizio 2 – Soluzione

```
...
for (i=0; i<size_v; i++) {
    trovato = 0;
    for (j=0; j<size_s && !trovato; j++) {
        if (values[i] == single[j])
            trovato = 1;
    }
    if (!trovato) {
        single[size_s] = values[i];
        size_s++;
    }
}

for (i=0; i<size_s; i++)
    printf("%d ", single[i]);

system("PAUSE");
return 0;
}
```

Esercizio 3

- Creare un programma che legga da input un numero non noto a priori di interi (al più 10) terminati da 0. Tale sequenza può eventualmente contenere numeri ripetuti. A tal fine, si realizzi una funzione apposita che riceva in ingresso un vettore (vuoto) e la sua dimensione fisica, e restituisca la dimensione logica.
- Si chieda poi l'inserimento di un valore k da cercare nell'array.
- Si realizzi una ulteriore funzione che stampi a video tutti gli indici relativi alle posizioni in cui il numero k è presente nell'array. La funzione riceverà quindi come parametri l'array, la sua dimensione logica e la soglia k .

Esercizio 3 – Soluzione

```
#include <stdio.h>
#include <stdlib.h>
#define DIM 10

int leggi(int vet[], int dim) {
    int size = 0, num;
    do {
        printf("Inserisci un numero: ");
        scanf("%d", &num);
        if (num!=0 && size<dim) {
            vet[size] = num;
            size++;
        }
    } while (num!=0 && size<dim);
    return size;
}

int main(void) {
    int num, size, i, k;
    int values[DIM];

    size = leggi(values, DIM);
    ...
```

Esercizio 3 – Soluzione

...

```
printf("Inserisci valore da cercare: ");  
scanf("%d", &k);  
  
for (i=0; i<size; i++)  
    if (values[i] == k)  
        printf("Trovato il valore %d all'indice %d\n ", k, i);  
  
system("PAUSE");  
return (0);  
}
```

Esercizio 4

- Creare un programma che legga da input un numero non noto a priori di interi (al più 10) terminati da 0. Tale sequenza può eventualmente contenere numeri ripetuti. A tal fine si definisca una opportuna funzione.
- Si realizzi una funzione che provveda a copiare in un secondo array tutti numeri che compaiono una sola volta nel primo array. La funzione riceverà come parametri il primo array, la sua dimensione logica, il secondo array con la sua dimensione fisica, e restituirà la dimensione logica del secondo array
- Si stampi a video il contenuto del secondo array

Esercizio 4 – Soluzione

```
#include <stdio.h>
#include <stdlib.h>
#define DIM 10

int leggi(int vet[], int dim) {
    int size = 0, num;
    do {
        printf("Inserisci un numero: ");
        scanf("%d", &num);
        if (num!=0 && size<dim) {
            vet[size] = num;
            size++;
        }
    } while (num!=0 && size<dim);
    return size;
}

int main(void) {
    int repeat[DIM];
    int single[DIM];
    int num, size, size_single, i, j, trovato;

    size = leggi(repeat, DIM);
```

Esercizio 4 – Soluzione

```
...
size_single = 0;
for (i=0; i<size; i++) {
    trovato = 0;
    for (j=0; j<size && !trovato; j++) {
        if (repeat[i]==repeat[j] && i!=j)
            trovato = 1;
    }
    if (!trovato) {
        single[size_single] = repeat[i];
        size_single++;
    }
}
for (i=0; i<size_single; i++)
    printf("%d ", single[i]);

system("PAUSE");
return (0);
}
```