**Dipartimento di Informatica e Sistemistica**
**Antonio Ruberti**

**"Sapienza" Università di Roma**

# Esercitazione 7

## *Corso di Fondamenti di Informatica*

### *Laurea in Ingegneria Informatica*

***(Canale di Ingegneria delle Reti e dei Sistemi Informatici - Polo di Rieti)***

Anno Accademico 2007/2008

**Tutor: Ing. Diego Rughetti**

# Esercizio 1

Scrivere un programma c che prende in ingresso da tastiera una serie di 20 numeri interi. I numeri immessi devono essere salvati all'interno di un array.

Una volta terminata l'immissione. Stampare a video l'array e successivamente ordinare una copia dell'array in ordine crescente. Scrivere una funzione C che permetta di effettuare l'ordinamento secondo l'algoritmo selection sort. Infine stampare a video l'array ordinato su una sola riga.

Per effettuare le operazioni utilizzare l'aritmetica dei puntatori e le funzioni di allocazione dinamica della memoria (malloc()), non la notazione con puntatore ed indice (array[i]).

# Soluzione (1)

```
int main(){
        int *myArray = malloc(20*sizeof(int));
        int *copyArray = malloc(20*sizeof(int));
        int passi, i;
        for (i = 0; i<20; i++){
                printf("inserisci numero: ");
                scanf("%d", (myArray + i));
        }
        for(i = 0; i<20; i++){
                *(copyArray+i) = *(myArray + i);
        }
        passi = sort(copyArray, 20);
        for(i = 0; i<20; i++){
                printf("%d, ", *(copyArray++));
        }
        printf("\n");
        printf("Passi s.s. = %d", passi);
        scanf("%d", i);
}
```

# Soluzione (2)

```
void scambia(int *vet,int i,int j){
        int tmp;
        tmp = *(vet+i);
        *(vet+i) = *(vet+j);
        *(vet+j) = tmp;
}
int  selectMax(int *vet, int dim){
        int i,max,imax=0;
        int passi = 0;
        max = *vet;
        for(i=0;i<dim; i++){
                if(*(vet+i) > max) {
                        max = *(vet+i); imax = i;
                }
                passi++;
        }
        scambia(vet, dim-1, imax);
        return passi;
}
int sort(int *vet, int dim){
        int i, passi = 0;
        for(i=0;i<dim; i++){
                passi += selectMax(vet,dim-i);
        }
        return passi;
}
```

# Esercizio 2

Scrivere un programma C che genera o prende in ingresso due array ordinati e li stampa a video. Successivamente il programma deve fondere i due array in un solo array ordinato e stampare l'array ottenuto a video. Implementare infine una funzione di check che verifica se un array è ordinato.

Nelle funzioni utilizzare il passaggio di parametri per indirizzo.

# Soluzione (1)

```
#include <stdio.h>
#include <stdlib.h>             // per la random

#define N 10000

void merge(int A[], int na, int B[], int nb, int C[]);
void generaArrayOrdinato(int A[], int n);
void stampaarray(int A[], int n);
int checkorder(int A[], int n);
void scambia(int* a, int* b);
/*
Main di prova: viene letta la dimensione (corrente) dell'array e generati i valori in modo casuale.
Quindi si ordina l'array e si controlla che risulti ordinato con la funzione checkorder
*/
int main(void){
                int na, nb;
                int *A = malloc(N*sizeof(int));
                int *B = malloc(N*sizeof(int));
                int *C = malloc(N*sizeof(int));
                printf("Numero di elementi dell'array A:"); scanf("%d",&na);
                generaArrayOrdinato(A,na);
                printf("Numero di elementi dell'array B:"); scanf("%d",&nb);
                generaArrayOrdinato(B,nb);
                printf("Array A:\n");stampaarray(A,na);
                printf("Array B:\n");stampaarray(B,nb);
                merge(A,na,B,nb,C);
                printf("Array risultato C\n");stampaarray(C,na+nb);
                printf("Checkorder: %d\n",checkorder(C,na+nb));
                scanf("%d", &na);
}
```

# Soluzione (2)

```
/*
Procedura di fusione di due array ordinati. Viene prima eseguito il ciclo di bilanciamento. Il ciclo termina non appena
uno dei due array viene esaurito. Di seguito viene ricopiato in C il resto dell'array non esaurito.
*/
void merge(int *A, int na, int *B, int nb, int *C){
                int i,j,k;
                i=j=k=0;
                while (i<na && j<nb){
                        if (*(A+i)<*(B+j)){
                                *(C+k)=*(A+i);
                                k++;
                                i++;
        }
                        else{
            *(C+k)=*(B+j);
            k++;
            j++;
        }
    }
    while (i<na){
         *(C+k)=*(A+i);
         k++;
         i++;
    }
                while (j<nb){
         *(C+k) = *(B+j);
         k++;
         j++;
    }
}
```

# Soluzione (3)

```
/*Funzioni di stampa, generazione di array*/
void generaArrayOrdinato(int *A, int n){
        int i;
        *A =(int)(rand()%100);
   for (i = 1; i<n; i++){
                A++;
      *A=*(A-1)+(int)(rand()%100);
   }
}
void stampaarray(int *A, int n){
        int i;
        for (i = 0; i<n; i++)
                printf("%d ", *(A+i));
        printf("\n");
}
/*Funzione per verificare l'ordinamento di un array*/
int checkorder(int *A, int n){
        int i;
        for (i=0;i<n-1;i++)
                if (*(A+i)> *(A+(i+1)))
                        return 0;
        return 1;
}
```

# Esercizio 3

Scrivere una funzione c che prende in ingresso tre array A1, A2, A3. La funzione, per ogni i calcola la somma tra A1[i] e A2[i] e salva il risultato in A3[i].
Scrivere un main che inizializzi i due vettori A1 e A2, invochi la funzione di somma ed infine stampi il contenuto di A3[].
Solo nella funzione utilizzare il passaggio di parametri per indirizzo ed utilizzare l'aritmetica dei puntatori.

# Soluzione (1)

```
void somma(int *A, int *na, int *B, int *nb, int *C);

int main(){
    int A[5]; int B[6]; int C[10];
    int i; int k = 0; int na; int nb;
    int *pointA, *pointB;
    pointA = &na;
    pointB = &nb;
    *pointA = 5; // Domanda: le variabili na ed nb a seguito di queste due assegnazioni

    *pointB = 6; //vengono modificate? Se si, che valori assumono? Perché?
    for (i = 4; i>=0; i--){
        A[k++] = i;
    }
    for (i = 0; i<10; i++){
        C[i] = 0;
    }
    for (i = 0; i<6; i++){
        B[i] = i;
    }
    somma(A, pointA, B, pointB, C);
    i=0;
    for (i = 0; i<10; i++){
        printf("%d\n", C[i]);
    }
}
```

# Soluzione (2)

```
void somma(int *A, int *na, int *B, int *nb, int *C){
    int h;
    int i = 0;
    int j = 0;
    int k = 0;
    int a, b;
    a = *na;
    b = *nb;
    while (i<a && j<b){
        *C = *A+*B;
        C++; A++; B++;
        i++; j++;

    }
    while (i<a){
                        *C=*A;
                        C++;A++;
                        i++;
    }
    while (j<b){
                        *C = *B;
                        C++; B++;
                        j++;
    }
}
```

# Esercizio 4

Scrivere una funzione che prende in ingresso una matrice N×N di reali e la modifica azzerandone tutti gli elementi sulle diagonali. Testare la funzione scrivendo un programma che legge da tastiera una matrice M×M di reali, applica la funzione e stampa la matrice modificata.
Utilizzare i puntatori e la loro aritmetica.

# Soluzione (1)

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

#define M 4
void diagNulle(int *mat);
void leggiMat(int *mat);
void stampaMat(int *mat);

int main()
{
	int *mat = malloc(M*M*sizeof(int));
	int i,j;
	printf("digita matrice %d per %d: \n",M,M);
	leggiMat(mat);
	printf("azzero le diagonali ...\n");
	diagNulle(mat);
	printf("risultato:\n");
	stampaMat(mat);
	system("PAUSE");
	return 0;
}
```

# Soluzione (2)

```
void diagNulle(int *mat){
  int i,j;
  for (i=0; i < M; i++)
     for (j=0; j<M ; j++)
        if (i==j || i == M -j -1)
          *(mat + (M*i)+j) = 0;
}
void leggiMat(int *mat) {
   int i,j;
   for (i=0 ; i<M ; i++)
         for (j=0 ; j < M ; j++)
           scanf("%d", (mat + (M*i) + j));
}
void stampaMat(int *mat) {
    int i,j;

    for (i=0 ; i<M ; i++){
         for (j=0 ; j < M ; j++)
           printf("%4d",*(mat +(M*i)+j));
         printf("\n");
    }
  }
```