Dipartimento di Informatica e Sistemistica
Antonio Ruberti

"Sapienza" Università di Roma

Esercitazione 6

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 l'array in ordine crescente. Scrivere delle funzioni c che permettano di effettuare l'ordinamento secondo gli algoritmi selection sort e bubble sort. Infine stampare a video l'array ordinato su una sola riga e i passi necessari per effettuare l'ordinamento con i due differenti algoritmi.

Soluzione (1)

```
int main(){
        int myArray[20];
        int copyArray[20];
        int passi;
        …… immissione da tastiera e salvataggio in un array
        //ordinamento e stampa dell'array
        for(i = 0; i<20; i++){
                copyArray[i] = myArray[i];
        }
        passi = sort(myArray, 20);
        for(i = 0; i<20; i++){
                printf("%d, ", myArray[i]);
        }
        printf("\n");
        printf("Passi s.s. = %d", passi);
        passi = bubble(copyArray, 20);
        printf("Passi b.s. = %d", passi);
}
```

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[0];
        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;
}
```

Soluzione (3)

```
int  bubble(int *vet, int n){
        int i, j, passi = 0;
        for ( i=0;i< n-1; i++)
                for (j=n-1;j>i; j--){
                        passi++;
                        if ( vet[j-1] > vet[j] )
                                scambia(vet,j-1,j);
                }
        return passi;
}

int bubble(int *vet, int n){
        int i,j,ordinato=1;
        int passi = 0;
        for ( i=0;i< n-1 && ordinato; i++){
                ordinato=1;
                for (j=n-1;j>i; j--)
                        passi++;
                        if ( vet[j-1] > vet[j] ){
                                scambia(vet,j-1,j);
                                ordinato = 0;
                        }
        }
        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 A[N],na,B[N],nb,C[N];
        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));
}
```

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++];
                else
                        C[k++]=B[j++];
        while (i<na)
                C[k++]=A[i++];
        while (j<nb)
                C[k++]=B[j++];
}
```

Soluzione (3)

```
/*Funzioni di stampa, generazione di array*/
void generaArrayOrdinato(int *A, int n){
        int i;
        A[0]=(int)(random()%100);
        for (i = 1; i<n; i++)
                A[i]=A[i-1]+(int)(random()%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.
Nella funzione utilizzare il passaggio di parametri per indirizzo.

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 i = 0;
    int j = 0;
    int k = 0;
    int a, b;
    a = *na;
    b = *nb;
    while (i<a && j<b)
        C[k++] = A[i++] + B[j++];
    while (i<a)
                    C[k++]=A[i++];
    while (j<b)
                    C[k++]=B[j++];
}
```

Esercizio 4

Scrivere un semplice programma c che definisca una serie di variabili intere ed una serie di puntatori ad interi. Dopo di che assegni dei valori opportuni alle variabili ed ai puntatori e per ognuno di essi stampi a monitor il contenuto, l'indirizzo e il valore puntato (in caso di puntatore).

Soluzione

```
int main(){
    int a, b, c, d;
    int *pointA, *pointB, *pointC;
    a = 1;
    b = 2;
    c = 3;
    printf("a = %d\n", a);
    printf("indirizzo di a = %p\n", &a);
    printf("b = %d\n", b);
    printf("indirizzo di b = %p\n", &b);
    printf("c = %d\n", c);
    printf("indirizzo di c = %p\n", &c);
    pointA = &a;
    pointB = &b;
    pointC = &c;
    printf("contenuto di pointA = %p\n", pointA);
    printf("indirizzo di pointA = %p\n", &pointA);
    printf("valore puntato da pointA = %d\n", *pointA);
    printf("contenuto di pointB = %p\n", pointB);
    printf("indirizzo di pointB = %p\n", &pointB);
    printf("valore puntato da pointB = %d\n", *pointB);
    printf("contenuto di pointC = %p\n", pointC);
    printf("indirizzo di pointC = %p\n", &pointC);
    printf("valore puntato da pointC = %d\n", *pointC);
    scanf("%d", &d);
}
```