Esercitazioni di Fondamenti di Informatica - Lez. 9
26/11/2020

Esercizi ripasso

1. **Si implementi una funzione ricorsiva che stampi la stringa ricevuta come parametro, in ordine inverso.**

   void stampa_inversa(**char** *s)

```c
#include <stdio.h>
#define MAX_CHAR 256

void stampa_inversa(char *s){
    if (*s == 0)
        return;

    stampa_inversa(s+1);
    printf("%c", *s);

}

int main() {

    char source[MAX_CHAR] = "HELLO";

    printf("L'inverso di %s è “,source);

    stampa_inversa(source);
    printf("\n");

    return 0;
}
```

2. Si implementi una funzione ricorsiva che calcoli la lunghezza di una stringa.

void strlen(**char** *s)

```c
#include <stdio.h>
#define LEN 100

// versione std:
int strlen_std(char *s){
    char *pc;
    int l = 0;

   for(pc = s; *pc != '\0'; pc++){
        l++;
    }
    return l;
}

// versione ricorsiva: la lughezza di una stringa s e' 1 + la lunghezza del RESTO
della stringa.
int strlen_ric(char *s){
    if (*s == 0)
        return 0;
    return 1 + strlen_ric(s+1);
    // oppure:
    //return 1 + strlen_ric(s++);
}

int main(void) {
    char testStr[LEN];
    int l;

    printf("Inserisci una stringa: ");
    scanf("%s",testStr);

    // Uso la funzione standard
    l = strlen_std(testStr);
    printf("Lunghezza funzione standard: %d\n", l);

    // Uso la funzione ricorsiva
    l = strlen_ric(testStr);
    printf("Lunghezza funzione ricorsiva: %d\n", l);

    return 0;
}
```

3. Si implementi una funzione ricorsiva che calcoli la somma degli elementi presenti in un array di interi

La funzione riceve il puntatore al primo elemento e il numero di elementi dell'array

void somma_ric(int *pVal, int n)

```c
#include <stdio.h>

int somma_ric(int * pValori, int n){
    if (n == 0)
        return 0;

    int primoValore = *pValori;

    return primoValore + somma_ric(pValori+1, n-1);
}

int main(int argc, const char * argv[]) {

    int valori[] = {10,20,30};
    int sum;

    sum = somma_ric(valori, 3);

    printf("La somma dei valori dell'array e': %d\n", sum);

    return 0;
}
```

4. Prova del 08/09/2020 esercizio 3

Dato un array di numeri interi a, restituisce 1 se la differenza tra i numeri opposti (rispetto al centro) è costante; restituisce 0 altrimenti. Per opposti, si intende il primo e l'ultimo, il secondo ed il penultimo. Se l'array avesse lunghezza dispari, l'elemento centrale non avrebbe opposto e la funzione dovrebbe ritornare 1.

```c
// logica:
// a) appena NON soddisfa, gli opposti, torno zero
// b) devo sapere lunghezza array -> parametro
// quando ricorro, avanzo puntatore e scalo lunhghezza di DUE
// devo passare ANCHE la diff del primo ed ultimo al "primo giro"
// suppongo primo meno ultimo
```

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

#define MAX_VAL 7


int diff_costante(int * arr, int len, int diff_da_verificare){
   int primo = *arr;
   if (len<=1)
       return 1;
   int ultimo = *(arr+len-1);

  // i valore assoluto per gestire il caso con differenza negativa
  // int res = abs(primo -ultimo));

   if ( primo  - ultimo != diff_da_verificare)
       return 0;

   int result = diff_costante(arr+1, len-2, diff_da_verificare);
   return result;
}

int main() {

   int valori[MAX_VAL] = {15,12,100,5, 96, 8, 11};

   int len = MAX_VAL;

  // inserire eventuale gestione valore assoluto

  int initial_diff = valori[0] - valori[len-1];

   int costanti = 0;
    costanti = diff_costante(valori+1, len-2, initial_diff);

   if (costanti==1)
       printf("la differenza è costante %d\n",initial_diff);
    else
        printf("la differenza non è costante %d\n",initial_diff);
   return 0;
}
```

5. Costruite un programma per gestire una lista di film. Per immagazzinare i film dovete usare la seguente struttura:

typedef struct _movie
{ char title[200];
char type[200];
int year; }
movie;

Per immagazzinare i film che vengono inseriti dall'utente dovete usare un array.
il dato contenuto in ogni casella dell'array non sarà un semplice char o un int, ma una struttura di tipo movie.
Nell'array i nuovi film devono essere inseriti in modo ordinato a seconda dell'anno in cui è stato girato il film (prima i film più vecchi poi quelli più nuovi).

```
movie film[100];
int numFilm;
```

// numFilm servirà a sapere quante caselle dell'array sono effettivamente usate

implementate le funzioni che devono essere richiamate opportunamente in un apposito main di test:

int add(movie f[],int numFilm)
//chiede i dati per un nuovo film, lo aggiunge nella posizione corretta e restituisce il numero di film aggiornato.
void print(movie f[],int numFilm)
//stampa la lista dei film.
*movie search(movie f[],int numFilm, char * title)*
//cerca un film nella lista in base al titolo e restituisce l'elemento dell'array che contiene tale film.
*int delete(movie f[],int numFilm,char * title)*
//rimuove un film senza lasciare buchi e restituisce il numero di film aggiornato.

```c
/* film */

#include <stdio.h>
#include <string.h>


typedef struct new_movie {
    char title[200];
    char type[200];
    int year;
} movie;
```

```c
#define MAX_FILMS 100

// prototipi:

int add(movie f[],int numFilm); //chiede i dati per un nuovo
film, lo aggiunge nella posizione corretta e restituisce il
numero di film aggiornato.

void print(movie f[],int numFilm); //stampa la lista dei film.

movie search(movie f[],int numFilm, char * title); //cerca un
film nella lista in base al titolo e restituisce l'elemento
dell'array che contiene tale film.

int delete(movie f[],int numFilm,char * title); //rimuove un
film senza lasciare buchi e restituisce il numero di film
aggiornato.

int main() {

    int numFilm=0;

    movie film[MAX_FILMS];
    int i;

    for (i=0; i<6; i++) {
        numFilm = add(film, numFilm);
    }
    print(film, numFilm);

    printf("Done!\n");
    printf("cerco Bonny e Clyde \n");
    movie m = search(film , numFilm, "Bonnie and Clyde");
    printf("trovato :%s lo cancello \n",m.type);

    numFilm=delete(film , numFilm, "Bonnie and Clyde");
    printf("ora i film sono ...%d \n",numFilm);
    print(film, numFilm);
    return 0;
}

// fake f. add per testare velocemente:


movie fakeReadMovie(){
    static int presi = 0;
    movie films[MAX_FILMS] = {
        {"The Seven Samurai", "guerra", 1954},
```

```
        { "Bonnie and Clyde", "gangster", 1967},
        { "Reservoir Dogs", "dramma", 1992},
        { "Airplane!", "comico" , 1980},
        { "Pan's Labyrinth!", "mith" , 2006},
        {"Doctor Zhivago", "drama", 1965},
    };
    movie m = films[presi];
    presi++;
    return m;
}

// f. di appoggio: se array per definizione ordinato, cerco
con FOR dove inserirlo:
int posizInserimento(int anno, movie f[], int numFilm){
    int i;
    for (i=0; i<numFilm; i++) {
        if  (anno <f[i].year){
            return i;
        }
    }
    return numFilm;

}
// f. di appoggio:
// un array e' cmq un array di puntatori.. non mi serve dire
da dove... gli passo quello corretto..
// ma attenzione al  numero elemewnti!
// ma devo partire dal FONDO
// per ipotesi (vedi la add) gia ho testato che HO SPAZIO in
fondo.
void scalaInSu(movie f[], int numFilm){
    int i;

    //printf("\n========\n");
    //print(f, numFilm);

    for (i=0; i<numFilm; i++) {
        f[numFilm-i] = f[numFilm-i-1];
    }
    //print(f, numFilm+1);
}

int add(movie f[],int numFilm) //chiede i dati per un nuovo
film, lo aggiunge nella posizione corretta e restituisce il
numero di film aggiornato.
{
    if (numFilm == MAX_FILMS-1){
        printf("al max! impossibile aggiungere\n");
        return numFilm;
```

```
    }

    // chiamata reale:
    //movie m = readMovie();

    // fake x testare:
    movie m = fakeReadMovie();

    int nuovaPos = posizInserimento(m.year, f, numFilm);

    if (nuovaPos == numFilm){
        f[numFilm] = m;
    }else{
        scalaInSu(f+nuovaPos, numFilm-nuovaPos);
        f[nuovaPos] = m;
    }

    // x testare:
    print(f, numFilm+1);

    return numFilm+1;
}

// f. di "apppoggio": stampa singolo film:
void printFilm(movie f)
{
    printf("%-20s %-20s %5d\n", f.title, f.type, f.year );
}

void print(movie f[],int numFilm) //stampa la lista dei film.
{
    printf("\n-------\n");
    int i;
    for (i=0; i<numFilm; i++) {
        printFilm(f[i]);

    }
}
movie search(movie f[],int numFilm, char *title)
//cerca un film nella lista in base al titolo e restituisce
l'elemento dell'array che contiene tale film.
{
    movie m = {"","",0};

    int i;
    for (i=0; i<numFilm; i++) {

        if (strcmp(f[i].title,title)==0)
            m =f[i];
```

```
        }
    return m;
}

//rimuove un film senza lasciare buchi e restituisce il numero
di film aggiornato.
int delete(movie f[],int numFilm,char * title)
{
    int i=0,j;

    for (i=0; i<numFilm; i++) {
        if (strcmp(f[i].title,title)==0)
            break;
    }
    printf("trovato %d %s\n", i,f[i].type);
    // quando esco dal ciclo i si è incrementato di 1

    for (j=i; j<numFilm; j++) {
        f[j] = f[j+1];
    }
    return numFilm-1;

}
```