Fondamenti di Informatica L-A (A.A. 2004/2005 - Ingegneria Informatica)
Prof.ssa Mello & Prof. Bellavista – Seconda Prova Intermedia del 02/12/2004 - durata 2h
COMPITO A

ESERCIZIO 1 (13 punti)

Una società di telefonia cellulare gestisce un programma di premiazione per "utenti fedeli". In particolare, per ogni cliente viene salvato su un file binario "**`punti.dat`**" il nome del cliente (al massimo 31 caratteri), e un numero intero che rappresenta i punti accumulati. Tali informazioni sono organizzate come una struttura **`user`**, opportunamente definita dal candidato.
Si scriva una funzione:

```
int readPoints (char usersFile[], user results[], int maxDim, int minPoints)
```

che, ricevuto in ingresso il nome di un file **`usersFile`**, un array **`results`** di strutture **`user`**, la dimensione massima dell'array **`maxDim`**, e un limite inferiore di punti **`minPoints`**, copi nell'array **`results`** i dati dei clienti che hanno almeno i punti specificati da **`minPoints`**. La funzione deve restituire come risultato il numero di utenti con almeno **`minPoints`**; si noti che tale risultato rappresenta anche la dimensione logica dell'array **`results`**. Qualora il file non sia accessibile, la funzione deve restituire il valore -1 **(6 punti)**.

Si scriva poi un programma main() che chieda all'utente il numero di clienti salvati sul file (tale numero sarà noto solo a tempo di esecuzione), e allochi dinamicamente un vettore **`V`** di **`user`** sufficientemente grande per poter contenere, nel caso peggiore, i dati di tutti gli utenti salvati in **`usersFile`**. Il programma dovrà poi chiedere all'utente il minimo punteggio e, utilizzando la funzione **`readPoints()`**, leggere da file e memorizzare in **`V`** i dati degli utenti che hanno almeno il punteggio minimo specificato.
Il programma infine deve stampare a video il nome ed il punteggio degli utenti contenuti in **`V`** se e solo se il nome comincia per "Me" **(7 punti).**

ESERCIZIO 2 (10 punti)

Si scriva una funzione **`leggi()`** che faccia inserire da console una serie di numeri interi positivi e li memorizzi in ordine crescente in una lista. L'utente può segnalare la fine della fase di inserimento numeri digitando il valore -1. Si supponga di possedere il tipo di dato astratto **`list`**, con le operazioni primitive associate e le addizionali operazioni **`list insord(element e, list l)`** e **`int member(element e, list l)`** viste a lezione. Tali funzioni possono anche non essere riportate nella soluzione.
Dopo aver caricato i valori nella lista, la funzione **`leggi()`** deve creare una nuova lista contenente i valori letti precedentemente ma senza ripetizioni e deve restituire questa ultima lista come risultato finale.

Si scriva poi una funzione **`stampaLista(list l)`** che, utilizzando la notazione a puntatori (e quindi non utilizzando operazioni primitive), stampi tutti gli elementi di valore dispari contenuti nella lista **`l`**.

ESERCIZIO 3 (6 punti)

Nel caso in cui il programma sorgente C seguente compili ed esegua correttamente, se ne indichino i valori stampati a tempo di esecuzione, motivando la risposta data. In caso di errori di compilazione o errori runtime, si descriva invece nel dettaglio la motivazione di tali errori.
Si indichino inoltre quali sono i blocchi in cui sono visibili le variabili **N** e **L**, specificando se si tratta di variabili locali o globali. Si motivi opportunamente la risposta.

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

int L = 9;
int N = 4;

void cp(char * v, char * w, int start, int end) {
    while ((start < end) && (*w!='\0')) {
      *(v+start) = *w;
      w++;
      start++;
    };
    *(v+start) = '\0';
    return;
}

int main () {
    char temp[] = "Topolino";
    char * nome;
    int N=L;
    int  i;

    nome = (char *) malloc((N) * sizeof(char));
    cp(nome, temp, 0, 13);

    printf("Adesso L vale: %d\n", L);

    for (i=0; (i<N && (*(nome+i)!= '\0')); i++)
      printf("%c", *(nome+i));

    return 0;
}
```

ESERCIZIO 4 (3 punti)

Si presenti brevemente come il linguaggio C effettua il passaggio di parametri nella chiamata a funzione/procedura. In particolare, si illustri che cosa succede nel caso di passaggio di un array e nel caso di passaggio di una struct.

Soluzione Compito A

Esercizio 1

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

#define DIM 32

typedef struct {
    char name[DIM];
    int points;
} user;

int readPoints (char usersFile[], user results[], int maxDim, int minPoints) {
    FILE * f;
    int i;
    int logicDim = 0;

    f = fopen(usersFile, "rb");
    if (f == NULL)
      return -1;

    for (i=0;     i<maxDim &&
                  (fread( &results[logicDim], sizeof(user), 1, f) > 0); i++) {
      if (results[logicDim].points >= minPoints)
            logicDim++;
    }

    fclose(f);
    return logicDim;
}

int main() {

    user * V;
    int i, maxUtenti;
    int logicDim, minPoints;

    printf("Inserire numero massimo di utenti da leggere: ");
    scanf("%d", &maxUtenti);
    V = (user *) malloc(sizeof(user) * maxUtenti);

    printf("Inserire punteggio minimo: ");
    scanf("%d", &minPoints);
    logicDim = readPoints("punti.dat", V, maxUtenti, minPoints);
    if (logicDim < 0)
      exit(-1);

    for (i=0; i<logicDim; i++)
      if ((V[i].name[0] == 'M') && (V[i].name[1] == 'e'))
            printf("L'utente %s ha %d punti.\n", V[i].name, V[i].points);
    free(V);
    return 0;
}
```

Esercizio 2

```
list leggi() {
      int temp;
      list a, result;

      a = emptylist();
      result = emptylist();
      do {
            printf("Inserire numero: ");
            scanf("%d", &temp);
            if (temp >= 0)
                  a = insord(temp, a);
      } while (temp != -1);

      while (!empty(a)) {
            if (!member(head(a), result) )
                  result = insord(head(a), result);
            a = tail(a);
      }
      return result;
}

void stampaLista(list l) {
      if (l==NULL)
            return;
      else {
            if ((l->value %2) == 1)
                  printf("%d\n", l->value);
            stampaLista(l->next);
      }
}
```

Esercizio 3

Il programma stampa:
Adesso L vale: 9
Topolino

Il programma **main** dichiara un array **temp** di caratteri, subito inizializzato come stringa a “Topolino”. Quindi **temp** avrà dimensione 9 (per via del terminatore). Poi viene allocata dinamicamente della memoria, assegnata al puntatore **nome**. In particolare viene allocato spazio per 9 caratteri (infatti la variabile **N**, locale al **main**, è stata inizializzata con il valore di **L**, cioè 9).

Poi viene invocata la funzione **cp**, che copia i caratteri memorizzati in **w**, nel vettore di destinazione **v**, a partire dalla posizione **start** del vettore **v**, fino alla posizione **end** (esclusa). La funzione si ferma in caso il vettore **w** contenga il terminatore di stringa. Al termine del ciclo comunque in **v** viene copiato anche un terminatore di stringa.

Infine il programma main si occupa di stampare il valore di **L** (che vale 9) e poi il contenuto della zona di memoria puntata da **nome**; tale area è stata “riempita”dalla funzione **cp**, e quindi viene stampato “Topolino”.