Fondamenti di Informatica L-A (A.A. 2002/2003) - Ingegneria Informatica
Prof.ssa Mello & Prof. Bellavista – Prova d'esame di lunedì 16/12/2002 – durata 2h:30m
COMPITO A

ESERCIZIO 1 (10 punti)

È dato un file di testo **log.txt** che contiene i dati relativi agli accessi ad un computer, un accesso per riga. Più precisamente, ogni riga contiene, nell'ordine:

- nome utente (non più di 20 caratteri senza spazi), uno e un solo spazio di separazione;
- sigla sistema operativo (non più di 10 caratteri senza spazi), uno e un solo spazio di separazione;
- ora di login (numero intero), uno e un solo spazio di separazione;
- ora di logout (numero intero), terminatore di riga.

Ad esempio, **log.txt**:

BILBO WINXP 10 18
FRODO WIN2000 14 16
SAM LINUX 10 15
GANDALF LINUX 17 21
PIPPO IRIX 22 24
.....

Si scriva una procedura che riceva come parametro di ingresso un file di testo **F** di strutture **accesso** contenenti i dati di accesso e restituisca come parametro di uscita un vettore **Vett** contenente le strutture di **F** relative agli accessi effettuati ad un sistema operativo **SO** per un numero di ore superiore a **ORE** ed il loro numero **N (5 punti)**.

```
void utenti(FILE * F, accesso  Vett[], char SO[], int ORE , int *N);
```

Si scriva inoltre un programma C che inserisca in un vettore **V1** (supposto di dimensione massima **DIM 100**) tutti gli utenti contenuti nel file **log.txt** che hanno fatto accesso al sistema operativo **LINUX** per più di **2** ore utilizzando la procedura **utenti()** precedentemente definita. Il programma deve inoltre stampare il contenuto di **V1** a terminale **(5 punti).**

ESERCIZIO 2 (5 punti)

Si scriva una funzione ricorsiva **window()** che data in ingresso una lista di interi ed un numero **num**, restituisca in uscita una lista contenente esclusivamente gli elementi maggiori di **num-100** e minori di **num+100**.
Si supponga di possedere il tipo di dato astratto **list** e le sue operazioni PRIMITIVE (che quindi possono NON essere riportate nella soluzione).

ESERCIZIO 3 (7 punti)

Dato il programma sorgente C seguente, si indichino i valori stampati a tempo di esecuzione, motivando la risposta data. Indicare inoltre quali sono i blocchi in cui è visibile la variabile M, sempre motivando la risposta.

```
#include <stdio.h>
#define N 7
#define FALSE 0
#define TRUE 1
typedef int vettore[N];

void b(vettore,int,int);
```

```
main ()
{
      int i;
      vettore a;
      for (i = 0; i < N; i++)    a[i]=(i%2 ? N-i : 0);
      b(a, 0, N-2);
      for (i = 0; i < N; i++)    printf ("%d\t", a[i]);
}

void b(vettore v, int iniz, int fine)
{
      int FATTO, I;
      float temp,M=0.0f;
      do {
            FATTO = FALSE;
            for (I = iniz; I <= fine; I++){
                  if (v[I] > v[I+1]){
                        FATTO = TRUE;
                        temp = v[I];
                        v[I] = v[I+1];
                        v[I+1] = temp;
                  }
            }
      } while (FATTO);
}
```

Esercizio 4 (6 punti)
Si consideri la seguente funzione F:

```
int F(int x){
      if (x++ >= 3) return F(x-3)+3;
      else return 0;
}
```

Si scriva il risultato della funzione quando invocata come F(8) e si disegnino i corrispondenti record di attivazione.

Esercizio 5 (2 punti)
Si consideri la grammatica G con scopo S e simboli terminali { **x , (,) , +** }
S::= A
A::= x | (B)
B::= A C
C::= + A

Si dica se la stringa **(x + x)** è sintatticamente corretta rispetto a tale grammatica e se ne mostri la derivazione left most.

Esercizio 6 (2 punti)
Un elaboratore rappresenta i numeri interi su 8 bit dei quali 7 sono dedicati alla rappresentazione del modulo del numero e uno al suo segno. Indicare come viene svolta la seguente operazione aritmetica e determinarne il risultato traslandolo poi in decimale per la verifica:

110-23

Fondamenti di Informatica L-A (A.A. 2002/2003) - Ingegneria Informatica
Prof.ssa Mello & Prof. Bellavista – Prova d'esame di lunedì 16/12/2002 – durata 2h:30m
COMPITO B

ESERCIZIO 1 (10 punti)

È dato un file di testo **stat.txt** che contiene i dati relativi alle statistiche dei giocatori di pallamano di una squadra, un giocatore per riga. Più precisamente, ogni riga contiene, nell'ordine:

- numero maglia (numero intero), uno e un solo spazio di separazione;
- cognome (non più di 30 caratteri senza spazi), uno e un solo spazio di separazione;
- numero di gol segnati (numero intero), uno e un solo spazio di separazione;
- numero medio di assist per partita (numero float), terminatore di riga.

Ad esempio, **stat.txt**:

07 BELLAVISTA 10 9.75
13 FRODO 14 2.00
99 SAM 10 0.75
5 MELLO 17 3.40
6 PIPPO 22 12.00
.....

Si scriva una procedura che riceva come parametro di ingresso un file di testo **F** di strutture **stat** contenenti le statistiche di una squadra di pallamano e restituisca come parametro di uscita un vettore **Vett** contenente le strutture di **F** relative ai giocatori con gol segnati superiori a **GOL** e assist per partita inferiore a **ASSIST** ed il loro numero **N (5 punti)**.

```
void pallamano(FILE * F, stat  Vett[], int GOL, float ASSIST, int *N);
```

Si scriva inoltre un programma C che inserisca in un vettore **V1** (supposto di dimensione massima **DIM 100**) tutte le strutture **stat** contenute nel file **stat.txt** per i giocatori che hanno segnato più di **4** gol avendo una media assist/partita inferiore a **5.50**, utilizzando la procedura **pallamano()** precedentemente definita. Il programma deve inoltre stampare il contenuto di **V1** a terminale **(5 punti).**

ESERCIZIO 2 (5 punti)

Si scriva una funzione ricorsiva **dup(list l, int num)** che data in ingresso una lista di interi ed un numero **num**, restituisca in uscita una lista contenente tutti gli elementi di valore uguale a **num** duplicati. Ad esempio, se invocata con l=[0,7,3,2,7,1] e num=7, la funzione dup() deve restituire [0,7,7,3,2,7,7,1].
Si supponga di possedere il tipo di dato astratto **list** e le sue operazioni PRIMITIVE (che quindi possono NON essere riportate nella soluzione).

ESERCIZIO 3 (7 punti)

Dato il programma sorgente C seguente, si indichino i valori stampati a tempo di esecuzione, motivando la risposta data. Indicare inoltre quali sono i blocchi in cui è visibile la variabile M, sempre motivando la risposta.

```
#include <stdio.h>
#define N 10
#define FALSE 0
#define TRUE 1
typedef int vettore[N];

void a(int, int, vettore);
```

```
main ()
{
      int i;
      vettore v;
      for (i = 0; i < N; i++)    v[i]=(i%2 ? 0 : N+i);
      a(2, N-4, v);
      for (i = 0; i < N; i++)    printf ("%d\t", v[i]);
}

void a(int iniz, int fine, vettore v)
{
      int FATTO, I;
      float temp,M=0.0f;
      do {
            FATTO = FALSE;
            for (I = iniz; I <= fine; I++){
                  if (v[I] > v[I+1]){
                        FATTO = TRUE;
                        temp = v[I];
                        v[I] = v[I+1];
                        v[I+1] = temp;
                  }
            }
      } while (FATTO);
}
```

Esercizio 4 (6 punti)
Si consideri la seguente funzione F:

```
int G(float y){
      if (y*2 >= 3) return G(y-3)+3;
      else return 0;
}
```

Si scriva il risultato della funzione quando invocata come G(7.50) e si disegnino i corrispondenti record di attivazione.

Esercizio 5 (2 punti)
Si consideri la grammatica G con scopo S e simboli terminali **{ $, € , (,) , - , + }**
S::= A
A::= $ | (B) | €
B::= A C
C::= + A

Si dica se la stringa **($ + €)** è sintatticamente corretta rispetto a tale grammatica e se ne mostri la derivazione left most.

Esercizio 6 (2 punti)
Un elaboratore rappresenta i numeri interi su 8 bit dei quali 7 sono dedicati alla rappresentazione del modulo del numero e uno al suo segno. Indicare come viene svolta la seguente operazione aritmetica e determinarne il risultato traslandolo poi in decimale per la verifica:

57+46

SOLUZIONE COMPITO A

ESERCIZIO 1

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define DIM 100

typedef struct {char nome[21]; char SO[11]; int login; int logout;}
    accesso;

void utenti(FILE * F, accesso Vett[], char SO[], int ORE, int *N) {
    accesso in; *N=0;

    while (fscanf(F,"%s %s %d %d\n",  in.nome, in.SO, &in.login,
    &in.logout) != EOF) {
      if ( (strcmp(SO,in.SO)==0) && ((in.logout-in.login)>=ORE) )
      {    Vett[*N]=in;
           *N= *N+1; }
    }
}

main() {
    accesso V1[DIM];
    int Num, i=0;
    FILE* f;

    if ((f=fopen("c:\\log.txt", "r"))==NULL) {
      printf("Il file non esiste!"); exit(1); }
    utenti(f, V1, "LINUX", 2, &Num);
    for (i=0; i<Num; i++)
      printf("%s %s %d %d\n", V1[i].nome, V1[i].SO, V1[i].login,
    V1[i].logout);
    fclose(f);
}
```

ESERCIZIO 2

```
list  window(list l, int num) {
      if empty(l) return emptylist();
      else if ((head(l)>num-100) && (head(l)<num+100))
           return cons(head(l), window(tail(l), num));
      else return window(tail(l), num);
}
```

ESERCIZIO 3

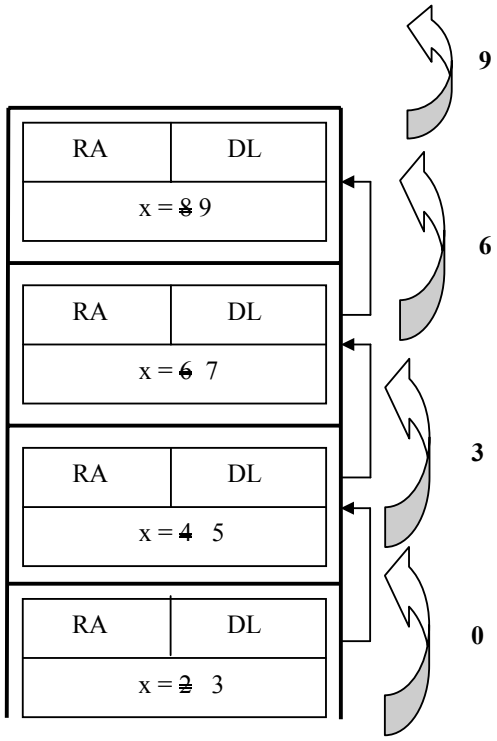
Il programma stampa:

0 0 0 0 2 4 6

Infatti il vettore a viene inizializzato con il seguente contenuto: {0,6,0,4,0,2,0}. Successivamente viene invocata la procedura b alla quale si passa come parametro il vettore a. La procedura b definisce un algoritmo di ordinamento di tipo bubblesort: perciò non fa altro che restituire l'array ordinato. In particolare la procedura b ripete un ciclo (do-while) fino a quando tutti gli elementi del vettore non saranno ordinati in maniera crescente. Ad ogni ciclo 'do-while' si innesca un ciclo 'for' che permette al primo elemento fuori sequenza di essere

scambiato con il successivo fino a quando non viene confrontato con un elemento maggiore.
La variabile M è una variabile locale alla funzione b, perciò il suo campo di visibilità è limitato alla sola funzione.

ESERCIZIO 4
La funzione restituisce il valore 9.



Esercizio 5
La stringa è sintatticamente corretta.
Derivazione *left most*:
`S -> A -> ( B ) -> ( A C ) -> ( x C ) -> ( x + A ) -> ( x + x )`

Esercizio 6
110-> 0 1101110
-23-> 1 0010111

Tra i moduli dei numeri si esegue una sottrazione:

```
    1101110
-   0010111
    ---------
    1010111
```

che vale 87 in base dieci.

SOLUZIONE COMPITO B

ESERCIZIO 1

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define DIM 100

typedef struct {int maglia; char cognome[31]; int gol; float assist;} stat;

void pallamano(FILE * F, stat Vett[], int GOL, float ASSIST, int *N) {
    stat in; *N=0;

    while(fscanf(F,"%d %s %d %f\n",   &in.maglia, in.cognome, &in.gol,
    &in.assist) != EOF) {
      if ((in.gol>GOL) && (in.assist>ASSIST))
      {    Vett[*N]=in;
           *N= *N+1; }
    }
}

main() {
    stat V1[DIM];
    int Num, i=0;
    FILE* f;

    if ((f=fopen("c:\\stat.txt", "r"))==NULL) {
      printf("Il file non esiste!"); exit(1); }
    pallamano(f, V1, 4, 5.50, &Num);
    for (i=0; i<Num; i++)
      printf("%d %s %d %f\n", V1[i].maglia, V1[i].cognome, V1[i].gol,
    V1[i].assist);
    fclose(f);
}
```

ESERCIZIO 2

```
list  dup(list l, int num) {
      if empty(l) return emptylist();
      else if (head(l)!=num) return cons(head(l), dup(tail(l), num));
           else return cons(head(l), cons(head(l), dup(tail(l), num)));
}
```

ESERCIZIO 3

Il programma stampa:

10 0 0 0 0 12 14 16 18 0

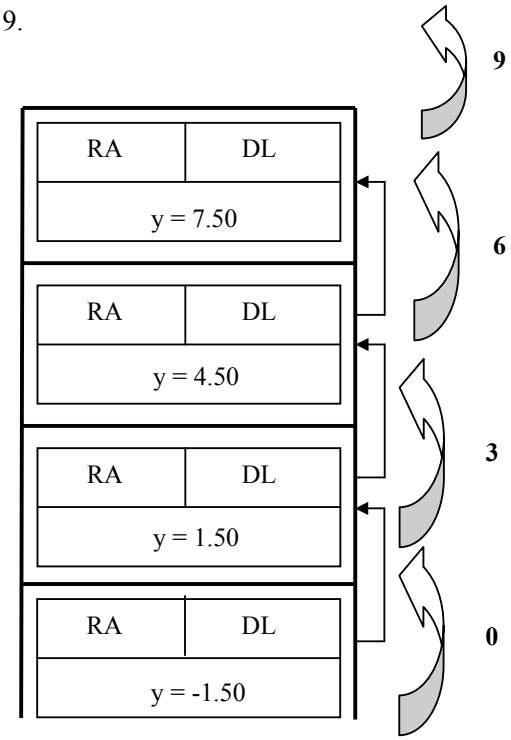
Infatti il vettore v viene inizializzato con il seguente contenuto: {10,0,12,0,14,0,16,0,18,0}. Successivamente viene invocata la procedura a alla quale si passa come parametro il vettore v.

La procedura a definisce un algoritmo di ordinamento di tipo bubblesort perciò non fa altro che restituire l'array ordinato. In particolare la procedura a cicla (do-while) fino a quando tutti gli elementi del vettore non saranno ordinati in maniera crescente. Ad ogni ciclo 'do-while' si innesca un ciclo 'for' che permette al primo elemento fuori sequenza di essere scambiato con il successivo fino a quando non viene confrontato con un elemento maggiore.

La variabile M è una variabile locale alla funzione b, perciò il suo campo di visibilità è limitato alla sola funzione.

ESERCIZIO 4

La funzione restituisce il valore 9.



Esercizio 5

La stringa è sintatticamente corretta.
Derivazione left most:

```
S -> A -> ( B ) -> ( A C ) -> ( $ C ) -> ( $ + A ) -> ( $ + € )
```

Esercizio 6

57-> 0 0111001
46-> 0 0101110

Tra i moduli dei numeri si esegue una somma:

```
   0111001
+  0101110
   ---------
   1100111
```

che vale 103 in base dieci.