Fondamenti di Informatica - A.A. 2017-2018
Prof. Vincenzo Caglioti
Appello del **26/01/2018**

POLITECNICO MILANO 1863

Cognome	**Nome**	**Matricola**	**Voto:** . . . /**30**

Quesito:	1	2	3	4	5		Tot.
Max:	5	6	6	8	5		30
Punti:							

Istruzioni:

- Per superare la prova è necessario conseguire almeno 18/30;
- non è consentito consultare libri, appunti, la calcolatrice o qualsiasi dispositivo elettronico, né comunicare;
- **in caso di copiatura (sia rilevata durante la prova sia in fase di correzione) a tutti gli studenti coinvolti verrà annullata la prova e non sarà consentita la partecipazione al prossimo appello;**
- tempo a disposizione: 2h 00m.

Stile del codice C:

- i commenti non sono necessari, ma potrebbero essere utili in caso di errore;
- è interesse dello studente assegnare a variabili, costanti e parametri nomi che ne rendano chiaro l'utilizzo nel programma.

INIZIARE LA SOLUZIONE DI OGNI ESERCIZIO SU UNA PAGINA

RESTITUIRE COMPILATO ANCHE NEL CASO IN CUI CI SI RITIRA

Quesito 1 (5 punti)
Si consideri il numero razionale 1/12. Calcolarne la mantissa (compresa tra 1 e 2) e l'esponente in base 2; fornire la rappresentazione del valore -1/12 in virgola mobile in singola precisione secondo il formato IEEE 754.

Mantissa 4/3 = 1,01010101.. ; Esponente: -4; Rappresentazione: |1, 01111011, 01010101010101010101010 |

Quesito 2 (6 punti)
Si spieghi sinteticamente in che cosa consistono, come si generano, e come si evitano i seguenti problemi: la produzione di spazzatura, i riferimenti fluttuanti ("dangling references") e i riferimenti a variabili automatiche.

Quesito 3. (6 punti). Si scriva una **procedura** che, ricevendo in ingresso una lista dinamica di caratteri, cancelli l'elemento in posizione centrale, scandendo la lista stessa solo una volta. Detto N il numero di elementi della lista, l'elemento in posizione centrale è il K-esimo, dove K è la parte intera di (N+1)/2. Se N=0 non occorre effettuare cancellazioni

```
typedef struct EL {char info;
                   struct EL *prox;} Elemento;
typedef Elemento *Lista;
void CancellaCentro (Lista *pL)
{       Elemento **prec;
        Elemento *corr, *temp;
        int n=0;

        prec = pL;
        corr = *pL;
        if (*pL == NULL) return;
        while (corr != NULL)
        {       n++;
                corr = corr->prox;
                if (n%2 ==1 && n/2 > 0) prec = &((*prec)->prox);
        }
        temp = *prec;
        *prec = temp->prox;
        free(temp);
}
```

Quesito 4 (**8 punti**). Una lista bidirezionale è una lista dinamica costituita da elementi ciascuno dei quali contiene, oltre al campo informativo, due campi puntatore: uno all'elemento successivo nella lista e uno all'elemento precedente. In una lista bidirezionale, così come l'ultimo elemento della lista ha il suo puntatore all'elemento successivo pari a NULL, il primo elemento ha il puntatore all'elemento precedente pari a NULL. Si scriva una **funzione ricorsiva** che, ricevendo in ingresso una classica lista dinamica monodirezionale, costruisca la lista bidirezionale contenente gli stessi elementi presenti nella lista in ingresso (nello stesso ordine) e la restituisca al chiamante.

```
typedef struct EL {float info;
                   struct EL *next;} ElemLista;
typedef ElemLista *Lista;

typedef struct EL {float info;
                   struct  EL*prox;
                   struct EL *prec;} ElemBidir;
typedef ElemBidir *ListaBidir;

ListaBidir  CostruisciListaBidirezionale (Lista L)
{
  ListaBidir L1, L2;

  if (L == NULL) return (NULL);

  L1 = CostruisciListaBidirezionale (L-> next);
  L2 = malloc(sizeof(ElemBidir));
  L2 -> info = L -> info;
  L2 -> prox = L1;
  L2 -> prec = NULL;
  if (L1 != NULL) L1 -> prec = L2;
  return (L2);
}
```

Quesito 5 (5 punti). Si scriva un programma C che svolge le seguenti operazioni:

1. inserisce (con un metodo a piacere) le cifre della matricola dello studente in un array di char chiamato Matricola;
2. apre (come file di testo) un file chiamato "input.txt" e confronta i caratteri in esso contenuti con gli elementi di Matricola;
3. stampa a schermo (rispettivamente) "OK" oppure "KO" a seconda che tra i caratteri contenuti nel file siano compresi o meno tutti gli elementi di Matricola, eventualmente ripetuti.
   Il programma stampa "KO" anche nel caso in cui risulti impossibile aprire il file.

NOTA. In C, int e char sono entrambi tipi integral, ovvero assimilabili a numeri interi: per tale ragione è possibile confrontare tra loro (con ==, <, <=, >, >=) un int e un char. Lo stretto legame tra int e char consente anche ad alcune funzioni C per la lettura di caratteri da file di restituire un int: questo passaggio da char a int non altera il valore numerico del dato, per cui se si confrontano il dato int restituito e il dato char originale essi risultano uguali.

```
#define NOMEFILE "./input.txt"
#define MATRICOLA "123456"
/* numero di matricola dello studente - vedere anche il parametro NUMCIFRE */
#define NUMCIFRE 6
/* numero di cifre che compongono MATRICOLA */

#include <stdio.h>

int main()
{
  typedef enum {false, true} boolean;
```

```
char Matricola[NUMCIFRE] = MATRICOLA;
/* vedere testo dell'esercizio */
boolean GiaTrovato[NUMCIFRE];
/* l'elemento di indice k vale false [true] se il carattere di indice k
   in Matricola non e' stato ancora trovato [e' stato trovato] nel file */
int Cont;
/* usato per scorrere gli array */
int CarattereLetto;
/* ultimo carattere letto dal file, rappresentato come intero (in
   conformita' al modo di operare della funzione fgetc) */
FILE *Stream;
boolean Esito;
/* vale true se tutti i caratteri della matricola sono stati trovati nel
   file, false altrimenti */

for (Cont = 0; Cont < NUMCIFRE; Cont++)
{
  GiaTrovato[Cont] = false;
}

Stream = fopen(NOMEFILE, "r");

if (NULL != Stream)
{
  CarattereLetto = fgetc(Stream);

  while (EOF != CarattereLetto)
  {
    printf("\n\n###DEBUG Ho letto il carattere %d, corrispondente a '%c'", CarattereLetto, CarattereLetto);

    for (Cont = 0; Cont < NUMCIFRE; Cont++)
    {
      if (CarattereLetto == Matricola[Cont])
      {
        GiaTrovato[Cont] = true;
      }
    }
    CarattereLetto = fgetc(Stream);
  }

  fclose(Stream);
}

Esito = true;

for (Cont = 0; Cont < NUMCIFRE; Cont++)
{
  if (false == GiaTrovato[Cont])
  {
    Esito = false;
  }
}

if (true == Esito)
{
  printf("\nOK\n");
```

```
  }
  else
  {
    printf("\nKO\n");
  }

  return 0;
}
```