Fondamenti di Informatica - A.A. 2013-2014
Ingegneria Informatica
Scuola di Ingegneria Industriale e dell'Informazione
Prof.ssa Cristiana Bolchini
Appello del **09/07/2014**

RESTITUIRE COMPILATO

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

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

Istruzioni:

- non è possibile consultare libri, appunti, la calcolatrice o qualsiasi dispositivo elettronico, né comunicare;
- si può scrivere con qualsiasi colore, anche a matita, ad eccezione del **rosso**.
- tempo a disposizione: 2h 15m

Stile del codice C:

- non è necessario inserire direttive `#include`;
- i commenti non sono necessari, ma potrebbero essere utili nel caso di errore;

Quesito 1 (4 punti) Punteggio ottenuto: .../4

Dati i due numeri A = $-4A_{16MS}$ e B = -59_{10MS} effettuare la conversione in base 2, notazione complemento a 2 (2C2), sul numero minimo di bit necessari a rappresentare entrambi gli operandi. Si effettuino quindi le operazioni A+B e A-B indicando esplicitamente se si verifica overflow o meno, e motivando la risposta. Mostrare i passaggi fatti.

Quesito 2 (4 punti) Punteggio ottenuto: .../4

Dato il numero A = -2.015625_{10MS} convertirlo in base 2, notazione IEEE 754, singola precisione, riportando tutti i bit della codifica. Rappresentare inoltre il valore convertito in base 16. Mostrare i passaggi.

Quesito 3 (5 punti) Punteggio ottenuto: .../5

Scrivere un programma che acquisisce tre interi che rappresentano anno, mese e giorno. Il programma calcola e visualizza il numero del giorno a partire dall'inizio dell'anno (per esempio il 3 febbraio corrisponde al giorno 34). Prestare attenzione agli anni bisestili.

Esempio:

```
Ingresso:  2014 2 3
Uscita:    34
```

Quesito 4 (5 punti) Punteggio ottenuto: .../5

Un numero perfetto è pari alla somma dei suoi divisori: per esempio, i divisori di 28 sono 1, 2, 4, 7 e 14 e 1+2+4+7+14 = 28, quindi 28 è un numero perfetto. Dati due numeri `m` e `n` questi sono numeri *amicali* se la somma dei divisori di `m` è uguale a `n`, e viceversa.

Scrivere un sottoprogramma `divisori` che ricevuto in ingresso un numero intero restituisce l'insieme dei suoi divisori in una variabile *opportunamente* dimensionata. Scrivere poi un ulteriore sottoprogramma `amicali` che ricevuto in ingresso due numeri interi restituisce 1 se i numeri sono amicali, 0 altrimenti.

Quesito 5 (6 punti) Punteggio ottenuto: .../6

Scrivere un programma che chiede all'utente il nome di 2 file contenenti ognuno una matrice 100x100 di numeri interi. Il programma legge le informazioni dai file sorgenti e crea un nuovo file (anche questo nome viene chiesto all'utente) contenente una matrice 100x100 i cui elementi sono calcolati ciascuno come la media dei valori dei corrispondenti elementi delle matrici sorgenti. Più chiaramente, il valore dell'elemento `i, j` della matrice risultato, è calcolato come la media dei valori degli elementi `i, j` delle 2 matrici sorgenti. I nomi dei file, completi di percorso ed estensione sono di al più 60 caratteri. I file, se esistono, contengono senz'altro 100x100 elementi.

Quesito 6 (6 punti) Punteggio ottenuto: . . . /6

Si consideri una versione semplificata della battaglia navale in cui le navi possono essere posizionate solo in orizzontale. Il campo di gioco di un singolo giocatore può essere rappresentato tramite la matrice `CampoGioco` di dimensione 10x10 in cui ogni cella della matrice assume valore 1 (presenza di una nave) o 0 (mare). Le navi possono essere lunghe una, due, tre, quattro o cinque celle. Ad esempio, nell'istanza della matrice `CampoGioco` c'è un campo di gioco in cui sono presenti 4 navi (per questioni di spazio, il campo è di dimensione 5x5): una nave lunga 4 nella prima riga, una nave lunga 1 nella terza riga, una nave lunga 2 e una lunga 1 nella quarta riga e una nave lunga 4 nella quinta riga.

0	1	1	1	1
0	0	0	0	0
0	0	0	1	0
1	1	0	0	1
1	1	1	1	0

Si scriva un sottoprogramma `analisiCampo` che – ricevuto in ingresso l'array bidimensionale `CampoGioco` e qualsiasi altro parametro ritenuto necessario – svolga, nell'ordine, le seguenti attività:

- visualizzi *per ogni riga*, il numero di navi presenti e la loro lunghezza
- visualizzi il numero di navi complessivamente presenti
- visualizzi la dimensione della nave più corta e di quella più lunga
- visualizzi, *per ogni lunghezza di nave trovata* il numero di navi di tale lunghezza.

Con riferimento al campo di gioco in questione, il sottoprogramma visualizzerà:

```
Riga 1: 1 nave lunga 4
Riga 3: 1 nave lunga 1
Riga 4: 1 nave lunga 2 1 nave lunga 1
Riga 5: 1 nave lunga 4

Totale navi: 5

Lunghezza nave piu' corta trovata: 1
Lunghezza nave piu' lunga trovata: 4

Numero di navi lunghe 1: 2
Numero di navi lunghe 2: 1
Numero di navi lunghe 3: 0
Numero di navi lunghe 4: 2
Numero di navi lunghe 5: 0
```

Nota. Non è necessario sviluppare i sottoprogrammi `instesta`, `inscoda`, `insordinato`, `del`, `lunghezzalista`, `esiste`.