Corso di Informatica@Matematica
Prova scritta del 17 settembre 2013

Tempo a disposizione: ore 2.
**Svolgere gli esercizi (1, 2, 3) e (4, 5) su due fogli differenti.**
**Gli esercizi 3 e 5 sono condizione necessaria per la sufficienza.**

1. (6 punti) La stringa `1111 1011 1111 1110` (gli spazi sono inseriti per semplicità di lettura) è la codifica in complemento a due (16 bit) di un intero $n$ :
(i) $n$ è pari o dispari?
(ii) qual è il valore in decimale di $-n$?
(iii) qual è la rappresentazione esadecimale assoluta di $-n$?
Giustificare.

2. (6 punti) Voglio invertire una lista (il primo elemento diventa l'ultimo, ecc.) lasciando però fermi gli elementi di indice dispari (si suppone 0 pari). È corretta la seguente funzione? Motivare.

```
def invpari(L):
    for i in range(0,len(L)//2,2):
        L[i],L[len(L)-1-i]=L[len(L)-1-i],L[i]
    return L
```

3. (7 punti) Una *catena* è una lista `C` di coppie `(d,i)`, in cui `d` è un dato qualsiasi (di cui non ci interessa il tipo) e `i` è un *puntatore* intero con `1<=i<=len(C)`. Data una catena `C` ed un suo elemento `(d,i)`, diciamo che `C[i]` è il *successore* di `(d,i)`; se `i` è `len(C)` (e quindi `C[i]` non esiste), `(d,i)` non ha successore ed è l'*ultimo* elemento della catena. Possiamo elencare (alcuni dei) dati (cioè la prima componente delle coppie) di una catena secondo l'ordine definito dalla nozione di successore (che chiameremo *ordine della catena*): primo l'elemento `C[0]`, poi il suo successore, quindi il successore di quest'ultimo e così via, fino a che non si incontra l'ultimo elemento.
Esempi: (1) l'ordine della catena per `C1=[(10,3),(20,2),(30,4),(40,1),(20,5)]` è `10,40,20,30,20`.
(2) L'ordine della catena per `C2=[(10,3),(20,4),(30,3),(40,1)]` è `10,40,20` (`30` non viene elencato perché si incontra prima un elemento senza successore.)
(3) L'ordine della catena per `C3=[(10,3),(20,2),(30,1),(30,1)]` è `10,30,20,30,20,30,20,30,...`, perché i puntatori producono un ciclo.

   (i) Una catena è lineare se l'ordine di catena non ha cicli: `C1` e `C2` sono lineari, `C3` non è lineare. Si scriva una funzione Python `elenca_lin(C)` che prende come parametro una catena `C` lineare, e restituisce l'ordine di catena di `C`, in forma di lista. Ad esempio, `elenca_lin(C1)` deve restituire `[10,40,20,30,20]`.

   (ii) Si scriva una funzione Python `elenca(C)` che prende come parametro una catena `C` (non necessariamente lineare) e restituisce l'ordine di catena di `C`, in forma di lista, eventualmente troncato alla prima ripetizione dei puntatori se vi è un ciclo. Ad esempio `elenca(C3)` deve restituire `[10,30,20,30]`.

4. (6 punti) Cosa stampa il seguente frammento di codice Python:

```
a=[0,1,[2,3]]
b=a[2]
a[1]=a
a[1][1][1][2][0]=5
print(b)
```

5. (7 punti) Si supponga di avere a disposizione le seguenti funzioni Python, che permettono di costruire e manipolare alberi (come sono realizzate non è significativo per l'esercizio):

```
def bin(a,b,c):       # Costruisce (restituisce) l'albero con radice etichettata a,
                      # figlio sinistro b e figlio destro c
def empty():          # Restituisce l'albero vuoto
def is_empty(a):      # True sse l'argomento è l'albero vuoto
def first_child(a):   # Restituisce una copia del figlio sinistro dell'albero a
def second_child(a):  # Restituisce una copia del figlio destro dell'albero a
def label(a):         # Restituisce l'etichetta della radice dell'albero a
```

   In un albero, l'*altezza* di un nodo `u` è la lunghezza dell'unico cammino che congiunge `u` alla radice. Per esempio, l'albero `bin(2,bin(3,empty(),empty()),bin(4,empty(),empty()))` ha zero nodi di altezza 2, due nodi di altezza 1 e un solo nodo di altezza 0 (la radice). Manipolando alberi *solo* con le funzioni assegnate, si scriva una funzione Python `numalt(a,h)` che dato un albero `a` e un naturale `h` restituisca il numero dei nodi di `a` che si trovano ad altezza `h`.