Programmazione 1 — Esercitazione 5

Cognome: **Nome:** **Matricola:**

1. Scrivere un predicato `Equal(As, Bs)` che controlla se le due liste `As` e `Bs` sono uguali elemento per elemento (**nota:** usare le pairslist).

2. Scrivere una funzione `Intersect(As, Bs)` che prende in input due tuple e restituisce una tupla che contiene gli elementi che sono presenti sia nella prima che nella seconda tupla. Utilizzare la funzione di test seguente:

```
1 def TestZero():
2     if Intersect((2,3,4,2,1,2,7), (2,3,2,3,4)) == (2, 3, 4):
3         return 'ok'
4     return 'failed'
5 print('Test zero: '+TestZero())
```

3. Si scriva una funzione `RimuoviDuplicati(As, Bs)` che prenda in input due liste e che rimuove dalla prima lista ogni elemento che compare nella seconda lista. La funzione non ritorna nulla, ma modifica la prima lista data in input. Usare le liste di Python e usare il metodo `L.remove(e)` che rimuove dalla lista `L` il primo elemento uguale a `e`.

```
1 def RimuoviDuplicati(As, Bs):
2     pass  # DA COMPLETARE
3 L1 = [2, 4, 2, 5, 6, 6, 3, 2, 9, 4]
4 L2 = [2, 4, 7]
5 RimuoviDuplicati(L1, L2)
6 print('L1 =', L1)
```

4. Scrivere una funzione `SortIncreasing(As)` che prende in input una lista di numeri `As` e restituisce una nuova lista con gli stessi elementi di `As`, ma ordinati in modo crescente (**nota:** usare le pairslist).

5. Scrivere una funzione `Sort(As, C)` che prende in input una lista di numeri `As` e una funzione di confronto tra due elementi della lista, e restituisce una nuova lista con gli stessi elementi di `As`, ma ordinati in base al criterio dato dalla procedura `C` (**nota:** usare le pairslist).

```
1 print(Sort(MakeRandInts(10,1,100), lambda x, y: x < y)) # Esempio di utilizzo
```

6. Si scriva una funzione `CalcolaFrequenza(D)` che prende in input un dizionario `D` che contiene una chiave per ogni carattere e come valore il numero di volte che il carattere appare in un dato testo, e restituisca un nuovo dizionario che contiene per ogni carattere (chiave) di `D` la sua percentuale di presenza nel testo. Usare il file di testo `LeAvventureDiPinocchio.txt` e calcolare le frequenze di ciascun carattere.

7. Si scriva una funzione `ComputeTable(Ls)` che prende in input una lista di n caratteri (alfabeto) e per ogni carattere costruisca due dizionari, il primo chiamato `char2int` che ha come chiave un carattere e come valore un numero identificativo unico compreso in $[1, n)$, e il secondo dizionario ha come chiave un numero e come valore il carattere di cui è la codifica. La funzione deve restituire i due dizionari. Per esempio:

```
char2int, int2char = ComputeTable('abc')
# dovrebbe essere: char2int = {'a': 0, 'b': 1, 'c': 2}
# dovrebbe essere: int2char = {0: 'a', 1: 'b', 2: 'c'}
print(char2int['a'], int2char[2]) # Deve stampare 0 e c
```

8. **PROGRAMMA 1:** Il cifrario di Cesare è uno dei più vecchi (semplice e insicuro) algoritmo di criptografia. L'idea di cifratura consiste nel sostituire ogni carattere di un messaggio con un carattere che nell'alfabeto occupa 3 cifre in avanti, ripartendo dall'inizio se si dovesse superare l'ultima lettera. Per esempio, una 'a' diventa una 'd' e una 'z' diventa una 'c'. Per decriptare un messaggio basta compiere l'operazione inversa usata per la cifratura. Per esempio, `buono studio!` diventa `exrqr vwxglr!`

   Scrivere un programma per decifrare il testo contenuto nel file `testo_segreto_facile.txt`. Suggerimento: calcolare e usare le frequenze di ciascun carattere contenuto nel libro "Le avventure di Pinocchio", usare le funzioni `ContaCaratteri`, `CalcolaFrequenza` e `ComputeTables`.

9. **CHALLENGE 3:** Decifrare il testo seguente, non generato con un cifrario di Cesare, che trovate nel file `testo_segreto_difficile.txt`:

   — ab, ab, ab f vbw ab. bczjzw xb vcbjfllb zqqz jwz tgbaz ezmz sw swrfamzcf ga czhzoob vfctfaf, f rbhqwb jzamfafcf qz vcbjfllz. zaow, lwppbjf rfsb pxf wq lbqf rz lbmmb, pbly mw qzlpwb lgtwmb f lpzvvb rwz. sgaugf zsswb, f tgba rwzhhwb.

   Per il **PROGRAMMA** e la **CHALLENGE** potete mandare il testo in chiaro e l'implementazione in python per email (facoltativo).