Menù

- Esercizio sulle liste
- Esercizio sulle tuple
- Esercizi sui dizionari
- Esercizi su classi e dizionari
- Progetti

Esercizi sulle liste

Realizza una funzione per la ricerca di un numero X in una matrice M (realizzata come liste annidate). Restituisci le coordinate di tale elemento come tupla.

```
def ricerca(A,X):
  N=len(A)
  M=len(A[0])
  i=0
  while (i<N):
    j=0
    while (j<M):
      if (A[i][j]==X):
        return (i,j)
      j=j+1
    i=i+1
  return 0
```

Esercizio sulle tuple

Realizza una funzione che prenda in input una tupla di numeri interi T e un'intero N e restituisca una tupla identica alla prima se non nel suo N-esimo elemento, sostituito con la sua espressione verbale.

Esempio: cambia((1,2,3),0) restituirà (“uno”,2,3).

```
def cambia(A,N):
  P="nove"
  if (A[N]==0):
    P="zero"
  elif (A[N]==1):
    P="uno"
  elif (A[N]==2):
    P="due"
  ...
  ...
  ...
  elif (A[N]==8):
    P=”otto”
  B=A[0:N]+(P,)+A[N+1:]

return B
```

Università degli Studi di Bologna–Polo di Forlì
Laboratorio di Fondamenti di Programmazione
Tutor: Dott. Gianluigi Roveda

Esercizi sui dizionari

Realizza una funzione che effettui una conversione tra date mantenute in tuple. La forma iniziale sarà della forma:

(10,3,1997) per indicare la data 10/3/1997

La forma finale dovrà essere: (decimo, Marzo,1997).

La funzione dovrà sfruttare i dizionari.

```
def converti_giorno(X):
  G={1:"primo",2:"secondo",3:"terzo", ...}
  return G[X]

def converti_mese(X):
  M={1:"gennaio",2:"febbraio",3:"marzo",4:"aprile", ...}
  return M[X]

def converti_data(D):
  gg=converti_giorno(D[0])
  mm=converti_mese(D[1])

  return (gg,mm,D[2])
```

Università degli Studi di Bologna–Polo di Forlì
Laboratorio di Fondamenti di Programmazione
Tutor: Dott. Gianluigi Roveda

Esercizi sui dizionari

Realizza una funzione dizionario inglese-italiano con almeno 10 termini (es.: apple, fruit, book, cup, day, east, green,hot, include, lemon) che restituisca 0 se la parola da tradurre non è presente nel dizionario (si ricorda l'esistenza del metodo has_key), altrimenti il termine tradotto.

```
def converti_ii(p):
  D={'apple':'mela','tea':'the','sun':'sole','music':'musica','sphere':'sfera'}
  X=D.has_key(p)
  if (X==1):
    return D[p]
  else:
    return 0
```

Esercizi su classi e dizionari

Generalizziamo: implementa una classe per l'inserimento di nuovi vocaboli e consultazione di una dizionario inglese - italiano.

```
class traduttore:
  diz_enit={}
  diz_iten={}
  errore=0

  def inserisci(self,a, b):
    self.diz_enit[a]=b
    self.diz_iten[b]=a

  def cerca_it(self,a):
    if self.diz_enit.has_key(a):
      return self.diz_enit[a]
    else:
      self.errore=1
      return -1

  def cerca_en(self,a):
    if self.diz_iten.has_key(a):
      return self.diz_iten[a]
    else:
      self.errore=2
      return -1
```

Esercizi su classi e dizionari

Alla soluzione dell'esercizio precedente aggiungere un metodo per l'eliminazione di una voce del vocabolario. Attenzione non basta cancellare la voce inglese, bisogna cancellare anche quella italiana.

```
def cancella(self,a):
  if self.diz_iten.has_key(a):
    b=self.diz_iten[a]
    if self.diz_enit.has_key(b):
      del self.diz_iten[a]
      del self.diz_enit[b]
      return 1
    else:
      errore=3
      return -1
elif self.diz_enit.has_key(a):
  b=self.diz_enit[a]
  if self.diz_iten.has_key(b):
    del self.diz_iten[b]
    del self.diz_enit[a]
    return 1
  else:
    errore=3
    return -1
else:
    errore=4
    return -1
```

Università degli Studi di Bologna–Polo di Forlì
Laboratorio di Fondamenti di Programmazione
Tutor: Dott. Gianluigi Roveda

Esercizi su classi e dizionari

Realizza una classe figlia della prima che possa anche contenere il significato in italiano delle voci contenute e si estenda il metodo inserisci in modo opportuno sfruttando il polimorfismo di Python.

```
class traduttore_v2(traduttore):

significati={}

def inserisci(self,a,b,c):
  self.diz_enit[a]=b
  self.diz_iten[b]=a
  self.significati[b]=c

def cerca_significati(self,a):
  if self.significati.has_key(a):
    return self.significati[a]
elif self.diz_enit.has_key(a):
  b=self.diz_enit[a]
  if self.significati.has_key(b):
    return self.significati[b]
  else:
    errore=5
    return -1
else:
  errore=6
  return -1
```

Proposte di progetto

Primo livello:

1. Programma per codificare e decodificare messaggi secondo il codice di Giulio Cesare con passi differenti
2. Giocotore automatico carta-sasso-forbice
3. Programma calcolatrice (per due persone si può anche fare a condizione di aggiungere le conversioni in ottale e esadecimale nei vari versi) con tasti per somma, sottrazione, moltiplicazione, divisione, scrittura in memoria del valore sul display, lettura del valore in memoria, radice, quadrato, inversione di segno, on/off
4. Programma per il calcolo delle intersezioni tra rette, parabole, elissi, circonferenze e iperbole dando i coefficienti delle curve.

Secondo livello:

1. Editor HTML (non WYSIWYG) simile a composer per HTML 1.0
   (http://www-fog.bio.unipd.it/corso_html/comandi.html)
2. Gioco dell'Othello per utenti umani con controllo automatico della validità della mossa.
3. Calcolatrice elementare per matrici: si necessiterà di gestire l'inserimento delle dimensioni delle matrici operandi, la compatibilità degli operandi
4. Giocatore automatico stupido (sceglie una carta in base a criteri banali ma non casuali) di briscola