Problema 1 La seguente funzione intende verificare se, dati un pixel e una lista di pixel, il colore di quel pixel è uguale alla somma dei colori di due altri pixel presenti nella stessa lista, utilizzando a tale scopo una funzione ausiliaria `sum()` (Nota: un colore c1 è somma di due colori c2 e c3 se il valore numerico di ogni componente nella codifica RGB di c1 è pari alla somma dei valori numerici delle corrispondenti componenti in c2 e c3). La funzione contiene delle parti non definite, indicate dalla presenza di "...". Specificare esplicitamente le parti mancanti, in modo da ottenere un codice eseguibile e funzionalmente corretto. Nota: ogni "..." potrebbe indicare la mancanza di uno o più elementi di codice. (**Attenzione**: se/dove appropriato, indicare l'indentazione delle parti di codice che vengono introdotte).
Scrivere le risposte usando il seguente formato:

riga num. *x*: *`codice mancante`* (allineata con riga num. *y*)

```
def exSum(px, pxlsL):
# ...
# @param pxlsL: list of Pixel
# @return bool
  for i in range(0, len(pxlsL)-1):
    for ... in range(i+1, len(pxlsL)):
       if sum(px, pxlsL[...], pxlsL[j]):
          return ...
  return ...

def sum(p1, p2, p3):
# @param p1: Pixel
# @param p2: Pixel
# @param p3: Pixel
# @return bool
  return getRed(p1) == getRed(p2)+getRed(p3) and
         getGreen(p1) == getGreen(p2)+getGreen(p3)  and
         getBlue(p1) == getBlue(p2)+getBlue(p3)
```

Problema 2 La seguente funzione, date due immagini A e B, verifica se ogni colore presente in A è pari alla somma di due colori presenti in B, utilizzando allo scopo la funzione `exSum()` definita nel problema 1. (Nota: un colore c1 è somma di due colori c2 e c3 se il valore numerico di ogni componente nella codifica RGB di c1 è pari alla somma dei valori numerici delle corrispondenti componenti in c2 e c3).
La funzione contiene delle parti non definite, indicate dalla presenza di "...". Specificare esplicitamente le parti mancanti, in modo da ottenere un codice eseguibile e funzionalmente corretto. Nota: ogni "..." potrebbe indicare la mancanza di uno o più elementi di codice. (**Attenzione**: se/dove appropriato, indicare l'indentazione delle parti di codice che vengono introdotte).
Scrivere le risposte usando il seguente formato:

riga num. *x*: *`codice mancante`* (allineata con riga num. *y*)

```
def check(A, B):
# @param A: picture
# @param B: picture
# @return bool
  pxls_1 = getPixels(A)
```

```
...
for ... in pxls_1 :
   ... exSum(px, pxls_2):
      return ...
return ...
```

Problema 3 Scrivere una funzione Python (più eventuali funzioni ausiliarie) che, data un'immagine A, verifica se A contiene almeno un pixel che abbia il valore della componente `Red` uguale alla somma dei valori delle componenti `Blue` e `Green` di un qualche altro pixel di A.

Problema 4 Scrivere una funzione Python (più eventuali funzioni ausiliarie) che, data un'immagine A, verifica se ogni pixel di A abbia il valore della componente `Red` uguale alla somma dei valori delle componenti `Blue` e `Green` di un qualche altro pixel di A.

Problema 5 Date le seguenti definizioni di funzioni e variabili, determinare: (i) i loro spazi dei nomi (*namespace*) locali e quello globale, e (ii) il risultato visualizzato sullo schermo se le funzioni vengono attivate a partire dalla funzione `start()`. Rispondere a (i) e (ii) considerando i seguenti due casi: (*a*) il comando (in grigio nel testo) "`global b`" è effettivamente presente nella funzione `Q()`; (*b*) il comando "`global b`" non è presente nella funzione `Q()`.

```
b = 14
def Q() :
  global b
  a = 6
  b = 12
```

```
def R(x):
     x = 8
     Q()
     print x
```

```
def start():
     a = 14
     R(b)
     print b
```

Problema 6 Date le seguenti definizioni di funzioni e variabili, determinare: (i) i loro spazi dei nomi (*namespace*) locali e quello globale, e (ii) il risultato visualizzato sullo schermo se le funzioni vengono attivate a partire dalla funzione `start()`. Rispondere a (i) e (ii) considerando i seguenti due casi: (*a*) il comando (in grigio nel testo) "`global a`" è effettivamente presente nella funzione `K()`; (*b*) il comando "`global a`" non è presente nella funzione `K()`.

```
a = 0.5

def K():
  global a
  a = 6.5

def F(x):
  x = 7
  K()
  print a+x
```

```
def start():
 F(a)
 print a
```

Problema 7 Scrivere una funzione Python che, data una stringa `s` e una lista `L` di interi (non negativi e senza ripetizioni) restituisce una nuova stringa formata (in ordine qualunque) dai soli caratteri di `s` la cui posizione in `s` corrisponde a un valore presente in `L` (Esempio: se `s="alfeo"`

e `L=[2, 8, 0]`, risposte corrette sono le stringhe `"af"` oppure `"fa"`, perché `'f'` si trova in posizione 2 e `'a'` si trova in posizione 0).

Problema 8 Scrivere una funzione Python che, data una stringa `s`, restituisce una nuova stringa formata da tanti caratteri `*` quanti sono i caratteri alfabetici (latini) presenti in `s`. Nota: in Python, data una qualunque stringa `x`, l'operatore `x.isalpha()` del tipo di dato `str` restituisce `True` se la stringa `x` è formata solo da caratteri alfabetici (latini), `False` altrimenti (Esempio: se `s="@lf35eo4$"`, la funzione restituisce la stringa `"****"`).

Problema 9 Scrivere una funzione Python che, data un'immagine A e un intero c, restituisce la somma delle componenti Green dei colori presenti nella colonna numero c.

Problema 10 Spiegare cosa si intende in generale per *dominio* di un tipo di dato, e poi dare una definizione precisa del dominio del tipo di dato `str` (*string*) in Python.

Problema 11 Scrivere una funzione Python che, data una stringa `s`, restituisce un valore intero uguale al numero di caratteri di `s` che non sono cifre numeriche.
(Esempio: se `s="lf35eo4"`, la funzione restituisce il valore `4`).
Nota: in Python, data una qualunque stringa `x`, l'operatore `x.isdigit()` del tipo di dato `str` restituisce `True` se la stringa `x` è formata solo da cifre numeriche (da 0 a 9).