```python
from random import randint

feld=[]             # Feld wird angelegt

# rekursive Suche auf dem Feld feld
# feld ist dabei eine globale Variable
def suche_rekursiv(suchwert, anfang, ende):
    mitte = (anfang + ende) // 2        # mitte ist die Stelle, deswegen Ganzzahldivision
    if feld[mitte] == suchwert:         # bedeutet gefunden = Abbruch
        ergebnis = mitte
    elif ende < anfang:                 # Feldgrenzen sind falsch (herum)
        ergebnis = -1                   # -1 ist marker für: keine Stelle gefunden = Abbruch
    elif feld[mitte] < suchwert:        # der Suchewert ist auf der rechten Seite
        ergebnis = suche_rekursiv(suchwert, mitte+1, ende) # rekursiv weitersuchen
    else:   #   feld[mitte] > suchwert:   der suchwert ist auf der linken Seite
        ergebnis = suche_rekursiv(suchwert, anfang, mitte-1) # rekursiv weitersuchen
    return ergebnis

# Elemente würfeln und ans Feld hängen
for i in range(120):
    feld.append(randint(1,100))

# Feld sortieren
feld.sort()

# das osrtierte Feld ausgeben
for element in feld:
    print(element, end = ' ')
print('\n')

suchwert = 10

# aufruf der rekursiven Suche
fundstelle = suche_rekursiv(suchwert, 0, len(feld)-1)

# Ausgabe der Stelle (wenn gefunden)
if fundstelle == -1:
    print('Der Suchwert ', suchwert, ' wurde nicht im Feld gefunden.')
else:
    print('Fundstelle fuer den Suchwert ', suchwert, ' ist die Stelle ', fundstelle, '.')

"""
Aufgabe 5:

im Screenshot auf der Webseite ist die 10 die 15. Zahl. Da als Fundstelle aber
die 14 angegeben ist, muss die Zählung bei 0 beginnen.
"""

"""
Aufgabe 6:

beim ersten Durchlauf ist die Mitte das 60. Feld
beim 2. Durchlauf ist die Mitte das 30. Feld
beim 3. Durchlauf ist die Mitte das 15. Feld  (Feldnummer 14)

Das bedeutet, das diese Suche nur 3 Vergleiche brauchte, um die Stelle zu finden.
"""

"""
Aufgabe 7:

die binäre Suche hat die Zeitkomplexität O(log(n))

Begründung:
beim 1. Durchlauf wird die Mitte bei 120 Elementen getestet.
beim 2. Durchlauf wird die Mitte bei 60 Elementen getestet.
beim 3. Durchlauf wird die Mitte bei 30 Elementen getestet.
beim 4. Durchlauf wird die Mitte bei 15 Elementen getestet.
beim 5. Durchlauf wird die Mitte bei 7/8 Elementen getestet.
beim 6. Durchlauf wird die Mitte bei 3/4 Elementen getestet.
beim 7. Durchlauf wird die Mitte bei 1/2 Elementen getestet.
beim 8. Durchlauf wird die Zahl gefunden oder es steht fest, dass die nicht da ist.
```

```
"""

"""
Aufgabe 8:

schon bei meinem 2. Test (siehe "bin_suche-aufgabe1+8.png")
ist die Fundstelle 14 (also der 15. Wert des Feldes die zweite 10.
Bedeutet: es wird keineswegs immer die erste Stelle des auftretesn gefunden.
"""

"""
Aufgabe 9:

  n1 = 120        t1 = 35 ms
  n2 = 500.000    t2 = ???

  n2 = 4000 * n1   also n2 ist das 4000-fache von n2
                   dann ist t2 das log(4000)-fache von t1

  NR: log(4000) = 12, denn 2^12 ist (rund) 4000  (exakt 4096)

  =>  t2 = log 4000 * t1 = 12 * 35 ms = 420 ms
 """
```