Сортирање и претраживање

Задатак 001: Написати програм који користи дату листу [100, 200, 300] као стек. Уклонити два елемента са стека па додати нов елемент (400) на стек. Приказати садржај стека.

```
def main():
    stek = [100, 200, 300]
    prikaz_steka(stek)
    skini_element(stek)
    prikaz_steka(stek)
    skini_element(stek)
    prikaz_steka(stek)
    dodaj_element(stek, 400)
    prikaz_steka(stek)

#stek je prazan ako je njegova duzina 0
def jeste_prazan(s):
    return len(s) == 0

#funkcija dodaje element u stek
def dodaj_element(s, element):
    print("Dodajem element", str(element), "u stek.")
    s.append(element)

#funkcija sklanja poslednje uneti element sa steka
def skini_element(s):
    print("Uklanjam poslednji element u steku.")
    if (jeste_prazan(s) != True):
        return str(s.pop())
    else:
        return "Stek je prazan."

def prikaz_steka(s):
    print("Stek trenutno izgleda ovako:", s)
    print()

main()
```

Задаци за самосталан рад:

1) Написати програм који користи празну листу као стек. Унети елементе 100 и 200 у стек. Приказати садржај стека.

2) Написати програм који користи дату листу [100, 200, 300] као стек. Уклонити све елементе из стека. Вратити елементе у стек обрнутим редоследом.

Задатак 002: Написати програм који користи ред као листа са елементима [‘prvi’, ‘drugi’, ‘treci’], убаци елемент ‘nulti’ као први елемент у реду, а затим се последњи елемент из реда избаци из реда. Приказати садржај реда.

```
def main():
    red = ['prvi', 'drugi', 'treci']
    red = unos_elementa(red, 'nulti')
    prikaz_reda(red)
    indeks = len(red) - 1
    red = odstranjivanje_elementa(red, indeks)
    prikaz_reda(red)

def odstranjivanje_elementa(r, i):
    r.pop(i)
    return r

def unos_elementa(r, element):
    r.insert(0, element)
```

```
    return r

def prikaz_reda(r):
    print(r)

main()
```

Задатак 003: Написати програм који користи ред као имплементацију класе collections.deque. Унети три целобројне вредности у ред, одстранити прву и последњу из реда. Приказати садржај реда.

```
from collections import deque
q = deque()
q.append(3)
q.append(9)
q.append(1)
print(q)
q.pop()
print(q)
q.popleft()
print(q)
```

Задаци за самосталан рад:

3) Написати програм који користи празан ред као листу. Унети у ред једноцифрени број, његов квадрат и његов куб. Одстранити из реда све осим последње унетог елемента.

4) Написати програм који користи ред као листу [100, 200, 300] да би се на крају добио следећи ред [0, 1, 100].

Задатак 004: Написати програм који користи секвенцијално претраживање у несортираној листи [1, 4, 5, 0] да би се пронашао број 0 у листи. Приказати одговарајућу поруку о проналажењу и позицији елемента 0 у листи.

```
def main():
    lista = [1, 4, 5, 0]
    prikaz_liste(lista)
    element = unos_elementa_za_pretragu()
    pronadjen, pozicija = pretraga(lista, element)
    prikaz_rezultata(lista, element, pronadjen, pozicija)

def prikaz_liste(A):
    print(A)

def unos_elementa_za_pretragu():
    a = int(input("Koji element se trazi u listi? "))
    return a

def pretraga(A, a):
    for x in range(len(A)):
        if A[x] == a:
            return True, x
    return False, None

def prikaz_rezultata(A, a, ima, poz):
    print("U listi ", str(A))
    print("trazen je element ", a)
    if ima:
        print("Trazeni element se nalazi u listi.")
        print("Element je pronadjen na poziciji", str(poz))
    else:
        print("Trazeni element se ne nalazi u listi.")

main()
```

Задатак 005: Написати програм који користи секвенцијално претраживање у несортираној листи [0, 4, 0, 0] да би се детектовало колико има елемената 0 у листи. Приказати одговарајућу поруку после проналажења свих елемента 0 у листи.

```
def main():
    lista = [0, 4, 0, 0]
    prikaz_liste(lista)
    element = unos_elementa_za_pretragu()
    broj_pojavljivanja = pretraga(lista, element)
    prikaz_rezultata(lista, element, broj_pojavljivanja)

def prikaz_liste(A):
    print(A)

def unos_elementa_za_pretragu():
    a = int(input("Koji element se trazi u listi? "))
    return a

def pretraga(A, a):
    b = 0
    for x in range(len(A)):
        if A[x] == a:
            b += 1
    return b

def prikaz_rezultata(A, a, poj):
    print("U listi ", str(A))
    print("trazen je element ", a)
    if poj > 0:
        print("Trazeni element se nalazi u listi.")
        print("Broj pozicija na kojima je pronadjen:", str(poj))
    else:
        print("Trazeni element se ne nalazi u listi.")

main()
```

Задаци за самосталан рад:

5) Написати програм који користи секвенцијално претраживање у сортираној листи [1, 2, 4, 5] да би се пронашао број 4 у листи. Приказати одговарајућу поруку о проналажењу и позицији елемента 4 у листи.

6) Написати програм који користи секвенцијално претраживање у несортираној листи [1, 4, 5, 0] да би се пронашао најмањи број у листи. Приказати одговарајућу поруку о вредности и позицији најмањег броја у листи.

Задатак 006: Написати програм који користи бинарно претраживање у сортираној листи [0, 1, 4, 15, 23, 99] да би се детектовало да ли постоји елемент 23 у листи. Приказати одговарајућу поруку о проналажењу и позицији елемента 23 у листи.

```
def main():
    lista = [0, 1, 4, 15, 23, 99]
    prikaz_liste(lista)
    pronadjen, pozicija = binarna_pretraga(lista)
    prikaz_rezultata(lista, pronadjen, pozicija)

def prikaz_liste(A):
    print(A)

def binarna_pretraga(A):
    nizi = 0
```

```
    visi = len(A) - 1
    while nizi <= visi:
        sredina = (visi + nizi) // 2
        if A[sredina] == 23:
            return True, sredina
        elif 23 < A[sredina]:
            visi = sredina - 1
        else:
            nizi = sredina + 1
    return False, None

def prikaz_rezultata(A, ima, poz):
    print("U listi ", str(A))
    print("trazen je broj ", 23)
    if ima:
        print("Broj 23 se nalazi u listi.")
        print("Broj 23 se nalazi na poziciji", poz)
    else:
        print("Broj 23 se ne nalazi u listi.")

main()
```

Задатак 007: Написати програм који користи методу избора да би сортирао од најмањег ка највећем, целе бројеве унутар листе [55, -34, 5, 0, 99, 23] .

```
def main():
    lista = [55, -34, 5, 0, 99, 23]
    prikaz_liste(lista)
    lista = sortiranje(lista)
    prikaz_liste(lista)

def prikaz_liste(A):
    print(A)

def sortiranje(A):
    n = len(A)
    for i in range (0, n - 1):
        for j in range (i + 1, n):
            if (A[i] > A[j]):
                b = A[i]
                A[i] = A[j]
                A[j] = b
    return A

main()
```

Задатак 008: Написати програм који користи методу избора да би сортирао од највећег ка најмањем, целе бројеве унутар листе [55, -34, 5, 0, 99, 23] .

```
def main():
    lista = [55, -34, 5, 0, 99, 23]
    prikaz_liste(lista)
    lista = sortiranje(lista)
    prikaz_liste(lista)

def prikaz_liste(A):
    print(A)

def sortiranje(A):
```

```
    n = len(A)
    for i in range (0, n - 1):
        for j in range (i + 1, n):
            if (A[i] < A[j]):
                b = A[i]
                A[i] = A[j]
                A[j] = b
    return A

main()
```

Задаци за самосталан рад:
7) Написати програм који користи бинарно претраживање у сортираној листи да би се детектовало да ли постоји тражени елемент у листи. Приказати одговарајућу поруку о проналажењу и позицији траженог елемента у листи.
8) Написати програм који користи методу избора да би сортирао од најмањег ка највећем, целе бројеве унутар листе.

Задатак 009: Написати програм који користи методу merge sort да би сортирао од најмањег ка највећем, целе бројеве унутар листе [38, 27, 43, 3, 9, 82, 10].

```
def merge_sort(a):
    if len(a) > 1:
        srednji = len(a) // 2       #pronalazenje srednjeg elementa
        leva_lista = a[: srednji]
        desna_lista = a[srednji :]
        merge_sort(leva_lista)      #sortiranje leve polovine liste
        merge_sort(desna_lista)     #sortiranje desne polovine liste
        i = j = k = 0

        while i < len(leva_lista) and j < len(desna_lista):
            if leva_lista[i] < desna_lista[j]:
                a[k] = leva_lista[i]
                i += 1
            else:
                a[k] = desna_lista[j]
                j += 1
            k += 1

        #provera da li je preostao neki element u listama
        while i < len(leva_lista):
            a[k] = leva_lista[i]
            i += 1
            k += 1

        while j < len(desna_lista):
            a[k] = desna_lista[j]
            j += 1
            k += 1

def stampanje_liste(a):
    for i in range(len(a)):
        print(a[i], end=" ")
    print()

def main():
    A = [38, 27, 43, 3, 9, 82, 10]
    print ("Pocetna lista: ", end="\n")
    stampanje_liste(A)
    merge_sort(A)
```

```
    print("Sortirana lista: ", end="\n")
    stampanje_liste(A)

main()
```

Задатак 010: Написати програм који користи методу bubble-sort да би сортирао од најмањег ка највећем, целе бројеве унутар листе [38, 27, 43, 3, 9, 82, 10].

```
def bubble_sort(A):
    for x in range(len(A) - 1, 0, -1):
        for i in range(x):
            if A[i] > A[i + 1]:
                y = A[i]
                A[i] = A[i + 1]
                A[i + 1] = y

A = [38, 27, 43, 3, 9, 82, 10]
bubble_sort(A)
print(A)
```

Задаци за самосталан рад:

9) Написати програм који користи методу merge sort да би сортирао од најмањег ка највећем, целе бројеве унутар листе `[0, 1, 2, 3, 0, 2, 3]`.

10) Написати програм који користи методу bubble-sort да би сортирао од највећег ка најмањем, целе бројеве унутар листе користећи функције за решавање појединачних корака алгоритма.