1. Ի՞նչ է փաթեթավորումը և փաթեթաթափումը Python-ում և ինչպե՞ս է այն աշխատում: (1 միավոր)
Պատասխան:
Python-ում փաթեթավորումը (packing) և փաթեթաթափումը (unpacking) օգտագործվում են տվյալների հաջորդականությունները ավելի հեշտությամբ վերագրելու և ստանալու համար:
Փաթեթավորում:
- Տարրերը դրվում են փակագծերի մեջ, ստեղծելով տուփլ կամ ցուցակ:
- Այս տուփլը կարող է վերագրվել մեկ փոփոխականի:

Օրինակ:
```
my_tuple = (1, 2, 3)
```

Փաթեթաթափում:
- Տուփլի կամ ցուցակի տարրերը կարող են վերագրվել մի քանի փոփոխականների:

Օրինակ:
```
x, y, z = my_tuple
```

2. Ինչպե՞ս կարելի է մեկ տվյալների տիպը վերածել մյուսի Python-ում: Տրամադրեք օրինակ: (1 միավոր)
Պատասխան:
Python-ում տվյալների տիպերի փոխակերպումը կատարվում է տարբեր ֆունկցիաների միջոցով:
- **int():** վերածում է թվային տող կամ լողացող թվի ամբող թվի:
- **float():** վերածում է թվային տող կամ ամբող թվի լողացող թվի:
- **str():** վերածում է ցանկացած տիպի օբյեկտը տողի:

Օրինակ:
```
x = "10"
y = 3.14

x_int = int(x)  # x_int is now an integer 10
y_int = int(y)  # y_int is now an integer 3

print(type(x_int))  # Output: <class 'int'>
print(type(y_int))  # Output: <class 'int'>
```

3. Ի՞նչ տիպի արժեքներ կարելի է պահել հավաքածուում և բառարանում: (1 միավոր)
Պատասխան:
- **Հավաքածուներ (sets):** Հավաքածուներում կարելի է պահել ոչ փոփոխական տիպերի արժեքներ, ինչպիսիք են թվերը, տողերը և այլն: Հավաքածուները չեն կարող պարունակել կրկնվող տարրեր:
- **Բառարաններ (dictionaries):** Բառարաններում կարելի է պահել բանալի-արժեք զույգեր: Բանալիները պետք է լինեն ոչ փոփոխական տիպի, իսկ արժեքները կարող են լինել ցանկացած տիպի:

4. Բացատրեք, թե ինչպես են ցուցակները տարբերվում տուփլերից Python-ում: (1 միավոր)
Պատասխան:
- **Ցուցակներ (lists):** Ցուցակները փոփոխական են, այսինքն, դրանց տարրերը

կարող են փոխվել կամ նոր տարրեր կարող են ավելացվել:

- **Տուփլեր (tuples):** Տուփլերը ոչ փոփոխական են, այսինքն, դրանց տարրերը չեն կարող փոխվել կամ նոր տարրեր չեն կարող ավելացվել:

5. Օգտագործողից ստացեք տողային մուտք և տպեք տողում բառերի քանակը, ենթադրելով, որ բառերը բաժանված են տարածությամբ: (4 միավոր)

```
text = input("Enter a string: ")
words = text.split()
print("Number of words:", len(words))
```

6. Գրեք Python ֆունկցիա, որը վերադարձնում է True, եթե հնարավոր է ստեղծել եռանկյունի տրված դրական ամբող թվերով a, b, c, և False հակառակ դեպքում: (4 միավոր)

```
def is_triangle(a, b, c):
    return a + b > c and a + c > b and b + c > a
```

7. Գրեք կարճ Python ֆունկցիա, minmax(data), որը վերցնում է մեկ կամ ավելի ոչ բացասական թվերի հաջորդականություն և վերադարձնում է ամենամեծ թիվը, որը բազմապատիկ է ամենափոքր թվի: Մի օգտագործեք ներկառուցված min կամ max ֆունկցիաները ձեր լուծումը իրականացնելիս: (Դիցուք, որ հաջորդականության բոլոր տարրերը մեծ են 1-ից:) (8 միավոր)

```
def minmax(data):
    smallest = data[0]
    largest = data[0]
    for num in data[1:]:
        if num < smallest:
            smallest = num
        if num > largest:
            largest = num
    return largest // smallest * smallest
```