

Hawlaha liiska Python

Computing

Unit Barnaamijyada Python ee xogta taxan



Natiijo

Waxaan ku dari karaa oo ka saari karaa walxaha oo aan kala saari karaa xogta ku jirta liiska.

Erayada muhiimka ah

ku dar (append)

ku darista qaab dhismeedka xogta jira

ka saar (remove)

ka qaadista shay ka mid ah qaab dhismeedka xogta jira

habka liiska (list method)

shaqo ku dhex jirta oo ku shaqaynaysa liiska

kala saar (sort)

habaynta xogta si macno leh

Hawlaha liiska Python



Isticmaal hababka si aad u beddesho waxa ku jira liiska



Kala saar walxaha liiska adigoo isticmaalaya kala saar (sort)

Maxaa la soo bandhigi doonaa marka barnaamijkan la shubo?

```
1 planets = ["Mercury", "Venus",  
2 "Earth", "Mars",  
3 "Jupiter", "Saturn",  
4 "Uranus", "Neptune"]  
5 position = planets.index("Venus")  
6 print(position)
```

planets

0	"Mercury"
1	"Venus"
2	"Earth"
3	"Mars"
4	"Jupiter"
5	"Saturn"
6	"Uranus"
7	"Neptune"

The program will print 1, because this is the index position that holds the item "Venus". Remember, in Python, list indexing starts at 0.

Liisku waa qaab xog ah oo loo isticmaali karo in lagu hayo waxyaabo badan oo xog ah.

Once a list is created, you may want to change the list by adding, **removing** or updating the items held in the list.

In Python, a list is a dynamic data structure, which means that it can change size as elements are **appended** or **removed**.

Waxaa jira tiro habab oo liis ah oo loo isticmaali karo in lagu beddelo waxa ku jira liiska iyo in lagu sameeyo hawlo.

list method	explanation	example
list.append(item)	ku dar shay dhamaadka liiska	<code>planets.append("Neptune")</code>
list.insert(index, item)	ku dar shay meel go'an oo index ah	<code>planets.insert(3, "Mars")</code>
list.pop(index)	ka saar shay meel go'an oo index ah	<code>planets.pop(2)</code>
list.remove(item)	ka saar shay gaar ah	<code>planets.remove("Venus")</code>
list.index(item)	raadi index-ka shayga	<code>planets.index("Saturn")</code>
list.count(item)	tiri inta jeer ee shayga soo baxo	<code>planets.count("Earth")</code>

Waa maxay farqiga u dhexeeya `list.append()` iyo `list.insert()`? Miyayna labadooduba si fudud ugu darayaan shay cusub?



Nuur

Labaduba waxay ku darayaan shay liiska, laakiin `list.append()` wuxuu kaliya ku dari karaa shay dhamaadka liiska, halka `list.insert()` uu kuu ogolaanayo inaad sheegto halka liiska shayga lagu daro.



Saciid

Habee liiska (list method) kee ayaad isticmaali lahayd si aad ugu darto shay (item) dhamaadka liiska (list)?

a

`list.append()`



b

`list.insert()`

c

`list.add()`

Maxaa dhacaya haddii aan doonayo in aan ka saaro shayga (item) ugu dambeeya liiska (list), laakiin aanan ogayn inta uu liisku (list) dheer yahay ama qiimaha (value) ku jira booska ugu dambeeya?



Nuur

Waxaad isticmaali kartaa `list.pop()` oo aadan cayimin lambarka tusaha (index number). Tani waxay si toos ah uga saari doontaa shayga (item) ugu dambeeya liiska (list). Tusaale ahaan, `planets.pop()`.



Saciid

Habee liiska (list method) kee ayaad isticmaali lahayd si aad uga saarto shayga (item) ugu dambeeya liiska (list) haddii aadan ogayn inta uu liisku (list) dheer yahay?

a

`list.pop(4)`

b

`list.pop()`



c

`list.remove(item)`

- 1) Fur barnaamijka bilowga ah `oak.link/planet-lists`
- 2) Dhammeystir sadarka 6 si 'Uranus' loogu daro dhamaadka liiska (list) meerayaasha (planets).
- 3) Tijaabi barnaamijka si aad u hubiso in Uranus ku jiro liiska (list) meerayaasha (planets) 'oo la yaqaan 1930'.
- 4) Dhammeystir sadarka 7 iyo 8 si 'Neptune' iyo 'Pluto' loogu daro dhamaadka liiska (list) meerayaasha (planets).

Soo saarista qoraalka (text output)

```
Meerayaasha la yaqaan 1930: ['Mercury', 'Venus', 'Earth', 'Mars',  
'Jupiter', 'Saturn', 'Uranus', 'Neptune', 'Pluto']
```

- 5) Sannadkii 2006, Pluto waa laga saaray liiska meeraha. Dhammaystir sadarka 11 si aad uga saarto Pluto liiska meeraha.
- 5) Ku shub barnaamijka si aad u hubiso in Pluto uusan ku jirin liiska meeraha 'wixii ka dambeeyay 2006'.

Soo saarista qoraalka (Text output)

```
Meeraha wixii ka dambeeyay 2006: ['Mercury', 'Venus', 'Earth',  
'Mars', 'Jupiter', 'Saturn', 'Uranus', 'Neptune']
```

oak.link/planet-lists-solution

```
1 planets = ["Mercury", "Venus",  
2 "Earth", "Mars",  
3 "Jupiter", "Saturn"]  
4 print("Meeraha qadiimiga ah:")  
5 print(planets)  
6 planets.append("Uranus") # Ku dar Uranus liiska  
7 planets.append("Neptune") # Ku dar Neptune liiska  
8 planets.append("Pluto") # Ku dar Pluto liiska  
9 print("Meeraha la yaqaan 1930:")  
10 print(planets)  
11 planets.remove("Pluto") # Ka saar Pluto liiska  
12 print("Meeraha wixii ka dambeeyay 2006:")  
13 print(planets)
```

Hawlgallada liiska Python (Python list operations)



Isticmaal habab (methods) si aad u beddesho waxa ku jira liiska (list)



U kala horreey waxyaabaha ku jira liiska (list) adigoo isticmaalaya kala-sooc (sort)

Ku kala hagaaji shayga liiska adigoo isticmaalaya kala-saar (sort)



Liisaska aad ilaa hadda aragtay kuma jiraan nidaam gaar ah.

Mararka qaarkood waa faa'iido in la kala saaro (sort) shayga ku jira liiska si loogu isticmaalo barnaamijka.

Tusaale ahaan, waxaa laga yaabaa inaad haysato liiska buundooyinka (scores) ciyaar oo aad rabto inaad u kala saarto (sort) si aad u helato buundada (score) ugu sarreysa.

index	0	1	2	3	4
qiime (value)	5	2	10	7	4

Waxaan isticmaali karaa kala-saar (sort) si aan u dhigo liiska magacyada sida alifbeetada (alphabetically).



Nuur

Waxaan isticmaali karaa kala-saar (sort) si aan u dhigo buundooyinka (scores) ciyaartayda sida kan ugu sarreeya una dhaadheer, ka dibna aan ka saaro buundada (score) ugu hooseeya liiska anigoo isticmaalaya `score.pop()`.



Saciid

Waxaa jira laba hawlgallo liis (list operations) oo aad isticmaali karto si aad u kala saarto (sort) qiimayaasha (values) ku jira liiska:

`list.sort()` waxaa loo isticmaali karaa in lagu kala saaro (sort) liiska sida kan ugu hooseeya una sarreeya (ascending).

index	0	1	2	3	4
scores	5	2	10	7	4
scores.sort()	2	4	5	7	10

Haddii liisku ka kooban yahay qoraal (text) halkii tirooyin (numbers), liiska waxaa loo kala saari lahaa (sorted) sida alifbeetada (alphabetically) A ilaa Z.

Ku kala hagaaji shayga liiska adigoo isticmaalaya kala-saar (sort)

`list.sort(reverse=True)` waxaa loo isticmaali karaa in lagu kala saaro (sort) liiska sida kan ugu sarreeya una hooseeya (descending).

index

buundooyin (scores)

`scores.sort(reverse=True)`

0	1	2	3	4
5	2	10	7	4
10	7	5	4	2

Haddii liisku ka koobnaan lahaa qoraal halkii tiro, liiska waxaa loo kala hor marin lahaa xarfaha Z ilaa A.

Ku kala horreey (order) shayyada liiska adigoo isticmaalaya kala-horayn (sort)



Waa maxay hawlgalka liiska (list operation) ee loo isticmaalay si loo kala horreeyo liiskan?

a

`scores.sort()`



b

`scores.sort(reverse=True)`

c

`scores.append()`

Qoraal soo-saar (Text output)

Buundooyin (Scores): [2,5,7,8,10]

Ku kala horreey (order) shayyada liiska adigoo isticmaalaya kala-horayn (sort)



Waa maxay hawlgalka liiska (list operation) ee loo isticmaalay si loo kala horreeyo liiskan?

a

`scores.sort()`

b

`scores.sort(reverse=True)`

c

`scores.append()`

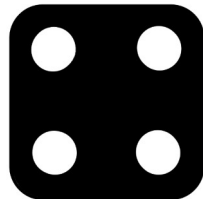
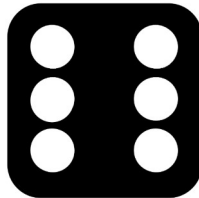


Qoraal soo-saar (Text output)

Buundooyin (Scores): [10,8,7,5,2]

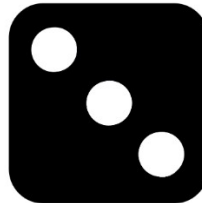
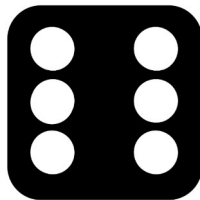
Hawl-galkaan (activity), waxaad samayn doontaa ciyaar laba-ciyaaryood oo dhagaxyo (dice) ah.

- Hal ciyaaryood waa weeraryahan (attacker), kan kalena waa difaacayaal (defender).
- Weeraryuhu wuxuu rogaa saddex dhagax (dice). Difaacuhu wuxuu rogaa laba dhagax.
- Dhagaxyada (dice) ee ciyaaryahan kasta waxaa loo kala horreeyaa si hoos-u-dhac ah (descending order) ugu sarreeya hore).
- Rididda ugu sarreysa ee weerarka (attacker) waxaa lala barbardhigayaa rididda ugu sarreysa ee difaaca (defender). Ciyaaryahanka leh rididda ugu hooseysa wuxuu lumiyaa hal dhibic (point). Haddii labada rididood ay isle'eg yihiin, weerarka (attacker) ayaa lumiya dhibicda (point).
- Kadib, rididda labaad ee ugu sarreysa ee weerarka (attacker) waxaa lala barbardhigayaa rididda labaad ee ugu sarreysa ee difaaca (defender). Ciyaaryahanka leh midka ugu hooseeya ee labada wuxuu lumiyaa hal dhibic (point). Haddii labada rididood ay isle'eg yihiin, weerarka (attacker) ayaa lumiya dhibicda (point).



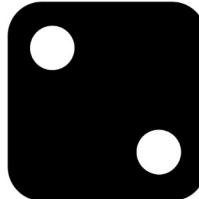
Tusaale ahaan:

weerarka (attacker):



Rididdo (Rolls) ayaa loo
kala-soocay (sorted)
habka hoos u dhaca
(descending order)

difaaca (defender):

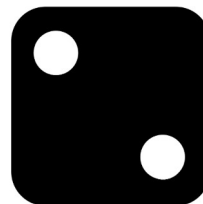
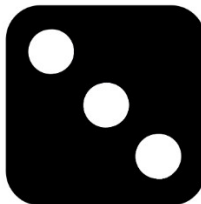
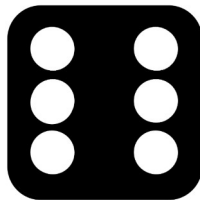


difaaca (defender) -1
dhibic (point)

difaaca (defender) -1
dhibic (point)

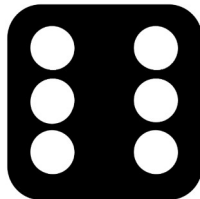
Tusaale ahaan:

weerarka (attacker):



Rididdo (Rolls) ayaa loo
kala-soocay (sorted)
habka hoos u dhaca
(descending order)

difaaca (defender):



weerarka (attacker)
-1 dhibic (point)

weerarka (attacker)
-1 dhibic (point)

- 1) Fur barnaamijka bilowga (starter program) oak.link/dice-roll-game
- 2) Dhammaystir sadarka 9 (line 9) si liiska (list) weerarka (attacker), oo ka kooban rididda (dice rolls) weerarka (attacker), loo kala-sooco (sorted).
- 3) Dhammaystir sadarka 10 (line 10) si liiska (list) difaaca (defender), oo ka kooban rididda (dice rolls) difaaca (defender), loo kala-sooco (sorted).
- 4) Or barnaamijka (program) oo hubi in liisaska (lists) si sax ah loo daabacay (printed).

Qoraal soo-saar (Text output)

```
Rididda (Rolls) ciyaaryahanka (Players)
Weerarka: [6, 3, 2]
Difaaaca: [5, 3]
Kala-saar (Sorted)
Weerarka: [6, 3, 2]
Difaaaca: [5, 3]
```

- 5) Ku dar baaq (if statement) barnaamijkaaga oo isbarbardhigaya duubka sare ee liiska weerarka iyo kan difaaca.
 - a) Haddii weerarka guuleysto, hoos u dhig defender_points 1.
 - b) Haddii difaacu guuleysto, hoos u dhig attacker_points 1.
 - c) Xusuusnow in haddii duubyada sare ay isle'eg yihiin, difaacu guuleysto.
- 5) Ku dar baaq (if statement) kale barnaamijkaaga si aad u isbarbardhigto duubyada labaad ee ugu sareeya.

```
1 from dice import dicerolls
2 attacker_points = 0
3 defender_points = 0
4 attacker = dicerolls(3)
5 defender = dicerolls(2)
6 print("Duubyada ciyaartoyda")
7 print(f"Weerarka: {attacker}")
8 print(f"Difaaaca: {defender}")
9 attacker.sort(reverse=True)
10 defender.sort(reverse=True)
11 print("Sorted")
12 print(f"Attacker: {attacker}")
13 print(f"Defender: {defender}")
14 haddii attacker[0] > defender[0]:
15     defender_points = defender_points - 1
16 haddii kale:
17     attacker_points = attacker_points - 1
18 haddii attacker[1] > defender[1]:
19     defender_points = defender_points - 1
20 haddii kale:
21     attacker_points = attacker_points - 1
22 print(f"Attacker points: {attacker_points}")
23 print(f"Defender points: {defender_points}")
```

`attacker.sort(reverse=True)`
`defender.sort(reverse=True)`

oak.link/dice-roll-game-solution

```
1 from dice import dicerolls
2 attacker_points = 0
3 defender_points = 0
4 attacker = dicerolls(3)
5 defender = dicerolls(2)
6 print("Players' rolls")
7 print(f"Attacker: {attacker}")
8 print(f"Defender: {defender}")
9 attacker.sort(reverse=True)
10 defender.sort(reverse=True)
11 print("Sorted")
12 print(f"Attacker: {attacker}")
13 print(f"Defender: {defender}")
14 if attacker[0] > defender[0]:
15     defender_points = defender_points - 1
16 else:
17     attacker_points = attacker_points - 1
18 if attacker[1] > defender[1]:
19     defender_points = defender_points - 1
20 else:
21     attacker_points = attacker_points - 1
22 print(f"Attacker points: {attacker_points}")
23 print(f"Defender points: {defender_points}")
```

```
if attacker[0] > defender[0]:
    defender_points = defender_points - 1
else:
    attacker_points = attacker_points - 1
    haddii attacker[1] > defender[1]:
        defender_points = defender_points - 1
    haddii kale:
        attacker_points = attacker_points - 1
```

oak.link/dice-roll-game-solution

Hawlaha waxaa lagu samayn karaa liisaska iyadoo la adeegsanayo hababka liiska sida ku darista, ka saarista iyo gelinta shayga.

Ku darista shay waxay ku darsaa dhamaadka liiska, halka gelintu ay shayga dhigi doonto meel gaar ah (index).

Liis waxaa loo kala saari karaa si kor u kaca ama hoos u dhaca ah si loo fududeeyo habaynta xogta.