Operations on lists .

Source: docs.python.org/3/tutorial/datastructures.html#more-on-lists

|  |  |  |
| --- | --- | --- |
|  | **Add and remove items** | |
|  | list.append(item)  e.g. numbers.append(42) | Add an item to the end of the list. |
|  | list.insert(index, item)  e.g. cities.insert(2, "Oslo") | Insert an item at a given position. |
|  | list.pop(index)  e.g. last = values.pop() | Remove the item at the given position in the list, and return it. If no index is specified, remove and return the last item in the list. |
|  | list.remove(item)  e.g. countries.remove("Japan") | Remove the first item from the list with a particular value. Raises a ValueError if there is no such item. |
|  | **Find and count items** | |
|  | list.index(item)  e.g. pos = planets.index("Mars") | Search for the first occurrence of an item in the list and return its (zero-based) index. Raises a ValueError if there is no such item. |
|  | list.count(item)  e.g. nb\_the = words.count("the") | Return the number of times an item appears in the list. |
|  | **Modify list** | |
|  | list.reverse()  e.g. values.reverse() | Reverse the items of the list. |
|  | list.sort()  e.g. names.sort()  e.g. numbers.sort(reverse=True) | Sort the items of the list.  Use the reverse=True argument to sort in **descending order**. |

Worked example: Third rock from the sun

This program searches for the index of Earth in the list of planets.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | planets = ["Mercury", "Venus",  "Earth", "Mars",  "Jupiter", "Saturn",  "Uranus", "Neptune"]  position = **planets.index("Earth")** + 1  print("Earth is planet number", position) |

Here are the contents of the planets list, with an index next to each item:

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

The solar system

Task 1 Planets

Mercury, Venus, Mars, Jupiter, and Saturn are visible from Earth. These planets have been known since antiquity.

By 1930, astronomers had added Uranus (1781), Neptune (1846), and Pluto (1930) to the list of known planets.

A score of discoveries in the early 2000s led to the demotion of Pluto; since 2006 it is no longer considered a planet and is classified instead as a *dwarf planet*.

In this task, you will create a program that recounts this short story about our view of the planets in our solar system.

**SCROLL DOWN**

**Step 1**

**Open** the program below on Bourne to Code it is just below the button where you downloaded this worksheet and is labelled Task 1

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | planets = ["Mercury", "Venus",  "Earth", "Mars",  "Jupiter", "Saturn"]  print("Planets in antiquity:")  print(planets)  # Add Uranus to the list  # Add Neptune to the list  # Add Pluto to the list  print("Planets by 1930:")  print(planets)  # Remove Pluto from the list  print("Planets after 2006:")  print(planets) |

Challenge Steps 2 to 5

**Step 2**

**Complete** line 6, so that "Uranus" is added to the end of the list of planets.

**Tip**: Use the command **planets.append(“Uranus”)** in the correct area

**Step 3**

**Run** the program to make sure that Uranus is included in the list of planets “known by 1930”.

|  |  |
| --- | --- |
| **Example** |  |
| Note: Use this example to check your program. | |
| This should be part of the program’s output: check that the items and their order are correct. | Planets by 1930:  ['Mercury', 'Venus', 'Earth', 'Mars', 'Jupiter', 'Saturn', 'Uranus'] |

**Step 4**

**Complete** line 7 and 8, so that "Neptune" and "Pluto" are also added to the end of the list of planets.

**Tip**: We used **planets.append(“Uranus”)** to add Uranus to the end of the list, modify this code to now add “Neptune”, then add “Pluto”

**Step 5**

**Run** the program, to make sure that Neptune and Pluto are included in the list of planets “known by 1930”.

|  |  |
| --- | --- |
| **Example** |  |
| Note: Use this example to check your program. | |
| This should be part of the program’s output: check that the items and their order are correct. | Planets by 1930:  ['Mercury', 'Venus', 'Earth', 'Mars', 'Jupiter', 'Saturn', 'Uranus', 'Neptune', 'Pluto'] |

Challenge Steps 6 to 7

**Step 6**

**Complete** line 11, so that "Pluto" is removed from the list of planets.

**Tip**: To add Pluto you used the command **planets.append(“Pluto”)**.Instead of append, which word from the list below will you use to remove Pluto from the list?

1. delete
2. remove
3. pop

**Step 7**

**Run** the program, to make sure that Pluto is not included in the list of planets “after 2006”.

|  |  |
| --- | --- |
| **Example** |  |
| Note: Use this example to check your program. | |
| This should be part of the program’s output: check that the items and their order are correct. | Planets after 2006:  ['Mercury', 'Venus', 'Earth', 'Mars', 'Jupiter', 'Saturn', 'Uranus', 'Neptune'] |

Task 2 Dwarf planets

In 2006, astronomers set out rules that would classify certain objects in the solar system as dwarf planets. At present there are five such objects (but this list is subject to change): Ceres, Pluto, Haumea, Eris, and Makemake, ordered according to their date of discovery.

There was controversy around which team the discovery of Haumea should be attributed to. The name originally proposed for it was Ataecina.

In this task, you will complete a program that displays the list of dwarf planets.

**Step 1**

**Open** the program below on Bourne to Code it is just below the button where you downloaded this worksheet and is labelled Task 2

|  |  |
| --- | --- |
| 1  2  3  4  5 | dwarves = # Create list of dwarves  # Change 2nd item to Haumea  # Add Pluto as 2nd item  print("Dwarf planets:")  print(dwarves) |

Challenge Steps 2 and 3

**Step 2**

**Complete** line 1, so that the list of dwarf planets comprises Ceres, Ataecina, Eris, and Makemake, in that order.

**Tip**: A list used these brackets **[ ]** , each word must be **comma** **separated** and use speech marks **“”**. Look at the Planets list if you need guidance.

**Step 3**

**Run** the program to make sure that the list of dwarf planets is correct.

|  |  |
| --- | --- |
| **Example** |  |
| Note: Use this example to check your program. | |
| This should be part of the program’s output: check that the items and their order are correct. | Dwarf planets:  ['Ceres', 'Ataecina', 'Eris', 'Makemake'] |

Challenge Steps 4 and 5

**Step 4**

**Complete** line 2, so that the second item in the list is assigned the value of "Haumea".

**Tip**: Complete this line of code **dwarves.insert(???, "???")**, the ??? will need to be completed before the code works

**Tip**: Item numbering starts from 0, so the second item has an index of **1**.

**Step 5**

**Run** the program to make sure that the second item in the list is Haumea, instead of Ataecina.

|  |  |
| --- | --- |
| **Example** |  |
| Note: Use this example to check your program. | |
| This should be part of the program’s output: check that the items and their order are correct. | Dwarf planets:  ['Ceres', 'Haumea', 'Eris', 'Makemake'] |

Challenge Steps 6 to 7

**Step 6**

**Complete** line 3, so that "Pluto" is added to the list, as its second item.

**Tip**: Use the same code as above but this time replace “**Haumea**” with “**Pluto**”

**Step 7**

**Run** the program to make sure that Pluto is now the second item in the list, preceded by Ceres and followed by Haumea, Eris, and Makemake, in that order.

|  |  |
| --- | --- |
| **Example** |  |
| Note: Use this example to check your program. | |
| This should be part of the program’s output: check that the items and their order are correct. | Dwarf planets:  ['Ceres', 'Pluto', 'Haumea', 'Eris', 'Makemake'] |

Explorer task Add them up

**Combine** your programs for planets (Task 1) and dwarf planets (Task 2) into one.

At the end of your new program, **add** the following lines:

|  |  |
| --- | --- |
| +  +  +  + | solar = planets + dwarves  solar.sort()  print("Planet-mass objects:")  print(solar) |

Can you **predict** what the program will display, *before* actually running it?

This resource is licensed under the Open Government Licence, version 3. For more information on this licence, see [ncce.io/ogl](http://ncce.io/ogl).