List Objects

If *len* is greater than zero, the returned list object's items are
set to ``NULL``. Thus you cannot use abstract API functions such as
:c:func:`PySequence_SetItem` or expose the object to Python code before
setting all items to a real object with :c:func:`PyList_SetItem`.

Unknown directive type "c:function".

```
.. c:function:: Py_ssize_t PyList_Size(PyObject *list)

   .. index:: builtin: len

   Return the length of the list object in *list*; this is equivalent to
   ``len(list)`` on a list object.
```

Unknown directive type "c:function".

```
.. c:function:: Py_ssize_t PyList_GET_SIZE(PyObject *list)

   Macro form of :c:func:`PyList_Size` without error checking.
```

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyList_GetItem(PyObject *list, Py_ssize_t index)

   Return the object at position *index* in the list pointed to by *list*.  The
   position must be non-negative; indexing from the end of the list is not
   supported.  If *index* is out of bounds (<0 or >=len(list)),
   return ``NULL`` and set an :exc:`IndexError` exception.
```

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyList_GET_ITEM(PyObject *list, Py_ssize_t i)

   Macro form of :c:func:`PyList_GetItem` without error checking.
```

Unknown directive type "c:function".

```
.. c:function:: int PyList_SetItem(PyObject *list, Py_ssize_t index, PyObject *item)

   Set the item at index *index* in list to *item*.  Return ``0`` on success.
   If *index* is out of bounds, return ``-1`` and set an :exc:`IndexError`
   exception.

   .. note::

      This function "steals" a reference to *item* and discards a reference to
      an item already in the list at the affected position.
```

Unknown directive type "c:function".

```
.. c:function:: void PyList_SET_ITEM(PyObject *list, Py_ssize_t i, PyObject *o)
```

```
     Macro form of :c:func:`PyList_SetItem` without error checking. This is
     normally only used to fill in new lists where there is no previous content.

     .. note::

        This macro "steals" a reference to *item*, and, unlike
        :c:func:`PyList_SetItem`, does *not* discard a reference to any item that
        is being replaced; any reference in *list* at position *i* will be
        leaked.
```

Unknown directive type "c:function".

```
   .. c:function:: int PyList_Insert(PyObject *list, Py_ssize_t index, PyObject *item)

      Insert the item *item* into list *list* in front of index *index*.  Return
      ``0`` if successful; return ``-1`` and set an exception if unsuccessful.
      Analogous to ``list.insert(index, item)``.
```

Unknown directive type "c:function".

```
   .. c:function:: int PyList_Append(PyObject *list, PyObject *item)

      Append the object *item* at the end of list *list*. Return ``0`` if
      successful; return ``-1`` and set an exception if unsuccessful.  Analogous
      to ``list.append(item)``.
```

Unknown directive type "c:function".

```
   .. c:function:: PyObject* PyList_GetSlice(PyObject *list, Py_ssize_t low, Py_ssize_t high)

      Return a list of the objects in *list* containing the objects *between* *low*
      and *high*.  Return ``NULL`` and set an exception if unsuccessful.  Analogous
      to ``list[low:high]``.  Indexing from the end of the list is not supported.
```

Unknown directive type "c:function".

```
   .. c:function:: int PyList_SetSlice(PyObject *list, Py_ssize_t low, Py_ssize_t high, PyObject *itemli

      Set the slice of *list* between *low* and *high* to the contents of
      *itemlist*.  Analogous to ``list[low:high] = itemlist``. The *itemlist* may
      be ``NULL``, indicating the assignment of an empty list (slice deletion).
      Return ``0`` on success, ``-1`` on failure.  Indexing from the end of the
      list is not supported.
```

Unknown directive type "c:function".

```
   .. c:function:: int PyList_Sort(PyObject *list)

      Sort the items of *list* in place.  Return ``0`` on success, ``-1`` on
      failure.  This is equivalent to ``list.sort()``.
```

Unknown directive type "c:function".

```
.. c:function:: int PyList_Reverse(PyObject *list)

    Reverse the items of *list* in place.  Return ``0`` on success, ``-1`` on
    failure.  This is the equivalent of ``list.reverse()``.
```

Unknown directive type "c:function".

```
.. c:function:: PyObject* PyList_AsTuple(PyObject *list)

    .. index:: builtin: tuple

    Return a new tuple object containing the contents of *list*; equivalent to
    ``tuple(list)``.
```