Abstract class

<table>
<thead>
<tr>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>virtual method()</td>
</tr>
<tr>
<td>method()</td>
</tr>
</tbody>
</table>

Inheritance (is-a) relationship

```
Base

Derived2 is-a Base
```

Object

```
classname: objectname
```

Aggregation and Composition (has-a) relationship

```
Whole

Part

Whole has Part as a part; lifetimes might be different; Part might be shared with other Wholes. (aggregation)

Whole

Part

Whole has Part as a part; lifetime of Part controlled by Whole, Part objects are contained in one Whole object. (composition)
```

Association (uses, interacts-with) relationship

```
A

A's role

B

B's role

Navigability - can reach B starting from A
```

Multiplicity in Aggregation, Composition, or Association

```
* - any number
1 - exactly 1
n - exactly n
0..1 - zero or one
1..* - 1 or more
n .. m - n through m
```

```
A
1

* B

Each A is associated with any number of B’s. Each B is associated with exactly one A.
```

adapted from http://umich.edu/~eeecs381/handouts/UMLNotationSummary.pdf - 2018
Basic UML Sequence Diagram Notation

- **objects**: one that starts the action at the left

- **time flow**:
  - `<create>>`
  - `object1` (object is activated (i.e., on the stack))
  - `object2`
  - `get_info(spec)`
  - `search-self`
  - `requested info`
  - `do_something()`
  - `<<destroy>>`

- **message sent (function call)**: information returned (non-void return)

- **object sends message to itself**

- **return with no information (void return) - use for clarity only**