Exercises - 1a

- Verify that you can create an isolate with some data in it, and compute something:

```
)load isolate
myis←isolate.New ''
myis.abc←1 2 3
myis.(+/abc)
```

Exercises - 1b

- Initialise isolates from various sources:

Source	Example
Namespace	`is←isolate.New MyNS`
Workspace	`is←isolate.New 'ws name'`
Name List	`is←isolate.New 'foo' 'goo' 'data'`
Empty	`is←isolate.New ''`

- Cheat sheet for making namespaces:

```
MyNS←⎕NS '' ⋄ MyNS.data←1 2 3 4
```

Exercises – 1c

- Create a vector of isolates and distribute data across them. Compute something in parallel. For example:

```
iss←isolate.New¨ '' '' ''
iss.data←↓3 4⍴⍳12
iss.(+/data)
```

Exercises – 2a

- Experiment with making calls to ⎕DL in several isolates at once.

```
isos←isolate.New¨'' '' ''
isos.⎕DL 5 10 15
```

- Notice the difference between assigning the result to a variable vs displaying the result in the session.

```
delays←isos.⎕DL 5 10 15
```

- Verify that you can perform structural functions on the result (like shape or reshape) without blocking

Exercises – 2b

- Experiment until comfortable with the use of

```
Values  Running  Failed  Available
```

... to inspect the results of asynchronous calls. For example:

```
      isos←isolate.New¨''  ''  ''
      delays←isos.⎕DL 5 10 15
      isolate.Values 'delays'
5.093  [Null]  [Null]
```

Exercise 3a

- Define a small stand-alone function which consumes significant CPU and call it via

```
      II   ⍺ or   isolate.ll
      I⍤   ⍺ or   isolate.llEach
```

- Open Windows Task Manager and look at the CPU and timings of invoking the function with the normal each operator (⍤) versus `I⍤`

- Or cheat:

```
      ]load …\isolatework\loop
```

Exercise 3b

- Using your function from 3a, experiment with calling functions derived from `II` and `IÏ` for example:

```
foo←{⎕DL 2×⍵}
foo_asynch←foo II
future←foo_asych 3
```

- Invoke your asynchronous function twice and displays the results as they become available. (hint/answer on next page)

Exercise 3b - answer

- Invoke your asynchronous function twice and displays the results as they become available:

```
      l1←foo_asynch 2 ⋄ l2←foo_asynch 3
      {⍎⍵←l1}&0 ⋄ {⍎⍵←l2}&0
4.062
6.094
```

Session 3 Summary

- `isolate.ll` (or `II`) are models of the parallel operator ⫴
- `isolate.llEach` (or `IÏ`) of what will be ⫴¨

- The parallel operator creates an isolate which is empty except for a copy of the operand function, and invokes the function
  - The isolate is subsequently discarded

- ”Classical” Dyalog threading can be used to launch a thread which will wait on a future and use the result when it arrives

Exercise 4

Invoke your own function under `IIX.PEACH`

```
]load […where you saved it…]\IIX
NS←⎕NS ''
NS.(foo←{⎕DL 2×⍵})
('foo' IIX.PEACH NS)4 0⍴1
```

Exercise 5a

- Write a function which updates a global variable in the containing space (`##.GLOBAL`).
- Turn call-backs on:

```
isolate.Config 'listen' 1
isolate.Reset 0
```

- Invoke your function under `IÏ` and verify that it is doing what you expect.

Exercise 5b

(Optional)

- Update your isolate function to loop and make several calls to a global function outputs something to the session.
- **Beware:** Do not display the result of your long-running function, there seems to be a conflict between delaying on a future and GUI message queue processing.

Exercise 6

- Enable server-side debugging:

```
isolate.Config 'onerror' 'debug'
isolate.Reset 0
```

- Put a bug in a function and invoke it in an isolate

Configuration Options

Option Name	Default	Description
drc	#	Location of CONGA namespace to use
homeport	7051	The lowest port number that will be used
homeportmax	7151	The highest port number to try listening on
isolates	99	Number or isolates allowed per process
listen	0	1 to allow isolates to issue callbacks to parent process
maxws	'64000'	By default, uses the same setting as the current APL session
onerror	'signal'	Signal errors to the line waiting for results
processes	1	The number of processes to start per processor
processors	4	Number of processors (default determined automatically)
runtime	1	Whether to run isolates using the runtime engine
workspace	'isolate'	Workspace to load when starting new isolates