1. In the Julia REPL, the key `?` enables to access *package mode*, from where software libraries can be installed and uninstalled.
2. In the Julia REPL, the key `]` enables to access *help mode*, from where documentation can be accessed.
3. In Julia, the first piece of code below (left) produces an error, while the second (right) plots the sine function (assuming that package `Plots` is already installed).

```
import Plots
plot(cos)
```

```
using Plots
plot(sin)
```

4. Explain in words what the following function does? (Assume $n \geqslant 0$)

```
f(n) = n in (0, 1) ? 1 : n * f(n-1)
```

Explanation:

5. In Julia, the two following commands produce the same result:

```
v = cos.([1.0, 2.0, 3.0])
v = [cos(1.0), cos(2.0), cos(3.0)]
```

6. In Julia, the following code plots the function $t \mapsto \cos(t-1)$

```
import Plots
shifted_cos(t, p) = cos(t - p)
Plots.plot(t -> shifted_cos(t, 1))
```

7. In Julia, the command `+(7, 5)` returns 12, while `1 .+ [1, 2, 3]` returns the array `[2, 3, 4]`.
8. In the following piece of code, the boolean expression on the last line evaluates to `true`:

```
import Base.exp
exp(a::Vector) = exp(sum(a))
exp([1, 2, -3]) == 1
```

9. In the following piece of code, the boolean expression on the last line evaluates to `false`:

```
import Base.>
>(a::String, b::String) = length(a) > length(b)
"Good afternoon" > "world"
```

10. All the code that forms the standard library of the Julia programming language is free both as in *free beer* (the price is 0) but also as in *free speech* (you are free to use, modify, and distribute the program).
11. In a Jupyter notebook, all the cells are independent. In particular, a variable defined in one cell cannot be employed in any of the following cells.
12. In a Jupyter notebook, the output displayed below a cell comes from the last expression evaluated in that cell. If that expression returns a value (like a number, string, DataFrame, or an image/plot object), Jupyter automatically displays it.
13. Loops (`while` and `for`) are much slower in Julia than in Python, and so they should be avoided as much as possible.
14. What is the value of `s` in the following piece of code?

```
struct Wolf end; struct Dog end
meet(a::Wolf, b::Wolf) = "Two wolves meet: they howl together."
meet(a::Wolf, b::Dog) = "A wolf meets a dog: the wolf growls and the dog is scared."
meet(a::Dog, b::Wolf) = meet(b, a)
meet(a::Dog, b::Dog) = "Two dogs meet: they wag their tails."
raksha, akela = Wolf(), Wolf()
s = meet(raksha, akela)
```

Answer key

1. **False.** The key `]` gives access to *package mode*; the key `?` gives access to *help mode*.

2. **False.** The key `]` gives access to *package mode*; the key `?` gives access to *help mode*.

3. **True.** With `import Plots`, the function `plot` is not in scope (only the module `Plots` is), so the first piece of code raises an `UndefVarError`. With `using Plots`, the function `plot` is usable, so the second piece of code plots the sine function.

4. The function computes the factorial: `f(n) = n!`. Indeed, `f(0) = f(1) = 1` and `f(n) = n * f(n - 1)` for $n \geqslant 2$.

5. **True.** Broadcasting `cos.` over an array thread the function elementwise, and the two commands produce the same vector.

6. **True.** The anonymous function `t -> shifted_cos(t, 1)` fixes the second argument to 1, so the curve plotted is $t \mapsto \cos(t - 1)$.

7. **True.** `+(7, 5)` is the function-call syntax for $7 + 5 = 12$, and `1 .+ [1, 2, 3]` adds 1 to every entry, giving `[2, 3, 4]`.

8. **True.** The new method for `Base.exp` computes `exp(sum(a))`, and `sum([1, 2, -3]) = 0`, so the expression evaluates to `exp(0) == 1`, which is `true`.

9. **False.** With the given method, `>` on strings compares their lengths, and "Good afternoon" has length 14 while "world" has length 5. Since $14 > 5$, the expression evaluates to `true`, not `false`.

10. **True.** Julia and its standard library are distributed under the permissive MIT license.

11. **False.** All the cells of a Jupyter notebook share the same kernel, so variables defined in one cell remain available in the following cells.

12. **True.** Jupyter displays the value of the last expression evaluated in a cell, if it returns a value.

13. **False.** Julia is a just-in-time compiled language: loops are fast (unlike in Python) and do not need to be avoided.

14. The value of `s` is the string `"Two wolves meet: they howl together."` because both `raksha` and `akela` are `Wolf` objects, hence the most specific matching method is `meet(a::Wolf, b::Wolf)`.