

Clojure Basics

```
(str " Hel lo" " " " Wor ld")
```

Math operators +, -, /, *

```
(quote (+ 1 2)) or '(+ 1 2) or `(+ 1 2)
```

```
(eval `(+ 1 2)) => 3
```

```
(def x 1)
```

```
(defn hello [name] (str " Hello " name)
)
```

```
(defn hello2 [& args]
  (for [x args] (hello x)))
```

```
(apply <fn> <ar gs>) (apply + [1 2 3])
```

```
(if <co nd> <ex p1> <ex p2>)
```

false => false, nil

```
(case grade
  :A "Great"
  :B "Good"
  :C "OK"
  "Not good")
```

```
(cond
  (>= grade 90) "Great"
  (>= grade 80) "Good"
  :else "Need Work")
```

```
(for [x (range 10)
      :let [y (+ x 2)]
      :when (< y 5)] y)
```

```
(for [x (range 3)
      y (range 2)] [x y])
```

```
(for [x (range 10)
      :let [y (+ x 2)]
      :while (< y 5)] y)
```

Destructuring

```
(first seq) => 1 (rest seq) => (234)
```

```
(nth seq 2) => 3 (last seq) => 4
```

```
(take 2 seq) => (1 2) (drop 2 seq) => (2 3)
```

```
(take- while (fn [x] (< x 3)) seq)
=> (1 2)
```

```
(filter even? seq) => (2 4)
```

Vectors as functions

```
(nth vect 2)
```

```
(vect 2) => 3
```

```
(get vect 2)
```

```
(subvec vect 2 4)
=> [3 4]
```

Maps

```
(get my-map :key) (:k
ey my-map)
```

Nested Map

```
(get-in my-map [:outer :inn
er])
```

Nested with vector

```
(get-in my-map [:outer
n :inner])
```

```
(keys my-map)
```

```
(vals my-map)
```



By Glowie

cheatography.com/glowie/

Not published yet.

Last updated 13th December, 2024.

Page 1 of 1.

Sponsored by CrosswordCheats.com

Learn to solve cryptic crosswords!

<http://crosswordcheats.com>