

[illegible]

11

1. 0000	1
2. 1000000000	1
3. 00000	1
4. 0000000000	2
5. 0000	3
6. 00000000	5
7. 0000Red/System	5
8. 00000	6
9. 000000	8
10. 0000	9
11. 0000000000	10
12. 000000	10

1.

Red DSL  
Red 1

`Red`

Github red/red

[illegible][illegible]

## 2. 1□□□□□□□□

**Red**

### 3. ☐ ☐ ☐ ☐ ☐

[illegible]

red/red Red

Tabs: 4

□□□□□□□□□□□□□□□□□□□□□□□□

□□□□

```
func [  
    arg1  
    arg2  
    ...  
][  
    print arg1  
    ...  
]
```

□□□

```
func [  
arg1                ;-- □□□□□□□□  
arg2  
...  
][  
    print arg1      ;-- □□□□□□□□  
    ...  
]
```

## 4. □□□□□□□□□□

□□□□□□□□□□□□ □□□□ □□□□ □□□□□□□□

□□□□□□□□□□□□□□□□

a: []

□□□□□□□□□□□□□□□□

```
[][]  
[]()
```

```
[  
    hello  
][                ;-- □□□□□□□□  
    world  
]
```

[illegible]

```
array: [[ ] [ ] [ ] [ ]]
list:  [ [ ] [ ] [ ] [ ] ]

either a = 1 [ ["hello"]] [ ["world"]]
either a = 1 [ [ "hello" ] ] [ [ "world" ] ]
```

[illegible]

b: either  $a = 1$   $[a + 1][3]$

□ □

1111

```
b: either a = 1 [  
    a + 1  
][  
    123 + length? mold a  
]
```

33

```
b: either a = 1
    [a + 1][123 + length? mold a]
```

Red either

[illegible]

```
print either a = 1 ["hello"][
    append mold a "this is a very long expression"
]

while [not tail? series][
    print series/1
    series: next series
]
```

5.

111

□ □ □ □ □ □

11

Red word

#####

#####2#####

####

```
code: 123456
name: "John"
table: [2 6 8 4 3]
lost-items: []

unless tail? list [author: select list index]
```

####

```
code_for_article: 123456
Mytable: [2 6 8 4 3]
lostItems: []

unless tail? list-of-books [author-property: select list-of-books selected-index]
```

#####

#####word#####

#####

#####logic!

#####length?#

index?#####

#####

####

```
make: func [...
reduce: func [...
allow: func [...
crunch: func [...
```

####

```
length: func [...
future: func [...
position: func [...
blue-fill: func [... ;-- fill-blue#####
```

#####OSRed#####API#####API#####

RedRed/System#####

```
tagMSG: alias struct! [
    hWnd    [handle!]
    msg      [integer!]
    wParam   [integer!]
    lParam   [integer!]
    time     [integer!]
    x        [integer!]
    y        [integer!]
]

#import [
    "User32.dll" stdcall [
        CreateWindowEx: "CreateWindowExW" [
            dwExStyle    [integer!]
            lpClassName   [c-string!]
            lpWindowName  [c-string!]
            dwStyle       [integer!]
            x             [integer!]
            y             [integer!]
            nWidth        [integer!]
            nHeight       [integer!]
            hWndParent    [handle!]
            hMenu         [handle!]
            hInstance     [handle!]
            lpParam       [int-ptr!]
            return:       [handle!]
        ]
    ]
]
```

**6.** □□□□□□

[illegible]

- 時間戳時間戳 GMT 時間戳時間戳
- 時間戳時間戳Red時間戳時間戳API時間戳時間戳

## 7. Red/System

**Red/System**

Red

## 8. 関数

関数定義は、`Red/System` の `Red/System` モジュールで定義される。関数定義は、`Red/System` の `Red/System` モジュールで定義される。

関数

```
do-nothing: func [][]
increment: func [n [integer!]][n + 1]

increment: func [n [integer!]] [
    n + 1
]

increment: func [
    n [integer!]
][
    n + 1
]
```

関数

```
do-nothing: func [
][
]

do-nothing: func [
][
]

increment: func [
    n [integer!]
][n + 1]
```

関数定義は、`Red/System` の `Red/System` モジュールで定義される。関数定義は、`Red/System` の `Red/System` モジュールで定義される。関数定義は、`Red/System` の `Red/System` モジュールで定義される。関数定義は、`Red/System` の `Red/System` モジュールで定義される。

関数定義は、`Red/System` の `Red/System` モジュールで定義される。関数定義は、`Red/System` の `Red/System` モジュールで定義される。関数定義は、`Red/System` の `Red/System` モジュールで定義される。関数定義は、`Red/System` の `Red/System` モジュールで定義される。

```
make-world: func [
  earth    [word!]
  wind     [bitset!]
  fire     [binary!]
  water    [string!]
  /with
          thunder [url!]
  /only
  /into
          space   [block! none!]
  /local
  plants animals men women computers robots
][
  ...
]
```

```
make-world: func [
  [throw] earth [word!]          ;-- 地球
  wind    [bitset!]              ;-- 風
  fire [binary!]                  ;-- 火
  water   [string!]               ;-- 水
  /with
    thunder [url!]
  /only
  /into space [block! none!]      ;-- /with 宇宙
  /local
    plants animals                ;-- 植物
    men women computers robots
][
  ...
]
```

```
increment: func ["Add 1 to the argument value" n][n + 1]

make-world: func [
    "Build a new World"
    earth      [word!]      "1st element"
    wind       [bitset!]    "2nd element"
    fire       [binary!]    "3rd element"
    water      [string!]
    /with      "Additional element"
    thunder [url!]
    /only      "Not implemented yet"
    /into      "Provides a container"
    space [unset!]    "The container"
    /local
    plants animals men women computers robots
][
    ...
]
```

```
make-world: func ["Build a new World" ;-- 世界の構築
  earth [word!]      "1st element"
  wind  [bitset!]    "2nd element" ;-- 風
  fire  [binary!]
  "3rd element"      ;-- `fire` 火
  water [string!]
  /with              "Additional element"
    thunder [url!]
  /only "Not implemented yet" ;-- 未実装のドキュメント
  /into
    "Provides a container" ;-- 提供するコンテナ
    space [unset!] "The container"
  /local
    plants animals men women computers robots
][
  ...
]
```

[illegible]



```
foo arg1 arg2 arg3 arg4 arg5

process-many
  argument1
  argument2
  argument3
  argument4
  argument5
```

```
foo arg1 arg2 arg3
    arg4 arg5

foo
    arg1 arg2 arg3
    arg4 arg5

process-many
    argument1
        argument2
            argument3
                argument4
                    argument5
```

```
head insert (copy/part [1 2 3 4] 2) (length? mold (2 + index? find "Hello" #"o"))

head insert
  copy/part [1 2 3 4] 2
  length? mold (2 + index? find "Hello" #"o")
```

## Red

- `;;` 行コメント
- `157` `53`
- `comment {…}` ブロックコメント

