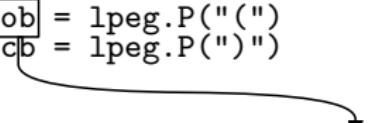


LParser is greedy

```
require "lpeg"  
  
ob = lpeg.P("(")  
cb = lpeg.P(")")  
  
pat = (1 - ob)^0 * ob * lpeg.P(1)^0 * cb  
  
txt = "_x abc (abc) "  
pat = lpeg.C(pat)  
x = lpeg.match(pat, txt)  
print (x)
```

LParser is greedy

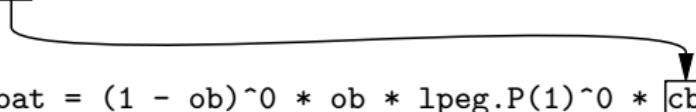
```
require "lpeg"  
ob = lpeg.P("(")  
cb = lpeg.P(")")  
  
pat = (1 - ob)^0 * ob * lpeg.P(1)^0 * cb
```



```
txt = "_x abc (abc) "  
pat = lpeg.C(pat)  
x = lpeg.match(pat, txt)  
print (x)
```

LPEG is greedy

```
require "lpeg"  
  
ob = lpeg.P("(")  
cb = lpeg.P(")")  
  
pat = (1 - ob)^0 * ob * lpeg.P(1)^0 * cb
```

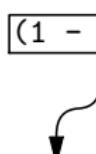


```
txt = "x abc (abc) "  
pat = lpeg.C(pat)  
x = lpeg.match(pat, txt)  
print (x)
```

LPeg is greedy

```
require "lpeg"  
  
ob = lpeg.P("(")  
cb = lpeg.P(")")  
  
pat = (1 - ob)^0 * ob * lpeg.P(1)^0 * cb  
  
txt = "x abc (abc) "  
pat = lpeg.C(pat)  
x = lpeg.match(pat, txt)  
print (x)
```

LPEG is greedy

```
require "lpeg"  
  
ob = lpeg.P("(")  
cb = lpeg.P(")")  
  
pat = [(1 - ob)^0] * ob * lpeg.P(1)^0 * cb  
  
txt = "x abc (abc) "  
pat = lpeg.C(pat)  
x = lpeg.match(pat, txt)  
print (x)
```

LPEG is greedy

```
require "lpeg"  
  
ob = lpeg.P("(")  
cb = lpeg.P(")")  
  
pat = [(1 - ob)^0] * ob * lpeg.P(1)^0 * cb  
  
txt = "x abc (abc) "  
pat = lpeg.C(pat)  
x = lpeg.match(pat, txt)  
print (x)
```

LParser is greedy

```
require "lpeg"  
  
ob = lpeg.P("(")  
cb = lpeg.P(")")  
  
pat = [(1 - ob)^0] * ob * lpeg.P(1)^0 * cb  
        
        
txt = "ux abc (abc) "  
pat = lpeg.C(pat)  
x = lpeg.match(pat, txt)  
print (x)
```

LParser is greedy

```
require "lpeg"  
  
ob = lpeg.P("(")  
cb = lpeg.P(")")  
  
pat = [(1 - ob)^0] * ob * lpeg.P(1)^0 * cb  
        
txt = "x abc (abc) "  
pat = lpeg.C(pat)  
x = lpeg.match(pat, txt)  
print (x)
```

LPEG is greedy

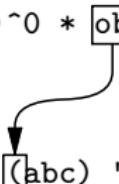
```
require "lpeg"  
  
ob = lpeg.P("(")  
cb = lpeg.P(")")  
  
pat = [(1 - ob)^0] * ob * lpeg.P(1)^0 * cb  
        
        
txt = "_x abc (abc) "  
pat = lpeg.C(pat)  
x = lpeg.match(pat, txt)  
print (x)
```

LPEG is greedy

```
require "lpeg"  
  
ob = lpeg.P("(")  
cb = lpeg.P(")")  
  
pat = [(1 - ob)^0] * ob * lpeg.P(1)^0 * cb  
  
txt = "x abc(abc) "  
pat = lpeg.C(pat)  
x = lpeg.match(pat, txt)  
print (x)
```

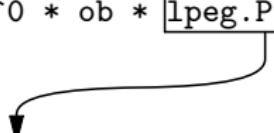
LParser is greedy

```
require "lpeg"  
  
ob = lpeg.P("(")  
cb = lpeg.P(")")  
  
pat = (1 - ob)^0 * ob * lpeg.P(1)^0 * cb  
  
txt = "x abc (abc) "  
  
pat = lpeg.C(pat)  
x = lpeg.match(pat, txt)  
print (x)
```



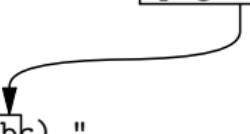
LPEG is greedy

```
require "lpeg"  
  
ob = lpeg.P("(")  
cb = lpeg.P(")")  
  
pat = (1 - ob)^0 * ob * lpeg.P(1)^0 * cb  
  
txt = "x abc (abc)"  
pat = lpeg.C(pat)  
x = lpeg.match(pat, txt)  
print (x)
```



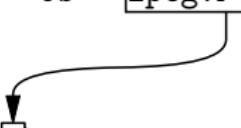
LPEG is greedy

```
require "lpeg"  
  
ob = lpeg.P("(")  
cb = lpeg.P(")")  
  
pat = (1 - ob)^0 * ob * lpeg.P(1)^0 * cb  
  
txt = "x abc (abc)"  
pat = lpeg.C(pat)  
x = lpeg.match(pat, txt)  
print (x)
```



LPEG is greedy

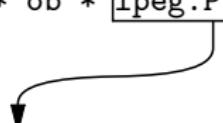
```
require "lpeg"  
  
ob = lpeg.P("(")  
cb = lpeg.P(")")  
  
pat = (1 - ob)^0 * ob * lpeg.P(1)^0 * cb  
  
txt = "_x abc (abc) "  
pat = lpeg.C(pat)  
x = lpeg.match(pat, txt)  
print (x)
```



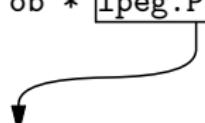
A curly brace originates from the expression `lpeg.P(1)^0` in the pattern definition. It points down to the character `c` in the string `_x abc (abc)` , specifically highlighting the part `(abc)`.

LPEG is greedy

```
require "lpeg"  
  
ob = lpeg.P("(")  
cb = lpeg.P(")")  
  
pat = (1 - ob)^0 * ob * lpeg.P(1)^0 * cb  
  
txt = "x abc (abc) "  
pat = lpeg.C(pat)  
x = lpeg.match(pat, txt)  
print (x)
```



LPEG is greedy

```
require "lpeg"  
  
ob = lpeg.P("(")  
cb = lpeg.P(")")  
  
pat = (1 - ob)^0 * ob * lpeg.P(1)^0 * cb  
  
  
txt = "x abc (abc)"  
pat = lpeg.C(pat)  
x = lpeg.match(pat, txt)  
print (x)
```

LPEG is greedy

```
require "lpeg"
```

```
ob = lpeg.P("(")
cb = lpeg.P(")")
```

```
pat = (1 - ob)^0 * ob * lpeg.P(1)^0 * cb
```

```
txt = "x abc (abc)"
```

```
pat = lpeg.C(pat)
x = lpeg.match(pat, txt)
print (x)
```

No Match